Continuous latent variable models for dimensionality reduction and sequential data reconstruction

by

Miguel Ángel Carreira-Perpiñán

Department of Computer Science
University of Sheffield, UK

February 2001

Dissertation submitted to the University of Sheffield for the degree of Doctor of Philosophy

Abstract

Continuous latent variable models (cLVMs) are probabilistic models that represent a distribution in a high-dimensional Euclidean space using a small number of continuous, latent variables. This thesis explores, theoretically and practically, the ability of cLVMs for dimensionality reduction and sequential data reconstruction.

The first part of the thesis reviews and extends the theory of cLVMs: definition in terms of a prior distribution in latent space, a mapping to data space and a noise model; maximum likelihood parameter estimation with an expectation-maximisation (EM) algorithm; specific cLVMs (factor analysis, principal component analysis (PCA), independent component analysis, independent factor analysis and the generative topographic mapping (GTM)); mixtures of cLVMs; identifiability, interpretability and visualisation; and derivation of mappings for dimensionality reduction and reconstruction and their properties, such as continuity, for each cLVM. We extend GTM to diagonal noise and give a corresponding EM algorithm.

We also describe a discrete LVM for binary data, Bernoulli mixtures, widely used in practice. We show that their log-likelihood surface has no singularities, unlike other mixture models, which makes EM estimation practical; and that their theoretical non-identifiability is rarely realised in actual estimates, which makes them interpretable.

The second part deals with dimensionality reduction. We define the problem and give an extensive, critical review of nonprobabilistic methods for it: linear methods (PCA, projection pursuit), nonlinear autoassociators, kernel methods, local dimensionality reduction, principal curves, vector quantisation methods (elastic net, self-organising map) and multidimensional scaling methods. We then empirically evaluate, in terms of reconstruction error, computation time and visualisation, several latent-variable methods for dimensionality reduction of binary electropalatographic (EPG) data: PCA, factor analysis, mixtures of factor analysers, GTM and Bernoulli mixtures. We compare these methods with earlier, nonadaptive EPG data reduction methods and derive 2D maps of EPG sequences for use in speech research and therapy.

The last part of this thesis proposes a new method for missing data reconstruction of sequential data that includes as particular case the inversion of many-to-one mappings. We define the problem, distinguish it from inverse problems, and show when both coincide. The method is based on multiple pointwise reconstruction and constraint optimisation. Multiple pointwise reconstruction uses a Gaussian mixture joint density model for the data, conveniently implemented with a nonlinear cLVM (GTM). The modes of the conditional distribution of missing values given present values at each point in the sequence represent local candidate reconstructions. A global sequence reconstruction is obtained by efficiently optimising a constraint, such as continuity or smoothness, with dynamic programming. We give a probabilistic interpretation of the method. We derive two algorithms for exhaustive mode finding in Gaussian mixtures, based on gradient-quadratic search and fixed-point search, respectively; as well as estimates of error bars for each mode and a measure of distribution sparseness. We discuss the advantages of the method over previous work based on the conditional mean or on universal mapping approximators (including ensembles and recurrent networks), conditional distribution estimation, vector quantisation and statistical analysis of missing data. We study the performance of the method with synthetic data (a toy example and an inverse kinematics problem) and real data (mapping between EPG and acoustic data). We describe the possible application of the method to several well-known reconstruction or inversion problems: decoding of neural population activity for hippocampal place cells; wind field retrieval from scatterometer data; inverse kinematics and dynamics of a redundant manipulator; acoustic-to-articulatory mapping; audiovisual mappings for speech recognition; and recognition of occluded speech.

Keywords: continuous latent variable models, generative models, generative topographic mapping (GTM), identifiability, Bernoulli mixtures, electropalatography, dimensionality reduction, inverse problems, missing data reconstruction, sequential data reconstruction, mapping inversion, conditional distribution, mode finding, continuity constraints, distribution sparseness, universal mapping approximators, vector quantisation, inverse kinematics, acoustic-to-articulatory mapping.

Preface

Some chapters of this thesis are largely self-contained and can be read without regard to the rest of the thesis, particularly chapter 8 (mode finding in Gaussian mixtures) and also chapters 2 (continuous latent variable models) and 4 (dimensionality reduction with non-probabilistic models). I have tried to keep the notation over the whole thesis as uniform as possible, sometimes at the expense of usual conventions in other fields. I encourage the reader to turn to the notation list and glossary in case of doubt. Occasionally, I have also included examples to illustrate the meaning and significance of the ideas presented, because the abstract, generic character of pattern recognition models and algorithms can sometimes give the impression of lacking practical applicability. Most figures should be readable when printed in black and white except perhaps some of those in chapters 9 and 10.

As I wrote this thesis, in particular the painstaking literature reviews, I have had many times the impression that any idea that at first seemed very new and original was actually a reelaboration of a previously known idea (although perhaps not well known). The contributions of this thesis are no exception. The fact that the development of new ideas depends so heavily on all the previous, inherited ideas and the relative smallness of the individual contributions make it difficult, in my opinion, to claim sole authorship of a new idea. Newton may have stood upon the shoulders of giants, but nowadays scientists are more like ants standing upon an anthill.

By a caprice of fortune, I am writing these lines the 20th anniversary of the infamous coup d'etat of 23-F in Madrid, which tried to end the then very young Spanish democracy that had succeeded Franco's dictatorship. I still remember the unusually deserted streets that afternoon.

I have no control on the future use (if any) by others of the ideas of this dissertation, but I hope they will only be used towards peaceful ends.

Miguel Á. Carreira-Perpiñán
Washington, D.C. February 23, 2001

Acknowledgements

First of all, I am grateful to my supervisor, Steve Renals, for having guided me with an open but practical mind, for having inspired me to look into some very interesting problems, for being always willing to answer questions or discuss problems and for infusing me with his intellectual honesty.

I thank Phil Green for giving me the opportunity to do my PhD in the Speech and Hearing research group of the Department of Computer Science and for his support throughout my stay in it. I have enjoyed technical and social discussions with many people in the group, specially Dave Abberley, Jon Barker, Guy Brown, Martin Cooke, Yoshi Gotoh, Phil Green, Ljubomir Josifovski, Konstantinos Koumpis, Simon Makin, Andy Morris, Mahesan Niranjan, Vinny Wan and Gethin Williams.

I thank the following people for reading and commenting on parts of the thesis manuscript: Jon Barker, Geoff Goodhill, Ljubomir Josifovski and Konstantinos Koumpis. Thanks also to Geoff Goodhill for his support to finish writing up the thesis while I was a postdoc in his lab.

I acknowledge the following people for, at some moment or other, having provided me with useful feedback or comments: Zoubin Ghahramani, Alexander Heimel, Michael Jordan, Lawrence Saul and Chris Williams. I also acknowledge the comments of anonymous referees in the work of chapters 3 and 8.

I am also grateful to Alan Wrench for providing me with the ACCOR data, that has been the basis for many of my experiments.

Some of the ideas presented in this thesis were stimulated by my attendance to three excellent courses: the NATO Advanced Study Institute *Learning in Graphical Models* (Ettore Majorana Centre, Erice, Italy, 1996); the NATO Advanced Study Institute *Generalization in Neural Networks and Machine Learning* (Isaac Newton Institute for Mathematical Sciences, Cambridge, UK, 1997); and the EU *Advanced Course in Computational Neuroscience* (International Centre for Theoretical Physics, Trieste, Italy, 1999). Financial support from the respective organisations is gratefully acknowledged.

I gratefully acknowledge financial support from the following bodies: the Spanish Ministry of Education and Science; the University of Sheffield; the European Union, ESPRIT Long Term Research Project SPRACH (20077); the Nuffield Foundation; and a travel grant from the NIPS foundation.

Finally, thanks to my parents, my brother and my family for their love in the long distance and to Min for bearing with me in the short distance and for the inspiring courage she showed during her own PhD studies.

Publications resulting from this thesis

The following publications have been produced during the course of this thesis. They are available online at `http://www.dcs.shef.ac.uk/~miguel` or directly from the author at `miguel@dcs.shef.ac.uk`, as well as the thesis itself and some Matlab software (see appendix C). A journal paper based on part III of the thesis is in preparation.

- M. Á. Carreira-Perpiñán. Mode-finding for mixtures of Gaussian distributions. *IEEE Trans. on Pattern Anal. and Machine Intel.*, 22(11): 1318–1323, Nov. 2000.
- M. Á. Carreira-Perpiñán. Reconstruction of sequential data with probabilistic models and continuity constraints. In S. A. Solla, T. K. Leen, and K.-R. Müller, editors, *Advances in Neural Information Processing Systems*, volume 12, pages 414–420. MIT Press, Cambridge, MA, 2000.
- M. Á. Carreira-Perpiñán and S. Renals. Practical identifiability of finite mixtures of multivariate Bernoulli distributions. *Neural Computation*, 12(1): 141–152, Jan. 2000.
- M. Á. Carreira-Perpiñán. One-to-many mappings, continuity constraints and latent variable models. In *Proc. of the IEE Colloquium on Applied Statistical Pattern Recognition*, Birmingham, UK, 1999.
- M. Á. Carreira-Perpiñán and S. Renals. A latent variable modelling approach to the acoustic-to-articulatory mapping problem. In J. J. Ohala, Y. Hasegawa, M. Ohala, D. Granville, and A. C. Bailey, editors, *Proc. of the 14th International Congress of Phonetic Sciences (ICPhS'99)*, pages 2013–2016, San Francisco, USA, Aug. 1–7 1999.
- M. Á. Carreira-Perpiñán. Mode-finding for mixtures of Gaussian distributions. Technical Report CS–99–03, Dept. of Computer Science, University of Sheffield, UK, Mar. 1999 (revised August 4, 2000).
- M. Á. Carreira-Perpiñán and S. Renals. Dimensionality reduction of electropalatographic data using latent variable models. *Speech Communication*, 26(4): 259–282, Dec. 1998.
- M. Á. Carreira-Perpiñán and S. Renals. Experimental evaluation of latent variable models for dimensionality reduction. In M. Niranjan, editor, *Proc. of the 1998 IEEE Signal Processing Society Workshop on Neural Networks for Signal Processing (NNSP98)*, pages 165–173, Cambridge, UK, Sept. 1998.
- M. Á. Carreira-Perpiñán. Density networks for dimension reduction of continuous data: Analytical solutions. Technical Report CS–97–09, Dept. of Computer Science, University of Sheffield, UK, Apr. 1997.
- M. Á. Carreira-Perpiñán. A review of dimension reduction techniques. Technical Report CS–96–09, Dept. of Computer Science, University of Sheffield, UK, Dec. 1996.

Contents

List of Figures

List of Tables

Notation

Generally, we denote scalars in italics (m, x, α), vectors in lowercase boldface ($\mathbf{x}$, $\boldsymbol{\mu}$), matrices in uppercase boldface ($\mathbf{A}$, $\mathbf{\Lambda}$) and sets in uppercase calligraphic ($\mathcal{X}$) or blackboard face ($\mathbb{R}$). We will use the notation $p(\mathbf{x})$ to mean either the probability that a discrete random variable X takes the value $\mathbf{x}$, $\Pr[X = \mathbf{x}]$, or the density of a continuous random variable X at the value $\mathbf{x}$, unless the context would make the notation ambiguous. Further definitions are given below in alphabetical order. In exceptional cases, some of these definitions may be overriden.

$\stackrel{\text{def}}{=}$	By definition equal to.
$'$ (prime)	It usually indicates a new variable related by some transformation to an old variable, e.g. $\mathbf{t}' = \mathbf{A}\mathbf{t}$. It can also mean the derivative of a function.
$\lvert\cdot\rvert$	The absolute value of a real number or the determinant of a matrix.
$\lVert\cdot\rVert$ or $\lVert\cdot\rVert_2$	The Euclidean, or sum of squares, or L_2 norm: $\lVert\mathbf{t}\rVert_2^2 \stackrel{\text{def}}{=} \sum_{d=1}^{D} x_d^2$; or, for a density p: $\lVert p\rVert_2^2 \stackrel{\text{def}}{=} \int p^2(\mathbf{t})\,d\mathbf{t} = \mathrm{E}_{p(\mathbf{t})}\left\{p(\mathbf{t})\right\}$.
$\lVert\cdot\rVert_\infty$	The maximum norm: $\lVert\mathbf{t}\rVert_\infty \stackrel{\text{def}}{=} \max_{d=1,\dots,D}\{\lvert x_d\rvert\}$.
$\lfloor x\rfloor$	The closest integer smaller or equal than x.
$\lceil x\rceil$	The closest integer greater or equal than x.
$\langle p, q\rangle$	Scalar product of densities p, q: $\langle p, q\rangle \stackrel{\text{def}}{=} \int p(\mathbf{t})q(\mathbf{t})\,d\mathbf{t}$.
$\mathbf{1} = (1, 1, \dots, 1)^T$	A vector of ones of the relevant dimension.
$\mathbf{A}^T$	The transpose of matrix $\mathbf{A}$.
$\mathbf{A}^+$	The pseudoinverse of matrix $\mathbf{A}$.
$\mathcal{B}(\mathbf{p})$	A (multivariate) Bernoulli distribution of parameter $\mathbf{p}$.
$\mathscr{C}$	A continuity constraint.
$\mathbb{C}_R^D$	D-hypercube centred in the origin with side $2R$: $\mathbb{C}_R^D = \{\mathbf{x} \in \mathbb{R}^D : \lVert\mathbf{x}\rVert_\infty \le R\} = [-R, R]^D$ (for the hollow hypercube replace $\le$ by $=$).
D	Dimension of the data space $\mathcal{T}$. Index variable: $d = 1, \dots, D$.
$d(\mathbf{x}, \mathbf{y})$	A distance between vectors $\mathbf{x}$ and $\mathbf{y}$ (e.g. the Euclidean distance).
$\mathrm{D}\left(p\Vert q\right)$	The Kullback-Leibler distance (or directed divergence, or relative entropy) from p to q, $\mathrm{D}\left(p\Vert q\right) \stackrel{\text{def}}{=} \mathrm{E}_p\left\{\ln\frac{p(\mathbf{x})}{q(\mathbf{x})}\right\}$, where p and q are probability mass or density functions.
δ_{ij}, $\delta(\mathbf{x})$	The Dirac delta function for discrete variables i and j, and for (multivariate) continuous variable $\mathbf{x}$, respectively.
δ_{nm}	In multidimensional scaling, the pairwise proximity between items n and m (e.g. the distance between data points $\mathbf{t}_n$ and $\mathbf{t}_m$).
$\Delta(\mathbf{x}, \mathbf{y})$	A discrete derivative or gradient of some objective function.
$\mathrm{diag}\left(\mathbf{A}\right)$	The "diag" operator sets all the off-diagonal elements of a matrix to zero.
$\mathrm{diag}\left(\lambda_1, \dots, \lambda_D\right)$	A diagonal matrix with $\lambda_1, \dots, \lambda_D$ in the diagonal.
ϵ	A small positive value: $0 < \epsilon \ll 1$.
$\mathbf{e}$	Zero-mean random variable added to another random variable to represent an error.
$\mathrm{E}_{p(\mathbf{t})}\left\{\mathbf{f}(\mathbf{t})\right\}$	Mean of $\mathbf{f}(\mathbf{t})$ with respect to the distribution of $\mathbf{t}$: $\mathrm{E}_{p(\mathbf{t})}\left\{\mathbf{f}(\mathbf{t})\right\} \stackrel{\text{def}}{=} \int \mathbf{f}(\mathbf{t})p(\mathbf{t})\,d\mathbf{t}$.
F	Number of basis functions in the latent space for GTM. Index variable: $f = 1, \dots, F$.
ϕ, Φ	The pdf and cdf of the univariate normal distribution.
$\mathfrak{F}$	A continuous function of the experimental conditions variables $\mathbf{z}$.
$\mathscr{F}$	A smoothness constraint.
$\mathcal{F}\{\cdot\}$	The Fourier transform. Also, the feature space in kernel PCA.
$\mathbf{f}$	The reconstruction mapping from latent or reduced-dimension representation space $\mathcal{X}$ to data space $\mathcal{T}$.
$\mathbf{f} \circ \mathbf{g}$	The composition of functions $\mathbf{f}$ and $\mathbf{g}$, $\mathbf{f}(\mathbf{g}(\cdot))$.

$\mathbf{F}$	The dimensionality reduction mapping from data space $\mathcal{T}$ to latent or reduced-dimension representation space $\mathcal{X}$.
$\mathbf{g}$	A forward mapping, i.e., a univalued mapping with a multivalued inverse. A forward mapping typically expresses a causal relation.
$\mathbf{g}$, $\mathbf{g}(\mathbf{t})$	Gradient vector (of the density $p(\mathbf{t})$ with respect to the independent variable $\mathbf{t}$).
$h(p)$	Differential entropy of density p: $h(p) \stackrel{\text{def}}{=} \mathrm{E}\left\{-\ln p\right\} = -\int p(\mathbf{t}) \ln p(\mathbf{t})\, d\mathbf{t}$.
$\mathbf{H}$, $\mathbf{H}(\mathbf{t})$	Hessian matrix (of the density $p(\mathbf{t})$ with respect to the independent variable $\mathbf{t}$).
$\mathbb{H}^D$	Half-space of $\mathbb{R}^D$: $\mathbb{H}^D = \{\mathbf{x} \in \mathbb{R}^D :\ x_D \geq 0\}$.
I	A projection pursuit index.
$I_{\mathcal{M}}(x)$	Indicator function, $I_{\mathcal{M}}(x) = \begin{cases} 1 & x \in \mathcal{M} \\ 0 & \text{otherwise} \end{cases}$.
$\mathbf{I}_D$	$D \times D$ identity matrix.
$\operatorname{im} \mathbf{f}$, $\operatorname{im} \mathbf{A}$	The image or range space of a mapping $\mathbf{f}$ or matrix $\mathbf{A}$, respectively: $\operatorname{im} \mathbf{f} \stackrel{\text{def}}{=} \{\mathbf{t} \in \mathcal{T} :\ \mathbf{t} = \mathbf{f}(\mathbf{x}) \text{ for some } \mathbf{x} \in \mathcal{X}\} = \mathbf{f}(\mathcal{X})$.
K	Number of samples of a Monte Carlo approximation of an integral (in particular, number of latent grid points for GTM). Index variable: $k = 1, \dots, K$.
$\ker \mathbf{A}$	The kernel or null space of a matrix $\mathbf{A}$: $\ker \mathbf{A} \stackrel{\text{def}}{=} \{\mathbf{x} \in \mathcal{X} :\ \mathbf{A}\mathbf{x} = \mathbf{0}\}$.
L	Dimension of the latent space $\mathcal{X}$. Index variable: $l = 1, \dots, L$.
$L(\mathbf{x})$	The logarithm of the density $p(\mathbf{x})$, $L(\mathbf{x}) \stackrel{\text{def}}{=} \ln p(\mathbf{x})$.
$\mathcal{L}(\mathbf{\Theta})$	The log-likelihood function (of some density model given a certain sample) of parameters $\mathbf{\Theta} = \theta_1, \theta_2, \dots$
$\mathcal{L}_2$	The space of square-integrable functions.
L_n norm	The norm $\left\lVert\mathbf{t}\right\rVert_n \stackrel{\text{def}}{=} \left(\sum_{d=1}^D \lvert t\rvert_d^n\right)^{1/n}$; or, for a density p: $\left\lVert p\right\rVert_n \stackrel{\text{def}}{=} \left(\int p^n(\mathbf{t})\, d\mathbf{t}\right)^{1/n} = \left(\mathrm{E}_{p(\mathbf{t})}\left\{p^{n-1}(\mathbf{t})\right\}\right)^{1/n}$.
M	Number of components of a mixture model. Index variable: $m = 1, \dots, M$.
$\boldsymbol{\mu}$	Location parameter of a model. Also, centroid or mean vector or reference vector in data space for mixture models, vector quantisation and Kohonen's self-organising maps.
N	Number of points of a data set $\{\mathbf{t}_n\}_{n=1}^N$. Index variable: $n = 1, \dots, N$.
$\mathcal{N}_D(\boldsymbol{\mu}, \mathbf{\Sigma})$	D-dimensional normal distribution of mean $\boldsymbol{\mu}$ and covariance matrix $\mathbf{\Sigma}$.
$\mathbb{N}$, $\mathbb{R}$	The sets of natural and real numbers, respectively. By convention, $\mathbb{N}$ does not include zero.
$\mathcal{P}$, $\mathcal{M}$	Sets of indices for present and missing variables, respectively.
$\sigma(x) = \frac{1}{1+e^{-x}}$	Sigmoid function.
$\mathbf{S}$	Sample covariance matrix of a data set $\{\mathbf{t}_n\}_{n=1}^N$: $\mathbf{S} \stackrel{\text{def}}{=} \frac{1}{N}\sum_{n=1}^N (\mathbf{t}_n - \overline{\mathbf{t}})(\mathbf{t}_n - \overline{\mathbf{t}})^T$.
$\mathbf{\Sigma}$	Covariance matrix parameter of a model.
$\mathbf{t}$	D-dimensional data variable.
$\overline{\mathbf{t}}$	Sample mean of a data set $\{\mathbf{t}_n\}_{n=1}^N$: $\overline{\mathbf{t}} \stackrel{\text{def}}{=} \frac{1}{N}\sum_{n=1}^N \mathbf{t}_n$.
τ	Iteration index (e.g. for EM algorithms); time.
$\mathbf{t}_{\mathcal{P}} \to \mathbf{t}_{\mathcal{M}}$	$\mathbf{t}_{\mathcal{M}}$ as a function of $\mathbf{t}_{\mathcal{P}}$.
$\{\mathbf{t}_n\}_{n=1}^N$	Set of N vectors not necessarily in sequence.
$\{\mathbf{t}^{(n)}\}_{n=1}^N$	Set of N vectors in (temporal) sequence.
$\mathcal{T}$	Data space. Also, the input space in kernel PCA.
θ	A parameter of a model; a threshold value.
$\mathbf{\Theta}$	The parameters of a model, $\mathbf{\Theta} = \theta_1, \theta_2, \dots$
$\operatorname{tr}(\mathbf{A})$	Trace of the matrix $\mathbf{A}$.
$\mathcal{U}_D(\mathcal{R})$	D-dimensional uniform distribution over the hyperrectangle $\mathcal{R}$, e.g. $\mathcal{U}(a, b)$ for the interval $[a, b]$.
$\mathbf{x}$	L-dimensional latent variable.
$\mathcal{X}$	Latent space.
$\mathbf{z}$	Experimental conditions variables.

Glossary

ASR	Automatic speech recognition.
BCM	Bienenstock-Cooper-Munro model of neuron selectivity.
branch	A submanifold of the trace of a forward function $\mathbf{g}$ where $\mathbf{g}$ is invertible (one-to-one).
CART	Classification and regression trees.
cdf	Cumulative distribution function.
EMA	Electromagnetic articulography.
EPG	Electropalatography.
FA	Factor analysis.
FIR	Finite impulse response (filter).
ICA	Independent component analysis.
iid	Independent identically distributed.
IFA	Independent factor analysis.
GMM	Gaussian mixture model.
GTM	Generative topographic mapping.
HMM	Hidden Markov model.
IPA	International Phonetic Association.
LPC	Linear predictive coding.
LSP	Line spectra pairs (a type of features extracted from the speech waveform).
MAP	Maximum a posteriori Probability.
MAR	Missing at random.
MARS	Multivariate adaptive regression splines.
MCAR	Missing completely at random.
MDS	Multidimensional scaling.
MFCC	Mel-frequency cepstral coefficients (a type of features extracted from the speech waveform).
ML	Maximum likelihood.
MLP	Multilayer perceptron.
PCA	Principal component analysis.
pdf	Probability density function.
phonemic vs phonetic transcription	A phoneme is an abstract unit or category of the phonetic system of a language and corresponds to several speech sounds which are perceived to be a single distinctive sound in the language but which are phonetically different (e.g. allophones, as the velar /k/ of ‘cool’ and the palatal /k/ of ‘keel’). Conventionally, phonemic transcriptions are enclosed between slashes while phonetic transcriptions are enclosed between square brackets, such as /fəˈnɛtɪks/ and [fəə̃ˈnɛtˢɪʔks], respectively, for the word ‘phonetics’.
PLP	Perceptual linear predictive coefficients (a type of features extracted from the speech waveform).
RBF	Radial basis function: $g(\mathbf{x}) = k\left(\frac{\lVert\mathbf{x}-\mathbf{a}\rVert}{b}\right)$ with centre $\mathbf{a}$, smoothing factor b and kernel function k (integrable, with integral different from 0, e.g. Gaussian).
ridge function	$g(\mathbf{x}) = \sigma(\mathbf{a}^T\mathbf{x} + \mathbf{b})$ with direction $\mathbf{a}$, threshold $\mathbf{b}$ and nonlinearity σ (typically sigmoid, e.g. the logistic function).
RMS	Root mean square (error).
shuffled	A data set $\{\mathbf{t}_n\}_{n=1}^N$ where the indices $n = 1, \dots, N$ have been randomly permuted, so that any correlation based on the ordering of the original data (e.g. proximity of consecutive points) has been destroyed.

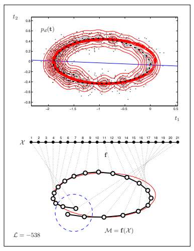
sigmoid	Function $\sigma : \mathbb{R} \to \mathbb{R}$ with $\lim_{x\to-\infty} \sigma(x) = 0$ and $\lim_{x\to+\infty} \sigma(x) = 1$, e.g. the logistic function $f(x) \stackrel{\text{def}}{=} \frac{1}{1+e^{-x}}$.
SOM	Self-organising map.
TIMIT	Texas Instruments and MIT acoustic phonetic database.
UMA	Universal mapping approximator.
VCV	Vowel-consonant-vowel cluster.
WER	Word error rate (percentage of words in the reference transcript that an ASR system incorrectly reported: insertions, substitutions and deletions).

Chapter 1

Introduction

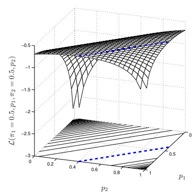
This thesis deals with the application of continuous latent variable models—a general, powerful class of probabilistic models—to two important pattern recognition problems: dimensionality reduction and sequential data reconstruction. The theoretical and practical investigation of each of these three issues, the model class and the two problems, makes up each of the three parts that the thesis consists of.

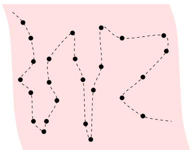
Dimensionality reduction is the process by which we represent a system that appears as having several degrees of freedom using a smaller number of degrees of freedom. For example, the position of an aeroplane with respect to the centre of the Earth is a three-dimensional vector (x, y, z), but we can conveniently represent it with only two variables, latitude and longitude, and consider all radial variation as noise. In a statistical learning or pattern recognition approach, such representation is not deduced by analysis but inferred or learned from observed data from the system: positions (x, y, z) where the plane has been observed, in our case. Continuous latent variable models are able to represent such a change of coordinates (the low-dimensional region where the aeroplane is constrained to be), to account for the amount in which the aeroplane can deviate out of that region (the noise) and to determine how likely each part of the region is to contain the aeroplane. Dimensionality reduction has applications such as feature extraction, visualisation and exploratory data analysis. Sequential data reconstruction consists of, given a sequence of vectors where there are some values that have gone missing, filling in those values to recover the original sequence. For the example of the aeroplane, imagine the pilot transmits its position (x, y, z) at regular intervals but that occasionally some of the xs or ys or zs do not reach the receiver due to interference. Can we recover the aeroplane trajectory? This is the problem of missing data reconstruction for sequential data.



In part I (chapters 2–3) we review and extend the theory of latent variable models. Chapter 2 concentrates on continuous latent variable models. These are parametric probabilistic models that represent a distribution in a high-dimensional Euclidean space using a small number of continuous, latent variables. Their aim is, given observed high-dimensional data, to find a low-dimensional manifold (a coordinate transformation) where the data would live if there was no noise, and to model the noise itself. This is achieved by the three components that define a continuous latent variable model: a prior distribution in the latent space, a smooth mapping from latent to data space and a noise model. Choosing functional forms for these gives specific latent variable models; for example, a linear mapping and normal prior and noise give the well-known factor analysis model. Unfortunately, not every choice leads to models that can be handled analytically and to convenient algorithms for parameter estimation. This is due to the difficulty of integrating functions of many variables. We discuss what choices are feasible and then describe all known types of latent variable models: factor analysis, principal component analysis, independent component analysis, independent factor analysis and the generative topographic mapping (GTM). Maximum likelihood parameter estimation of latent variable models can usually be done with an EM algorithm, whose simplicity and monotonic convergence makes it very convenient. We show how to construct and estimate finite mixtures of latent variable models and their relation to Gaussian mixtures. We discuss the interpretability of the latent variables; the identifiability of each specific latent variable model, i.e., whether different parameter estimates correspond to identical observed distributions; and the issue of visualising the latent variable representation. We then turn to dimensionality reduction with latent variable models. A dimensionality reduction mapping can be defined from observed to latent variables via Bayes' theorem; we discuss, for every latent variable model as well as for mixtures of them, some properties of this mapping, including its continuity and why it is not the inverse of the mapping from latent to observed space. Finally, we illustrate several of the previous concepts with a worked example of a latent variable model. We complement the chapter with appendices giving details on some mathematical properties of latent variable models; and we extend the GTM model to diagonal noise and give a corresponding EM algorithm. Such a model is useful when the observed variables have different local scales of dispersion, which cannot be accounted for by sphering the data.

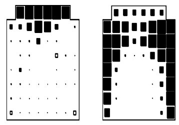
In chapter 3 we consider a discrete latent variable model that we use in chapter 5: finite mixtures of multivariate Bernoulli distributions. This model, widely used for binary data, is non-identifiable (different values for the parameters can give rise to exactly the same distribution), which casts doubts upon the interpretability of parameter estimates—since they may not be unique. We give empirical evidence, with synthetic and real data, that the theoretical non-identifiability of Bernoulli mixtures is rarely realised in actual estimates, which can then be confidently interpreted. We also show that—unlike for other mixture models—the log-likelihood surface of Bernoulli mixtures has no singularities of infinite value and therefore that EM estimation is practical, always converging to a proper maximum likelihood estimate (chapters 5 and 9 show how singularities do affect the estimation of mixtures of factor analysers and Gaussian mixtures, respectively).





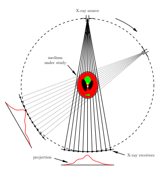
Part II (chapters 4–5) deals with dimensionality reduction. Chapter 4 is an extensive, critical review of nonprobabilistic dimensionality reduction methods, that we compare to latent variable models. We first define the problem of dimensionality reduction, propose a classification of dimensionality reduction problems, show how the geometry of high-dimensional Euclidean spaces gives rise to the curse of the dimensionality and discuss how the concept of intrinsic dimension of a sample is based on a-priori expectations. We then review linear methods (PCA, projection pursuit), nonlinear autoassociators, kernel methods, local dimensionality reduction, principal curves, vector quantisation methods (elastic net, self-organising map) and multidimensional scaling methods (based on straight-line or geodetic distances). We close the chapter with a digression on dimensionality reduction with discrete variables.

In chapter 5 we empirically evaluate, in terms of reconstruction error, computation time and visualisation, several latent-variable methods for dimensionality reduction of electropalatographic (EPG) data, which are binary vectors: PCA, factor analysis, mixtures of factor analysers, GTM and Bernoulli mixtures. We compare the results of these methods with those of earlier, nonadaptive EPG data reduction methods and derive two-dimensional maps that can be used for the analysis of EPG sequences in speech research and therapy. We also show that the intrinsic dimensionality of the EPG data may be quite lower than that suggested by earlier studies.



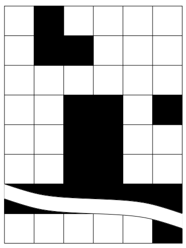
$\mathbf{\Lambda}_4$ $\boldsymbol{\mu}_4$

$\pi_4 = 0.1721$



Part III (chapters 6–10) of the thesis proposes a new method for missing data reconstruction of sequential data inspired by latent variable models. Missing data reconstruction is a very general problem that includes as special cases mapping approximation (regression) and mapping inversion (of many-to-one mappings). The latter is not generally the same as what are properly called inverse problems. Chapter 6 is devoted to briefly reviewing inverse problem theory and its methods (Bayesian and nonprobabilistic), to relate Bayesian inverse problem theory to latent variable models and to examine when an inverse problem is reducible to a mapping inversion problem.

Chapter 7 defines the problem of data reconstruction of a sequence, describes the method proposed and discusses related methods. The phenomenon of missing data happens when, in a sequence of real-valued vectors, some components of some vectors are unobserved, deleted or somehow unavailable. For example, in regression the predictor variables are always present and the response ones always missing. The problem is then to fill in the missing values so that the resulting sequence is as close to the original one as possible. The difficulty is that, at a given vector, the present values usually do not uniquely determine the missing ones; but, if the variation of the vectors along the sequence is continuous, neighbouring vectors are constrained to be close and this can be used to determine the missing data. This suggests the two ideas on which the proposed method is based: multiple pointwise reconstruction and constraint optimisation. Multiple

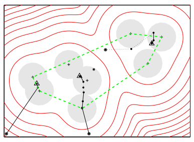


Missing data pattern (mask)

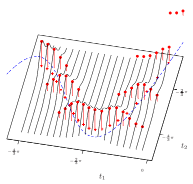
pointwise reconstruction consists of finding the values of the missing variables that are compatible with the values of the present variables, and results in a set of candidate reconstructions for each vector in the sequence; in other words, it represents multivalued mappings. We propose the use of the modes of the conditional distribution of the missing variables given the present ones as such local candidate reconstructions. We investigate how to quantify the sparseness of a conditional distribution: the degree to which the conditioned-on variables determine the values of the other variables. The flexible construction of arbitrary conditional distributions requires a joint density model for the data. We implement the joint density model with a continuous latent variable model (GTM) that results in a Gaussian mixture: this allows multimodal conditional distributions (corresponding to multivalued mappings) which can be efficiently computed. Once we have the candidate reconstructions of each vector in the sequence, a global sequence reconstruction is obtained by optimising a continuity constraint using dynamic programming. We show how to extend the constraint to several dimensions (e.g. a continuous field rather than a continuous sequence) and to other constraint types, such as smoothness; and we give a probabilistic interpretation of the method.

We describe the possible future application of the method to several well-known reconstruction or inversion problems: decoding of neural population activity for hippocampal place cells; wind field retrieval from scatterometer data; inverse kinematics and dynamics of a redundant manipulator; the acoustic-to-articulatory mapping; audiovisual mappings for speech recognition; and recognition of occluded speech.

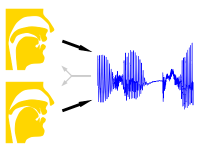
Several of the basic ideas of our method, or related ideas, have been used in different contexts earlier in the literature and so we give a careful review of such approaches, comparing them to ours. They include work based on the conditional mean or on universal mapping approximators (including ensembles and recurrent networks), conditional distribution estimation, vector quantisation and statistical analysis of missing data.



Our multiple pointwise reconstruction approach requires finding all the modes of a Gaussian mixture. This problem is solved in chapter 8 for a subclass of Gaussian mixtures. We derive two mode-finding algorithms based on gradient-quadratic search and fixed-point search, respectively, as well as estimates of the error bars for each mode, based on a normal approximation with the local Hessian. Apart from their use in our reconstruction method, these algorithms are generally applicable to other problems, such as clustering (e.g. determination of subclustering within galaxy systems) or Bayesian analysis (scrutiny of the posterior modes). The mode-finding algorithms mildly rely on the conjecture that the number of modes of a Gaussian mixture (of the subclass mentioned) is at most the number of components in the mixture and that the modes lie in the convex hull of the components. We prove this conjecture for a very particular case only; the general case remains unproven. The chapter is complemented with a mathematical appendix which gives, among other results, bounds for the gradient, Hessian and entropy of a Gaussian mixture.



The next two chapters evaluate the performance of our method and of other methods, notably the conditional mean and neural networks. Chapter 9 uses two synthetic-data, low-dimensional examples of forward mappings with multivalued inverses: a toy problem, which allows the visualisation of the behaviour of the method; and an inverse kinematics problem for a two-link, planar robot arm (a prototypical case of mapping inversion). The method is shown to recover the original sequence even when many of the values are missing, whether the underlying mapping is one-to-one or one-to-many (which causes the other methods to fail). The examples also show the sensitivity, for regression problems, of the method to a nonsmooth joint density model.



Chapter 10 evaluates the method with real, high-dimensional data: the mapping between acoustic waveform and EPG frames. This problem is related to the acoustic-to-articulatory mapping problem of speech research, which we review, emphasising its potential role in improving automatic speech recognition. We show the difference between the acoustic-EPG mapping and the acoustic-to-articulatory mapping. The experimental results confirm that the method works well for arbitrary missing data patterns but is sensitive to the density model quality for regression problems. We also give computation times for the reconstruction.

The main ideas and contributions of the thesis are summarised in chapter 11, which also suggests directions for further research.

Part I

Theory of continuous latent variable models and Bernoulli mixtures

Chapter 2

The continuous latent variable modelling formalism

This chapter gives the theoretical basis for continuous latent variable models. Section 2.1 defines intuitively the concept of latent variable models and gives a brief historical introduction to them. Section 2.2 uses a simple example, inspired by the mechanics of a mobile point, to justify and explain latent variables. Section 2.3 gives a more rigorous definition, which we will use throughout this thesis. Section 2.6 describes the most important specific continuous latent variable models and section 2.7 defines mixtures of continuous latent variable models. The chapter discusses other important topics, including parameter estimation, identifiability, interpretability and marginalisation in high dimensions. Section 2.9 on dimensionality reduction will be the basis for part II of the thesis. Section 2.10 very briefly mentions some applications of continuous latent variable models for dimensionality reduction. Section 2.11 shows a worked example of a simple continuous latent variable model. Section 2.12 give some complementary mathematical results, in particular the derivation of a diagonal noise GTM model and of its EM algorithm.

2.1 Introduction and historical overview of latent variable models

Latent variable models are probabilistic models that try to explain a (relatively) high-dimensional process in terms of a few degrees of freedom. Historically, the idea of latent variables arose primarily from psychometrics, beginning with the g factor of Spearman (1904) and continuing with other psychologists such as Thomson, Thurstone and Burt, who were investigating the mental ability of children as suggested by the correlation and covariance matrices from cognitive tests variables. This eventually led to the development of factor analysis. Principal component analysis, traditionally not considered a latent variable model, was first thought of by Pearson (1901) and developed as a multivariate technique by Hotelling (1933). Latent structure analysis, corresponding to models where the latent variables are categorical, originated with Lazarsfeld and Henry (1968) as a tool for sociological analysis and had a separate development from factor analysis until recently. Bartholomew (1987) gives more historical details and references.

The statistics literature classifies latent variable models according to the metric (continuous) or categorical (discrete) character of the variables involved, as table 2.1 shows. Also, in a broad sense many probabilistic models commonly used in machine learning can be considered as latent variable models inasmuch as they include probability distributions for variables which are not observed: mixture models (where the variable which indexes the components is a latent variable), hidden Markov models (the state sequence is unobserved),

This chapter is an extended version of references Carreira-Perpiñán (1996, 1997).

<table>
<tr><td></td><td></td><td colspan="2">Observed variables</td></tr>
<tr><td></td><td></td><td>Metrical</td><td>Categorical</td></tr>
<tr><td rowspan="3">Latent variables</td><td>Metrical</td><td>Factor analysis</td><td>Latent trait analysis</td></tr>
<tr><td rowspan="2">Categorical</td><td>Latent profile analysis</td><td>Latent class analysis</td></tr>
<tr><td colspan="2">Analysis of mixtures</td></tr>
</table>

Table 2.1: Classification of latent variable models (adapted from Bartholomew, 1987).

Helmholtz machines (Dayan et al., 1995) (the activations of each unit), elastic nets (Durbin et al., 1989) (the tour knots), etc. In this chapter we will concentrate exclusively on latent variable models where both the latent and the observed variables are continuous (although much of the general treatment applies to discrete variables too), going much further than the linear-normal model of factor analysis. In some points discrete variables will appear as a result of a Monte Carlo discretisation. The latent class model where the observed variables are binary and there is a single discrete latent variable corresponds to the mixture of multivariate Bernoulli distributions, which is discussed in chapter 3 and used in chapter 5.

In this work, we follow (with minor variations) the theoretical framework currently accepted in statistics for latent variable models, for which two good references are Everitt (1984) and Bartholomew (1987). However, these works do not include any of the more recent latent variable models such as GTM, ICA or IFA, and they do not discuss issues of importance in machine learning, such as the continuity of the dimensionality reduction mapping or the mixtures of continuous latent variable models. The work of MacKay (1995a) on density networks, which is the name he uses for nonlinear latent variable models, was pioneering in the introduction of the latent variable model framework to the machine learning community (in spite of an odd choice of journal).

The traditional treatment of latent variable models in the statistics literature is (Krzanowski, 1988):

1. formulate model (independence and distributional assumptions)
2. estimate parameters (by maximum likelihood)
3. test hypotheses about the parameters

which perhaps explains why most of the statistical research on latent variables has remained in the linear-normal model of factor analysis (with a few exceptions, e.g. the use of polynomial rather than linear functions; Etezadi-Amoli and McDonald, 1983; McDonald, 1985): rigorous analysis of nonlinear models is very difficult. Interesting developments concerning new models and learning algorithms have appeared in the literature of statistical pattern recognition in the 1990s, as the rest of this chapter will show.

Traditionally, most applications of latent variables have been in psychology and the social sciences, where hypothesised latent variables such as intelligence, verbal ability or social class are believed to exist; they have been less used in the natural sciences, where often most variables are measurable (Krzanowski, 1988, p. 502). This latter statement is not admissible under the generative point of view exposed in sections 2.2 and 2.3, whereby the processes of measurement and stochastic variation (noise) can make a physical system appear more high-dimensional than it really is. In fact, factor analysis has been applied to a number of "natural science" problems (e.g. in botany, biology, geology or engineering) as well as other kinds of latent variable models recently developed, such as the application of GTM to dimensionality reduction of electropalatographic data (Carreira-Perpiñán and Renals, 1998a) or of ICA to the removal of artifacts in electroencephalographic recordings (Makeig et al., 1997). Besides that, the most popular technique for dimensionality reduction, principal component analysis, has recently been recast in the form of a particular kind of factor analysis—thus as a latent variable model.

Whether the latent variables revealed can or cannot be interpreted (a delicate issue discussed in section 2.8.1), latent variable models are a powerful tool for data analysis when the intrinsic dimensionality of the problem is smaller than the apparent one and they are specially suited for dimensionality reduction and missing data reconstruction—as this thesis will try to demonstrate.

2.2 Physical interpretation of the latent variable model formalism

In this section we give a physical flavour to the generative view of latent variables. This generative view will be properly defined in section 2.3 and should be compared to the physical interpretation given below.

2.2.1 An example: eclipsing spectroscopic binary star

We will introduce the idea of latent variables through an example from astronomy, that of *eclipsing spectroscopic binary stars* (Roy, 1978). A binary system is a pair of stars that describe orbits about their common centre of mass, the two components being gravitationally bound together. An eclipsing spectroscopic binary is viewed as a single star because the components are so close that they cannot be resolved in a telescope, but its double nature can be revealed in several ways:

- If the orbit plane contains or is close to the line of sight, the components will totally or partially eclipse each other, which results in regular diminutions in the binary star's brightness.

- The Doppler effect[1] of the orbital velocities of the components produces shifts to red and blue on the spectral lines of the binary star.

A well-known example of this kind of star system is Algol (β Per), which is actually a ternary system.

We will consider a simplified model of eclipsing spectroscopic binary star, in which one star (the primary star) is much more massive than the other, thus remaining practically stationary in the centre-of-mass system. The secondary star follows an elliptical orbit around the primary one, which is located at one of the foci. Figures 2.1 and 2.2 show the trajectory of the secondary star and the basic form of the variation with the polar angle θ (true anomaly) of:

- The radial velocity v_r of the secondary star (i.e., the projection of the velocity vector along the line of sight from the Earth), to which the spectrum shift is proportional. We are assuming a stationary observer in Earth, so that the measured radial velocity must be corrected for the Earth's orbital motion about the Sun.
- The combined brightness B.

We would like to provide a scenario which, though idealised, is very close to an experimental situation that an astronomer could find. Thus, we are ignoring complicated effects, such as the facts that: the shape of both stars can be far from spherical, as they can fill their lobes (bounded by the Roche limit); the stars do not have uniform brightness across their discs, but decreasing towards the limb (limb darkening); the relativistic advance of the periastron with time; the perturbation due to third bodies; etc. Thus, the true theoretical curve for the brightness will not be piecewise linear and the actual curve will vary from one binary system to another, but it will always conserve the form indicated in fig. 2.2: a periodic curve with two falls corresponding to the eclipses. Similarly, the spectral shift will be a periodic oscillating function.

Now, an astronomer could collect a number of paired measurements (B, v_r) and construct with them brightness and radial velocity (or spectrum) curves such as those of fig. 2.2 (although perhaps replacing the true anomaly θ by a more natural variable in this case, such as the time t at which the measurement was collected). Detailed analysis of the brightness and spectrum curves under the laws of Kepler can provide knowledge of the eccentricity and inclination of the orbit, the major semiaxis, the radii, masses and luminosities of the stars, etc.

The knowledge of the astronomer that all measured variables (B and v_r) depend on a single variable, the true anomaly θ, is implicit. That is, given θ, the configuration of the binary system is completely defined and so are the values of all the variables we could measure: B, v_r, even r, etc.

Now let us suppose that we have such a collection of measurements but that we do not know anything about the underlying model that generated them, i.e., we do not know that the observed brightness and spectral shifts are (indirectly) governed by Kepler's law. Let us consider the following question: could it be possible that, although we are measuring two variables each time we observe the system, just one variable (not necessarily B or v_r) would be enough to determine the configuration of the system completely?[2] More concisely: could the number of degrees of freedom of the binary system be smaller than the number of variables that we measure from it?

As we discuss in section 4.4, this is a very difficult question to answer if the measurements are corrupted by noise or if each measurement consists of more than about 3 variables. But if we plotted the collection of pairs (B, v_r) in a plane coordinate system, with each variable being associated with one axis, we would observe the following fact (fig. 2.3 left): the points do not fall all over the plane, spanning an extensional (two-dimensional) area, but fall on a curve (dotted in the figure). If we accept that our instruments are imperfect, so that each point is slightly off the position where it should be, we would observe a situation like that of fig. 2.3 right: the points now occupy an extensional area (several oval patches), but it is apparent that the region over which the measurements fall is still a curve. This gives away the fact that the system has only one degree of freedom, however many different, apparently unrelated variables we want to measure from it. In this example, the intrinsic dimensionality of the binary system is one but we measure two variables.

Accepting that the intrinsic dimensionality of a system is smaller than the number of variables that we measure from it, the latent variable framework allows the construction of a generative model for the system.

[1] If a source emitting radiation has a velocity v relative to the observer, the received radiation that normally has a wavelength λ when the velocity relative to the observer is 0 will have a measured wavelength λ' with $\frac{\lambda'-\lambda}{\lambda} = \frac{v}{c}$, where c is the speed of light and the source is approaching for $v < 0$ and receding for $v > 0$. Thus, the Doppler shift is defined as $\frac{\Delta\lambda}{\lambda} = \frac{v}{c}$, being positive for red shift and negative for blue shift.

[2] Of course we could argue that two variables could not be enough to determine the system configuration completely, but we will suppose that this is not the case.

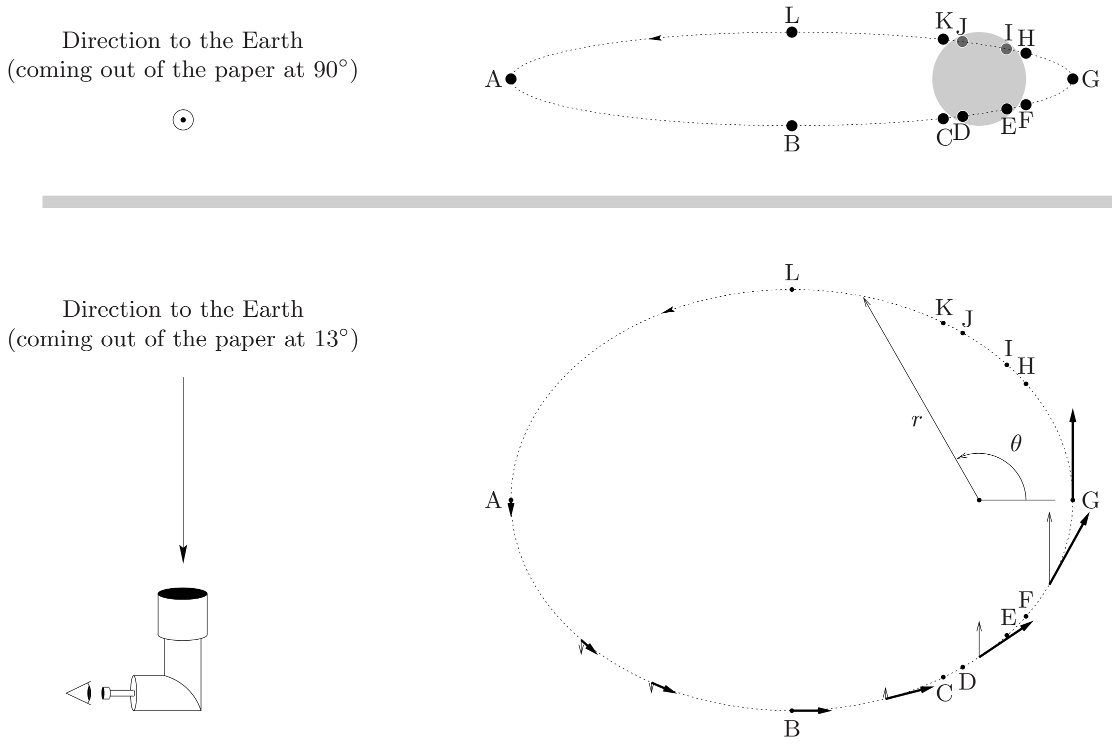


Figure 2.1: Schematic of the orbit for a binary system. The primary star is assumed stationary with the secondary following a Keplerian closed orbit around it: an ellipse of eccentricity $\epsilon = \frac{2}{3}$ with the primary star in a focus. The figure on the top shows the view of the binary system as seen from the Earth with a direction which has an inclination of 13° over the orbit plane and is orthogonal to the major axis of the orbit. The bottom figure shows the true orbit; the vectors of the velocity (thick arrows) and the radial velocity (in the direction of the Earth; solid arrows) are given for some points in the orbit.

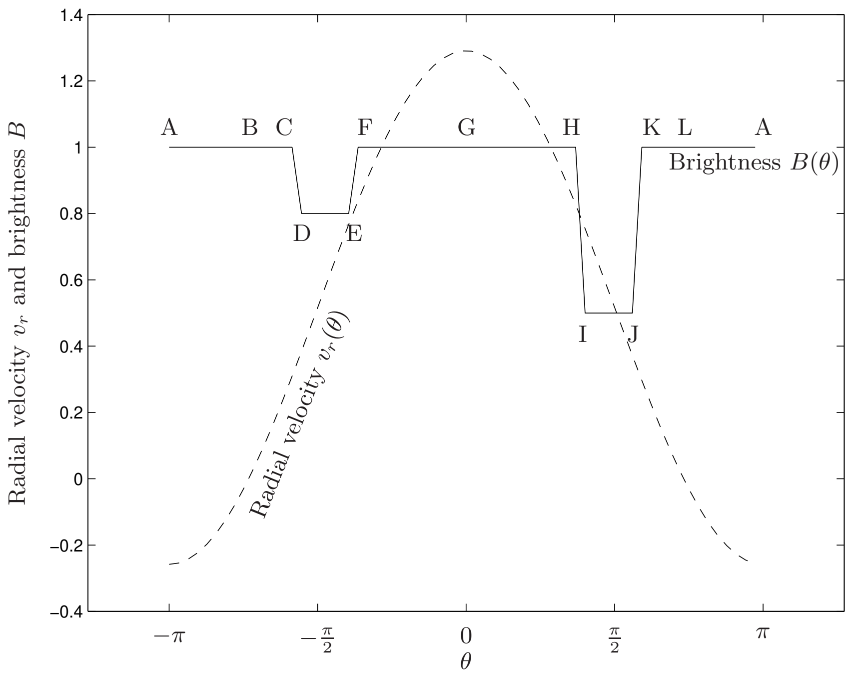


Figure 2.2: Variation of the radial velocity v_r and of the combined brightness B as a function of the true anomaly θ for the binary system of fig. 2.1. The units of the vertical axis of the graph have been normalised by a multiplying factor, so that only the shape of the curves is important. The brightness curve assumes that the secondary star has a very small radius compared to that of the primary one.

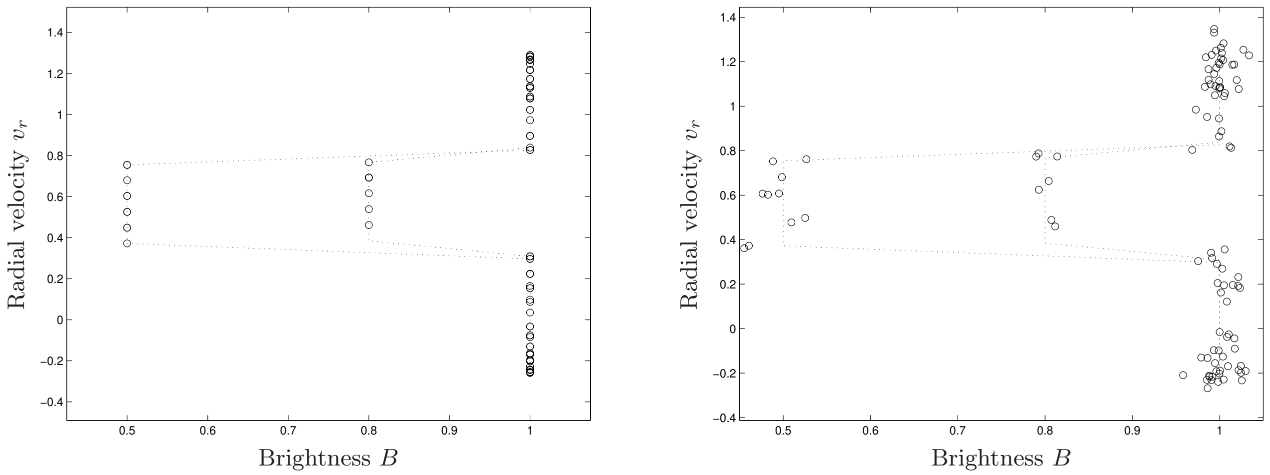


Figure 2.3: Plots of a number of measurements (B, v_r) in a coordinate system where the brightness goes in abscissas and the radial velocity in ordinates. The left graph corresponds to an ideal, noiseless measuring device: the measurements fall exactly on a curve. Due to the discontinuity of the brightness B, the curve in (B, v_r) space consists of several disconnected segments. The right graph corresponds to a real, noisy measuring device: the measurements still fall approximately on a curve, but now span a truly two-dimensional area. Again, due to the discontinuity of the brightness B, we observe several patches.

This generative model can, under certain circumstances, mimic the behaviour of the system inasmuch as it is a black box that generates data.

2.2.2 Physical interpretation of latent variables: measurement and noise

We can give a physical interpretation to the latent variable model formalism, to be defined more rigorously in section 2.3. Consider a (physical) system governed by L independent variables (or degrees of freedom) $x_1, \dots, x_L$, which we can represent as an L-dimensional vector $\mathbf{x} = (x_1, \dots, x_L)^T$ taking values in a certain subset $\mathcal{S}$ of $\mathbb{R}^L$, called *latent* or *state space* (we will consider only continuous variables). That is, every state of the system is given by a particular vector $\mathbf{x}$ in state space $\mathcal{S}$. The variables $x_1, \dots, x_L$ can be called *generalised coordinates* in classical mechanics or *quantum numbers* in quantum mechanics. In our example of the eclipsing spectroscopic binary star, the latent or state space is given by the variable θ varying continuously in the $[-\pi, \pi]$ interval[3].

The intrinsic dimensionality L of the system is unknown to the observer, who designs an experimental setup that allows to obtain one observation of the physical system by measuring a fixed number D of variables on it. Usually the number of observed variables D will be bigger (perhaps much bigger) than the true number of degrees of freedom L, because the observer needs to capture as much information from the system as possible. Any measured variable must be a function of the latent variables, so that we can represent the operation of obtaining one observation from the system by a *measurement* mapping $\mathbf{f} : \mathcal{S} \subset \mathbb{R}^L \longrightarrow \mathcal{M} \subset \mathbb{R}^D$. We will assume that the measurement mapping $\mathbf{f}$ is nonsingular[4], so that the observed variables will span an L-dimensional manifold $\mathcal{M} = \mathbf{f}(\mathcal{S})$ in $\mathbb{R}^D$; this manifold will be nonlinear in general. We refer to $\mathbb{R}^D$ as *data* or *observed space*. In our example of the eclipsing spectroscopic binary star, the observed variables would be the brightness B and the radial velocity v_r (or the spectral shift $\Delta\lambda$), which span a nonlinear one-dimensional manifold in $\mathbb{R}^2$, i.e., a plane curve. That is, if we plotted each observation in a two-dimensional coordinate system, with one axis being the brightness B and the other one the radial velocity v_r, all the points would fall on a curve instead of filling the plane or a two-dimensional region of it. This would make apparent the true dimensionality of the system (or, to be more strict, a lower bound of the true dimensionality).

Another set of observed variables, not measurable in practice for an eclipsing spectroscopic binary star, could be the x and y coordinates of the position vector of the secondary star drawn from the primary star (or

[3]One could argue that the state of the system is specified by the position variable θ and the velocity $(\dot{r}, \dot{\theta})$, but for our example both $\dot{r}$ and $\dot{\theta}$ are a function of θ.

[4]We say that a mapping $\mathbf{f}$ is *nonsingular* if the dimension of the image of $\mathbf{f}$ is equal to the dimension of its domain: $\dim \mathcal{S} = \dim \mathbf{f}(\mathcal{S})$. If $\mathbf{f}$ is linear, this means that the matrix associated with $\mathbf{f}$ is full-rank, or equivalently, that the image variables are linearly independent. If $\mathbf{f}$ is nonlinear, its Jacobian must be full-rank. See section A.7 for a discussion of L-manifolds in $\mathbb{R}^D$.

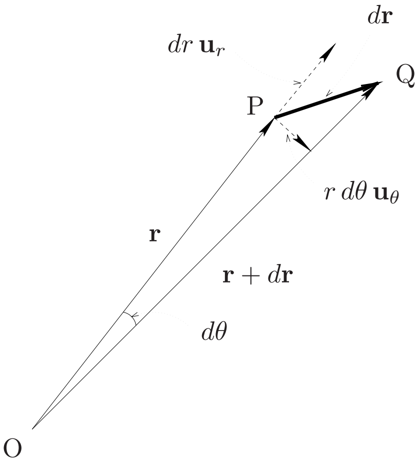


Figure 2.4: Differential displacement of a mobile point in plane polar coordinates (r, θ).

the polar coordinates (r, θ)!), which span precisely the elliptical orbit of figure 2.1. Yet another set of observed variables could be obtained by taking a picture of the binary system (again not possible practically for our example) and digitising it at $N \times N$ pixels; thus, $D = N^2$ could be made as large as desired, but the intrinsic dimension of the data, equal to the dimension of the generating process, would always be 1.

The mapping $\mathbf{f}$ describes an ideal **measurement process**, in which each point in latent space is exactly mapped onto a point in the L-dimensional manifold $\mathcal{M}$ in observed space. Thus, it would be possible to invert $\mathbf{f}$ and, for any point in $\mathcal{M}$, recover its corresponding point in latent space (again assuming that $L < D$ and that $\mathbf{f}$ is invertible): $\mathbf{f}^{-1} : \mathcal{M} \in \mathbb{R}^D \longrightarrow \mathcal{S} \in \mathbb{R}^L$. However, a real measurement process will introduce an error so that, for a latent point $\mathbf{x}$, the observer will measure a point $\mathbf{f}(\mathbf{x}) + \mathbf{e}$ in data space, where $\mathbf{e} \in \mathbb{R}^D$ is the error. The nature of this error is stochastic, i.e., repeated measurements of the system being in the same state would give different points in data space, even though the mapped point would be the same. The measurement error will hide to some extent the true, low-dimensional nature of the data, transforming the L-dimensional manifold $\mathcal{M}$ into a D-dimensional region in data space. The higher the error level, the more distortion $\mathcal{M}$ will suffer. It is customary to refer to this error as **noise**. We can assume that it follows a certain probability law, prescribed by a density function $p(\mathbf{t}|\mathbf{x})$, which gives the probability that latent point $\mathbf{x}$ will be observed as data point $\mathbf{t}$; we call this the *noise model*. The distribution of the noise will usually be a relatively generic, symmetric, zero-mean distribution, such as a normal $\mathcal{N}(\mathbf{f}(\mathbf{x}), \mathbf{\Sigma})$, but any other distribution is acceptable if there is a reason for it (e.g. if one suspects that the error is systematic, a skewed distribution will be necessary).

2.2.3 Probabilities and trajectories

Here we analyse the probability distribution associated with planar movement[5]. This is an example intended to show how complex probability distributions of latent variables can arise even in a simple situation.

Assume we have a curve $\mathcal{C}$ parameterised by a certain variable θ; in this section, we will consider plane polar coordinates (r, θ) with $r(\theta) \geq 0$ and $-\pi \leq \theta \leq \pi$. We assume that any point of the curve $\mathcal{C}$ has a unique corresponding value of the variable θ, i.e., that $r(\theta)$ is invertible. We assume that a moving object is following the curve $\mathcal{C}$ with a certain instantaneous velocity of modulus $v(\theta) \geq 0$ at each point $(r(\theta), \theta)$. Given $r(\theta)$ and $v(\theta)$, we want to find a function $p(\theta)$ that gives at each point the probability density of finding the moving object at that point, i.e., the probability density of seeing the point under a given polar angle θ. That is, if we took a number of pictures of the object, we would find it more often in those parts of the curve where the velocity is small, or in those values of the variable θ for which the trajectory $r(\theta)$ varies quickly. Then, $p(\theta)\,d\theta$ is, by definition of probability density function, the probability that the object is in the interval $(\theta, \theta + d\theta)$. This probability will be proportional to the time dt that the object takes to move from $(r(\theta), \theta)$ to $(r(\theta + d\theta), \theta + d\theta)$: $p(\theta)\,d\theta \propto dt$. Assuming that the velocity is constant during the infinitesimal time interval dt, the arc length corresponding to the displacement $d\mathbf{r}$ will be $ds = v\,dt$, where $ds = |d\mathbf{r}|$ and $dr = d\,|\mathbf{r}|$ (see fig. 2.4). From the figure, we have $ds = \sqrt{(dr)^2 + (r\,d\theta)^2}$, where $dr = d\,|\mathbf{r}|$ is the variation in the radial

[5]Naturally, movement is not the only reason why probability distributions can appear in physics.

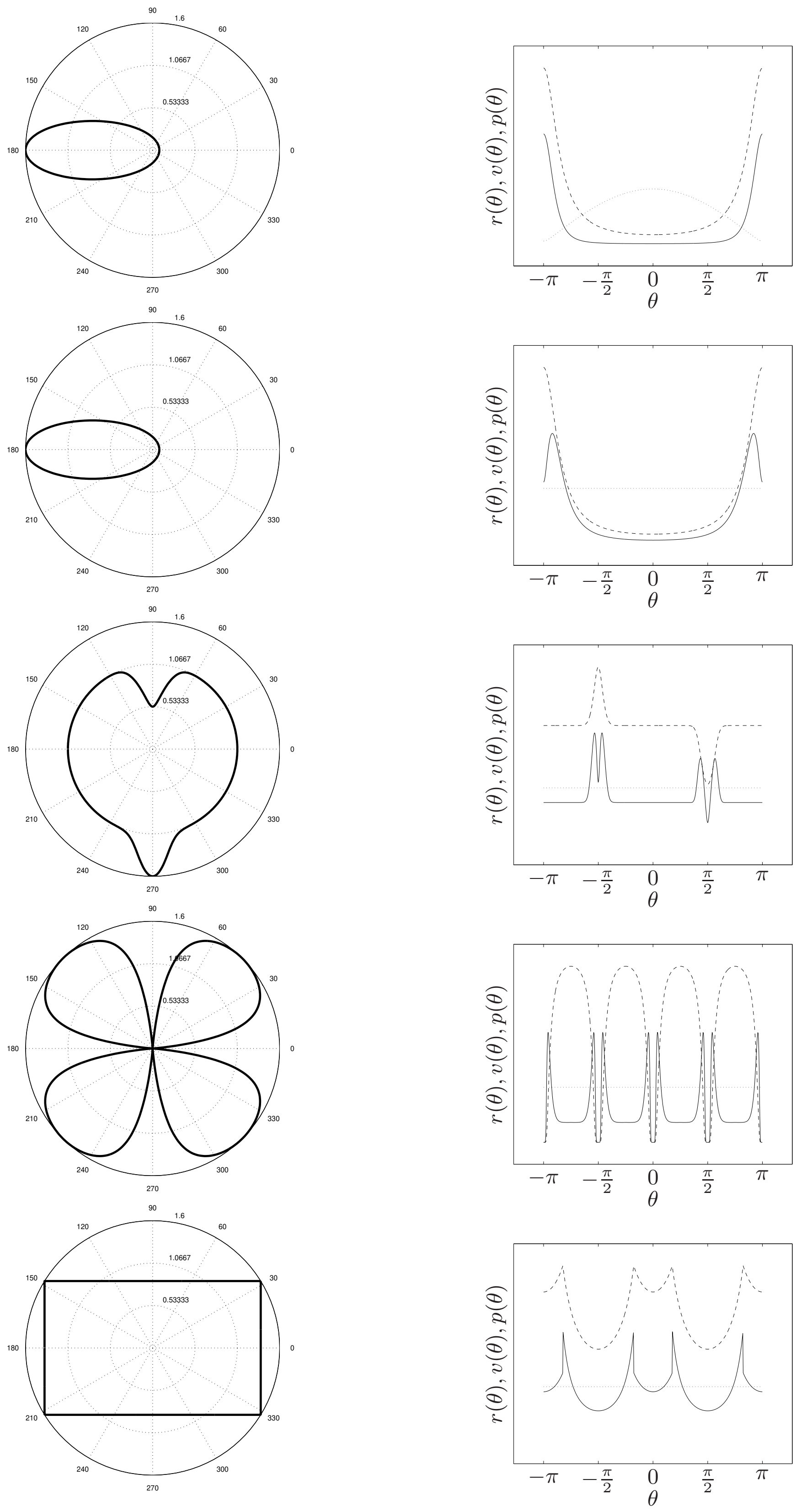


Figure 2.5: Examples of trajectories and the distributions generated by them. Each row represents a choice of the moduli of the radius vector $r(\theta)$ and the velocity $v(\theta)$ as functions of the polar angle θ, and each choice produces a certain probability distribution $p(\theta)$ of finding a mobile point at a certain polar angle θ. For each row, the figure on the left shows the trajectory $(r(\theta), \theta)$ in polar coordinates for $-\pi \leq \theta \leq \pi$ (thick line) and the graph on the right the form of the functions $r(\theta)$ (dashed line), $v(\theta)$ (dotted line) and $p(\theta)$ (solid line). All rows assume constant velocity modulus v except the first one, which assumes the Keplerian velocity (2.2).

direction and $r\,d\theta$ the variation in the tangential one. Then:

$$p(\theta) \propto \frac{dt}{d\theta} = \frac{1}{v}\frac{ds}{d\theta} = \frac{1}{v}\sqrt{\left(\frac{dr}{d\theta}\right)^2 + r^2} = \frac{1}{v}\sqrt{\dot{r}^2 + r^2}. \tag{2.1}$$

This function must be normalised dividing by $\int \frac{1}{v}\sqrt{\dot{r}^2 + r^2}\,d\theta$, the integral extended over the domain of θ. We can see from eq. (2.1) that the probability of finding the object in an interval $(\theta, \theta + d\theta)$ will be higher when:

- its velocity $v(\theta)$ is small there, or
- the distance to the origin $r(\theta)$ is high, or
- the trajectory varies quickly in the interval $(\theta, \theta + d\theta)$, i.e., the radial velocity $\dot{r}$ is large;

all in accordance with intuition. The trajectories for which the probability density is constant, when the point moves uniformly in θ-space, are given by the solutions $r(\theta)$, $v(\theta)$ of the differential equation $k^2v^2 = r^2 + \dot{r}^2$ for $k \in \mathbb{R}^+$. For example, for constant velocity this equation has two different solutions (where $R = kv$ is a positive constant):

- $r = R$, i.e., a circle centred at the origin;
- $r^2 + \dot{r}^2 = R^2 \Rightarrow r = R\,|\sin(\theta - \theta_0)|$, i.e., a circle passing through the origin.

For the familiar case of closed-orbit Keplerian movement of a two-body problem, the moduli of the radius vector and the velocity vary as a function of the polar angle θ as follows:

$$r(\theta) = \frac{a(1-\epsilon^2)}{1 + \epsilon\cos\theta} \qquad v(\theta) = \sqrt{\mu\left(\frac{2}{r} - \frac{1}{a}\right)}. \tag{2.2}$$

That is, the trajectory is an ellipse of eccentricity $0 \leq \epsilon < 1$ and major semiaxis a with a focus in the origin and $\mu = G(M + m)$, where G is the gravitational constant and M and m the masses of both bodies (Roy, 1978). According to the law of areas (Kepler's second law), the area swept by the radius vector per unit time is constant, i.e., $\left|\frac{\mathbf{r}\times d\mathbf{r}}{dt}\right|$ = constant (from fig. 2.4, $\left|\frac{1}{2}\mathbf{r} \times d\mathbf{r}\right|$ is the area of the triangle OPQ). Expanding $\mathbf{r} = r\mathbf{u}_r$ and $d\mathbf{r} = dr\,\mathbf{u}_r + r\,d\theta\,\mathbf{u}_\theta$ in the radial and tangential directions we obtain $r^2\frac{d\theta}{dt}$ = constant. From here and from the fact that $p(\theta) \propto \frac{dt}{d\theta}$ we obtain $p(\theta) \propto r^2$. The same result can be obtained by substituting the expressions for v and r from eq. (2.2) into eq. (2.1). Taking into account the following integral (Gradshteyn and Ryzhik, 1994):

$$\int_{-\pi}^{\pi} \frac{d\theta}{(1 + \epsilon\cos\theta)^2} = \frac{2\pi}{(1-\epsilon^2)^{3/2}},$$

the exact expression for $p(\theta)$ turns out to be:

$$p(\theta) = \frac{(1-\epsilon^2)^{3/2}}{2\pi}\frac{1}{(1 + \epsilon\cos\theta)^2} = \frac{r^2}{2\pi a^2\sqrt{1-\epsilon^2}}.$$

Figure 2.5 shows the trajectory and the form of the variation of r, v and p with θ for the example of Keplerian closed movement and for other trajectories for constant velocity. Observe how many different forms the distribution $p(\theta)$ can take, all physically possible, even for this simple example of planar movement. If we consider θ as a latent variable and the point coordinates (x, y) (for example) as observed variables, then $p(\theta)$ will be the prior distribution in latent space, as defined in section 2.3. Therefore, the distribution in latent space can be very complex, contrasting with the simple prior distributions of the latent variable models of section 2.6 (all of which are either normal of uniform, except for ICA). See sections 2.3.2 and 2.8 for a further discussion of this issue.

2.3 Generative modelling using continuous latent variables

In latent variable modelling the assumption is that the observed high-dimensional data is generated from an underlying low-dimensional process. The high dimensionality arises for several reasons, including stochastic variation and the measurement process. The objective is to learn the low dimensional generating process (defined by a small number of latent or hidden variables) along with a noise model, rather than directly

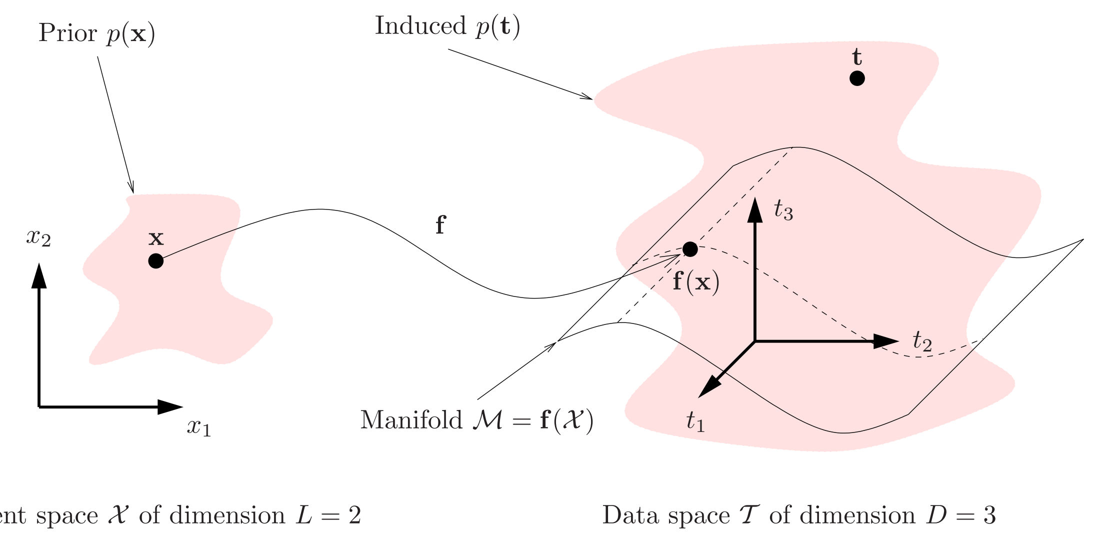


Figure 2.6: Schematic of a continuous latent variable model with a 3-dimensional data space and a 2-dimensional latent space.

learning a dimensionality reducing mapping. We will consider that the latent variables are mapped by a fixed transformation into a higher-dimension observed space (measurement procedure) and noise is added there (stochastic variation). In contrast with this *generative*, bottom-up point of view, statisticians often consider latent variable models from an *explanatory*, top-down point of view (Bartholomew, 1987): given the empirical correlation between the observed variables, the mission of the latent variables is to explain those correlations via the axiom of local independence (explained in section 2.3.1); i.e., given an observed distribution, find a combination of latent distribution and noise model that approximates it well.

We will consider that both the observed and the latent variables are continuous. Call $\mathcal{T} \subseteq \mathbb{R}^D$ the D-dimensional **data** or **observed space**[6]. Consider an unknown distribution $p(\mathbf{t})$ in data space, for $\mathbf{t} \in \mathcal{T}$, of which we only see a sample $\{\mathbf{t}_n\}_{n=1}^N \subset \mathcal{T}$. In latent variable modelling we assume that the distribution in data space $\mathcal{T}$ is actually due to a small number $L < D$ of latent variables acting in combination. We refer to this L-dimensional space as the **latent space** $\mathcal{X} \subseteq \mathbb{R}^L$.

Thus, a point $\mathbf{x}$ in latent space $\mathcal{X}$ is generated according to a **prior distribution** $p(\mathbf{x})$ and it is mapped onto data space $\mathcal{T}$ by a smooth, nonsingular mapping $\mathbf{f} : \mathcal{X} \to \mathcal{T}$. Because $\mathcal{M} = \mathbf{f}(\mathcal{X})$ is an L-dimensional manifold in $\mathcal{T}$, in order to extend it to the whole D-dimensional data space we define a distribution $p(\mathbf{t}|\mathbf{x}) = p(\mathbf{t}|\mathbf{f}(\mathbf{x}))$ on $\mathcal{T}$, called the **noise** or **error model**. Figure 2.6 illustrates the idea of latent variable models.

The joint probability density function in the product space $\mathcal{T} \times \mathcal{X}$ is $p(\mathbf{t}, \mathbf{x})$ and integrating over the latent space gives the marginal distribution in data space:

$$p(\mathbf{t}) = \int_{\mathcal{X}} p(\mathbf{t}, \mathbf{x})\, d\mathbf{x} = \int_{\mathcal{X}} p(\mathbf{t}|\mathbf{x}) p(\mathbf{x})\, d\mathbf{x}. \tag{2.3}$$

This is called the **fundamental equation of latent variable models** by Bartholomew (1984). Thus, a model is essentially a specification of $p(\mathbf{x})$ and $p(\mathbf{t}|\mathbf{x})$—the specification of the mapping $\mathbf{f}$ can be absorbed into that of $p(\mathbf{t}|\mathbf{x})$. The only empirical evidence available concerns $p(\mathbf{t})$ through the sample $\{\mathbf{t}_n\}_{n=1}^N$ and so the only constraint on $p(\mathbf{x})$ and $p(\mathbf{t}|\mathbf{x})$, apart from the need to be nonnegative and integrate to 1, is given by eq. (2.3). In general, there are many combinations of $p(\mathbf{x})$ and $p(\mathbf{t}|\mathbf{x})$ that can satisfy (2.3) for a given $p(\mathbf{t})$.

Eq. (2.3) can also be seen as a continuous mixture model (Everitt and Hand, 1981), with the latent variable $\mathbf{x}$ "indexing" continuously the mixture component. In fact, any density function can be considered as a mixture density where extra variables have been integrated over.

Eq. (2.3) (or, for that matter, any marginalisation) can also be seen as a particular case of a *Fredholm integral equation of the first kind*. A Fredholm integral equation of the first kind (in one dimension) has the form:

$$f(t) = \int_{-\infty}^{\infty} K(t, x) g(x)\, dx$$

[6]In the statistical literature the variables in the data space are usually called *manifest variables*.

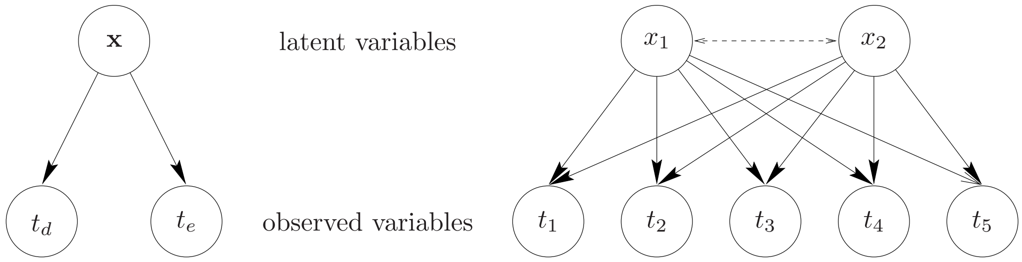


Figure 2.7: Graphical model representation of latent variable models. *Left*: the axiom of local independence. *Right*: a latent variable model with $D = 5$ observed variables and $L = 2$ latent variables. The dotted line indicates that the latent variables may or may not be independent in a particular model.

where the function $K(t, x)$ is called the kernel and both K and f are known while g is unknown. If all functions are probability densities and specifically $K(t, x) \equiv p(t|x)$ we obtain eq. (2.3). Note the similarity of the Fredholm integral equation with a matrix equation $\mathbf{f} = \mathbf{Kg}$, whose solution is $\mathbf{g} = \mathbf{K}^{-1}\mathbf{f}$. In fact, most inverse problems (to which chapter 6 is dedicated) are Fredholm integral equations of the first kind. Press et al. (1992, chapter 18) contains a brief discussion of numerical methods for Fredholm and Volterra integral equations and further references.

2.3.1 Noise model: the axiom of local independence

If the latent variables are to be efficient in representing faithfully the observed variables, we should expect that, given a value for the latent variables, the values of any group of observed variables are independent of the values of any other group of observed variables. Otherwise, the chosen latent variables would not completely explain the correlations between the observed variables and further latent variables would be necessary. Thus, for all $d, e \in \{1, \dots, D\}$,

$$p(t_d|t_e, \mathbf{x}) = p(t_d|\mathbf{x}) \Rightarrow p(t_d, t_e|\mathbf{x}) = p(t_d|t_e, \mathbf{x})p(t_e|\mathbf{x}) = p(t_d|\mathbf{x})p(t_e|\mathbf{x})$$

and we obtain that the distribution of the observed variables conditioned on the latent variables, or the noise model, is factorial (the subindex in p_d emphasises that the component distributions $p_d(t_d|\mathbf{x})$ need not be the same):

$$p(\mathbf{t}|\mathbf{x}) \stackrel{\text{def}}{=} \prod_{d=1}^{D} p_d(t_d|\mathbf{x}). \tag{2.4}$$

That is, for some $L \leq D$, *the observed variables are conditionally independent given the latent variables*. This is usually called the **axiom of local (or conditional) independence** (Bartholomew, 1984; Everitt, 1984). It will prove very convenient in section 7.12.3 because choosing factorial noise models simplifies the calculations of conditional distributions considerably.

It should be noted that, rather than an assumption, the axiom of local independence is a definition of what it means to have fully explained the joint distribution of the observed variables in terms of the latent ones. The aim is to find the smallest number of latent variables $L \leq D$ for which it holds (it does, trivially, for $L = D$ by taking $x_d \equiv t_d$, provided that (2.3) is satisfied). However, in practice one will need to try several values of L and select the best one.

Thus said, there have been some suggestions of models that violate the axiom of linear independence. One example has been proposed for the output distribution of a hidden Markov model: Gopinath et al. (1998) model the output distribution of state s as a factor analyser with nondiagonal covariance matrix $\mathbf{\Sigma}_T = \mathbf{U}\mathbf{\Psi}_s\mathbf{U}^T$ where $\mathbf{\Psi}_s$ is diagonal and $\mathbf{U}$ is an orthogonal matrix shared among states. It is not clear why this constrained covariance model should have any advantage over other parameter-tying methods (Young, 1996) or over using a mixture of factor analysers as output distribution (Saul and Rahim, 2000b).

The graphical model (Jensen, 1996; Jordan, 1998; Pearl, 1988; Whittaker, 1990) representing the local independence is shown in fig. 2.7 (left). Fig. 2.7 (right) shows the graphical model for a latent variable model with $D = 5$ and $L = 2$.

Regarding the actual choice of the functional form of the noise model $p(\mathbf{t}|\mathbf{f}(\mathbf{x}))$, it seems reasonable to use a density function with the following properties:

Sphering $\mathbf{t}' \overset{\text{def}}{=} \boldsymbol{\Sigma}^{-1/2}(\mathbf{t} - \boldsymbol{\mu})$ Transformation $\mathbf{t}'' \overset{\text{def}}{=} \begin{pmatrix} \sigma^{-1} & 0 \\ 0 & 1 \end{pmatrix} (\mathbf{t} - \boldsymbol{\mu})$

$$\mathbf{t} \overset{\text{def}}{\sim} \tfrac{1}{2}\mathcal{N}\left(\begin{pmatrix} -1 \\ 0 \end{pmatrix}, \begin{pmatrix} \sigma^2 & 0 \\ 0 & 1 \end{pmatrix}\right) + \tfrac{1}{2}\mathcal{N}\left(\begin{pmatrix} 1 \\ 0 \end{pmatrix}, \begin{pmatrix} \sigma^2 & 0 \\ 0 & 1 \end{pmatrix}\right)$$

$$\boldsymbol{\mu} = \mathbf{0},\ \boldsymbol{\Sigma} = \begin{pmatrix} \sigma^2+1 & 0 \\ 0 & 1 \end{pmatrix}$$

$$\mathbf{t}' \sim \tfrac{1}{2}\mathcal{N}\left(\begin{pmatrix} -(\sigma^2+1)^{-1/2} \\ 0 \end{pmatrix}, \begin{pmatrix} \sigma^2(\sigma^2+1)^{-1} & 0 \\ 0 & 1 \end{pmatrix}\right) + \tfrac{1}{2}\mathcal{N}\left(\begin{pmatrix} (\sigma^2+1)^{-1/2} \\ 0 \end{pmatrix}, \begin{pmatrix} \sigma^2(\sigma^2+1)^{-1} & 0 \\ 0 & 1 \end{pmatrix}\right)$$

$$\boldsymbol{\mu}' = \mathbf{0},\ \boldsymbol{\Sigma}' = \mathbf{I}$$

$$\mathbf{t}'' \sim \tfrac{1}{2}\mathcal{N}\left(\begin{pmatrix} -\sigma^{-1} \\ 0 \end{pmatrix}, \mathbf{I}\right) + \tfrac{1}{2}\mathcal{N}\left(\begin{pmatrix} \sigma^{-1} \\ 0 \end{pmatrix}, \mathbf{I}\right)$$

$$\boldsymbol{\mu}'' = \mathbf{0},\ \boldsymbol{\Sigma}'' = \begin{pmatrix} \sigma^{-2}(\sigma^2+1) & 0 \\ 0 & 1 \end{pmatrix}$$

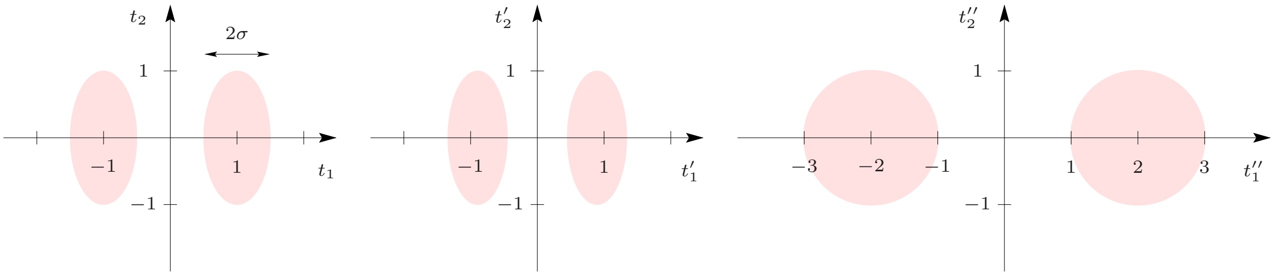


Figure 2.8: Sphering does not yield spherical local noise in general, as demonstrated here via a mixture of two normal distributions. *Left*: original distribution. *Centre*: sphered distribution. *Right*: linear transformation to local spherical noise (unknown, since the local noise is unknown in advance). Each normal distribution is represented as a unit standard deviation elliptical boundary in Mahalanobis distance. In each case, $\sigma = \frac{1}{2}$ and $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ are the mean and covariance matrix, respectively, of the mixture, computed as in section 8.7.

- It is centred at $\mathbf{f}(\mathbf{x})$, which would be the only possible point in the absence of noise:

$$\forall \mathbf{x} \in \mathcal{X}: \ \mathrm{E}\left\{\mathbf{t}|\mathbf{x}\right\} = \mathbf{f}(\mathbf{x}). \tag{2.5}$$

  This is a relaxed form of the self-consistency condition (4.2) of principal curves.

- It decays gradually as the distance to $\mathbf{f}(\mathbf{x})$ increases, according to some parameter related to the noise covariance. However, it need not be symmetric around $\mathbf{f}(\mathbf{x})$.

- It assigns nonzero density to every point in the observed space, for two reasons:
  - No region of the observed space should have null probability unless knowledge about the problem at hand dictates otherwise.
  - Noise models which have a distribution with finite support (i.e., assigning nonnull probability only to points inside a finite neighbourhood of the centre, $\mathbf{f}(\mathbf{x})$) make very difficult to compute the data distribution, $p(\mathbf{t})$. This is so because the integration in latent space (2.3) to compute $p(\mathbf{t})$ for a given observed point $\mathbf{t}$ is restricted to those latent points $\mathbf{x}$ for which the neighbourhood of $\mathbf{f}(\mathbf{x})$ contains $\mathbf{t}$ (so that $p(\mathbf{t}|\mathbf{x}) > 0$).

- It should have a diagonal covariance matrix to account for different scales in the different observed variables $t_1, \dots, t_D$. This latter aspect cannot be overcome in general by sphering the data and using an isotropic noise model, because if the data has clusters, the intercluster distances affect the covariances; fig. 2.8 illustrates this with a simple example. The same happens if the data manifold is nonlinear. For cases where the dispersion attains orders of magnitude, a logarithmic transformation may be helpful.

In all the specific latent variable models discussed in section 2.6 (except ICA, which assumes no noise in its standard formulation) a normal noise model is assumed (thus symmetric), whose covariance matrix is either diagonal (factor analysis, IFA) or spherical (PCA, GTM). This is primarily due to the mathematical tractability of the multivariate normal distribution with respect to linear combinations, marginalisation and conditioning and its closure with respect to those operations—which simplifies the computation of integral (2.3) and the derivation of an EM algorithm for parameter estimation. It also adds flexibility to the model, since $p(\mathbf{t})$ ends up being a Gaussian mixture, which is a universal density approximator[7] (although the model itself may not be; see below). Yet another justification comes from the central limit theorem: if the noise is due to

[7]However, for mixtures the shape of each component is not crucial for density approximation, as is well known from kernel density estimation (Titterington et al., 1985; Scott, 1992), as long as there is a high enough number of components. Most localised kernels (decreasing from the central point) give rise to universal density approximators.

the combined additive action of a number of uncontrolled variables of finite variance, then its distribution will be asymptotically normal.

However, one disadvantage of the normal distribution is that its tails decay very rapidly, which reduces robustness against outliers: the presence of a small percentage of outliers in the training set can lead to significantly poor parameter estimates (Huber, 1981). Besides, many natural distributions have been proven to have long tails and would be better modelled by, say, a Student-t distribution or by infinite-variance members of the Lévy family[8], such as the Cauchy (or Lorentzian) distribution (Casti, 1997; Shlesinger et al., 1995). Another potential disadvantage of the normal distribution is its symmetry around the mean, when the noise is skewed. Skewed multivariate distributions may be obtained as normal mixtures or as multivariate extensions of univariate skewed distributions[9], but in both cases the analytical treatment may become very complicated—even if the axiom of local independence is followed, in which case a product of univariate skewed distributions may be used.

It should be noted that the noise may depend on the point in latent space (or on the point in data space). For example, in the binary system of section 2.2.1, the noise in the apastron area (point A in fig. 2.1) may be smaller than in the periastron area and its surroundings (points C to K) due to the interference with the central star. If, say, a normal noise model $\mathcal{N}(\mathbf{f}(\mathbf{x}), \mathbf{\Psi})$ is assumed, then its covariance $\mathbf{\Psi}$ should be a function of $\mathbf{x}$ too: $\mathbf{\Psi} = \mathbf{\Psi}(\mathbf{x})$ (fig. 2.9). However, this would require implementing $\mathbf{\Psi}(\mathbf{x})$ via some function approximator (e.g. a multilayer perceptron) and the mathematical treatment becomes very complicated. None of the models described in section 2.6 implement a noise model dependent on $\mathbf{x}$, and therefore none of them would be able to represent a data distribution such as the one depicted in fig. 2.9. For example, GTM could probably capture the mapping $\mathbf{f}$ but would only find an average value for the noise covariance, thus missing the data density. In principle one could think that GTM could approximate any smooth density function, since the data density has the same form as a kernel estimator: eq. (2.8) or eq. (2.43) where the components are spherical Gaussian kernels of width, or smoothing parameter, σ. However, in the kernel estimator the kernel centres are free to move in data space while in GTM they are constrained by the mapping $\mathbf{f}$. Therefore, GTM is not a universal density approximator.

In latent variable models that use a constant noise model and a sampled latent space, like GTM, another disadvantage appears. The observed space distribution is a mixture in which all components have the same noise model, but are located at different places in observed space, as in eq. (2.8). Since the width of each component is the same, those areas of observed space that have low probability (few samples) will be assigned few, widely separated components compared to high-probability areas. As a result, the density estimate in those low-probability areas will not be smooth, presenting a characteristic ripple that gives rise to spurious modes, as discussed in section 7.9.1. This phenomenon is well-known in kernel density estimation with fixed-width kernels (Silverman, 1986, pp. 17–18) and is particularly noticeable in the tails of the distribution being approximated.

2.3.1.1 Latent variable models and principal curves

Hastie and Stuetzle (1989) define principal curves (reviewed in section 4.8) as smooth curves (or, in general, manifolds) that pass through the middle of a data set and satisfy the self-consistence condition (4.2), which we reproduce here for convenience:

$$\forall \mathbf{x} \in \mathcal{X} : \ \mathrm{E}\left\{\mathbf{t} | \mathbf{F}(\mathbf{t}) = \mathbf{x}\right\} = \mathbf{f}(\mathbf{x}) \tag{4.2}$$

where the projection or dimensionality reduction mapping $\mathbf{F} : \mathbf{t} \in \mathcal{T} \longrightarrow \mathcal{X}$ is defined as the point in $\mathcal{X}$ whose image by $\mathbf{f}$ is closest to $\mathbf{t}$ in the Euclidean distance in $\mathcal{T}$. Ignoring boundary effects, $\mathbf{F}^{-1}(\mathbf{x})$ (the points in $\mathcal{T}$ projecting on $\mathbf{x}$) will be a subset of the manifold orthogonal to $\mathcal{M} = \mathbf{f}(\mathcal{X})$ at $\mathbf{f}(\mathbf{x})$, as fig. 2.10 shows.

[8] The Lévy, or stable, family of probability distributions (Feller, 1971, sec. VI.1–3) is defined as $\mathcal{F}\{P_N(x)\} \stackrel{\text{def}}{=} \mathcal{F}\{e^{-N|x|^\beta}\}$, where $\mathcal{F}$ is the Fourier transform of the probability $P_N(x)$ for N-step addition of random variables and $\beta \in (0, 2)$. Closed-forms for the pdf are only known in a few cases. All its members have an infinite variance and are scale invariant. The case $\beta = 1$ gives a Cauchy distribution and the limit case $\beta = 2$ gives the Gaussian distribution. The addition of N Lévy-distributed random variables follows a Lévy distribution. Lévy's theorem states that the addition of a number of random variables follows asymptotically a Lévy distribution (the Gaussian distribution if all the random variables have finite variance).

[9] Few useful multivariate extensions of nonnormal distributions exist. One such extension may be the multivariate skew-normal distribution $\mathcal{SN}(\mathbf{\Sigma}, \boldsymbol{\alpha})$ discussed by Azzalini and Capitanio (1999), with density:

$$p(\mathbf{t}) \stackrel{\text{def}}{=} 2\phi_D(\mathbf{t}; \mathbf{\Sigma})\Phi(\boldsymbol{\alpha}^T\mathbf{t}) \qquad \mathbf{t} \in \mathbb{R}^D$$

where ϕ_D is the D-variate pdf of the normal distribution with zero mean and covariance $\mathbf{\Sigma}$, Φ is the univariate cdf of the standard normal distribution $\mathcal{N}(0, 1)$ and the parameter $\boldsymbol{\alpha} \in \mathbb{R}^D$ partly regulates the skewness, with $\boldsymbol{\alpha} = \mathbf{0}$ giving the symmetric $\mathcal{N}(\mathbf{0}, \mathbf{\Sigma})$. This extension has some interesting properties, such as the fact that its marginal distributions are skew-normal too.

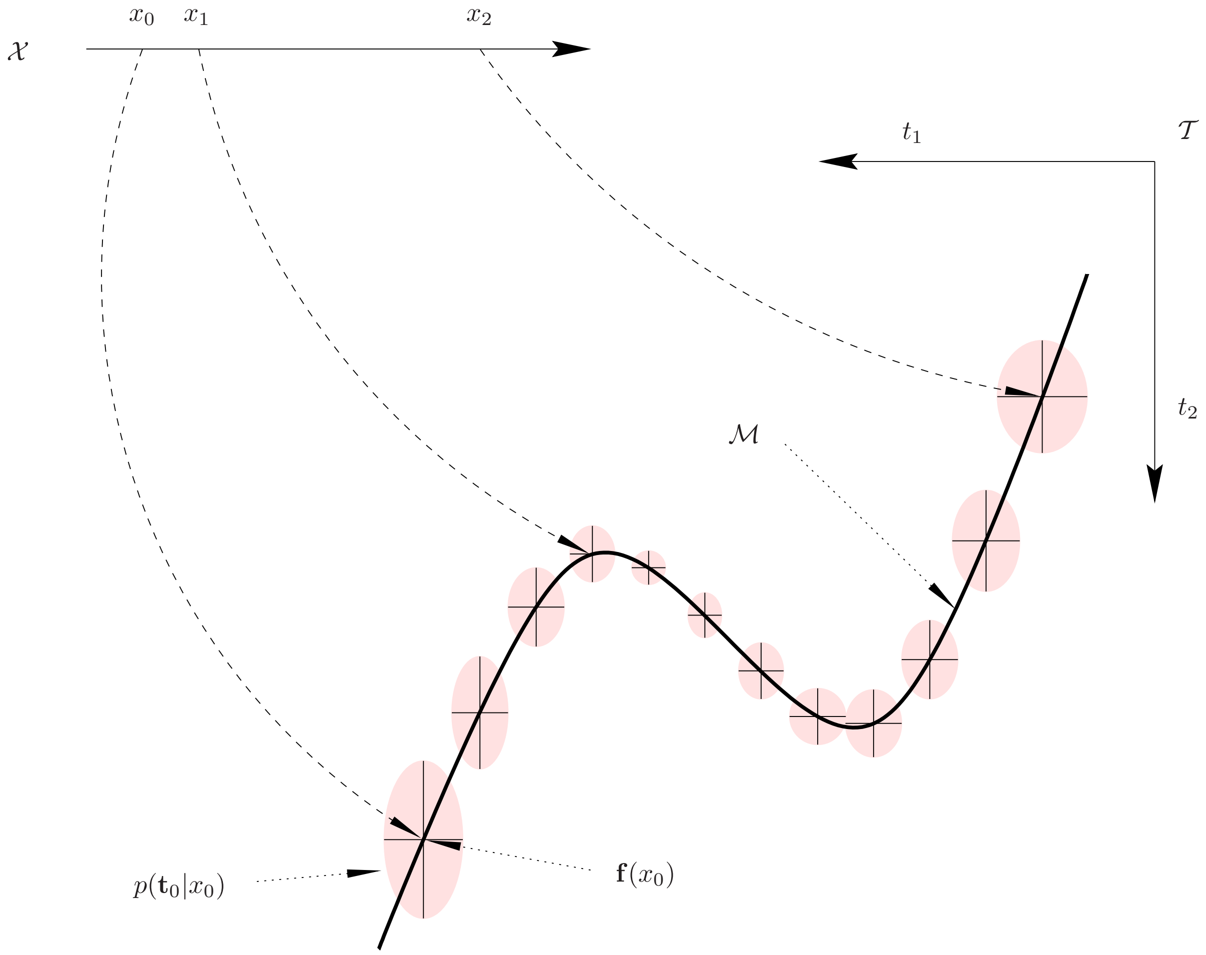


Figure 2.9: Variable noise in a latent variable model: the noise distribution $p(\mathbf{t}|\mathbf{x})$ depends on the latent point $\mathbf{x}$. The half bars on each ellipse measure one standard deviation of the corresponding noise distribution.

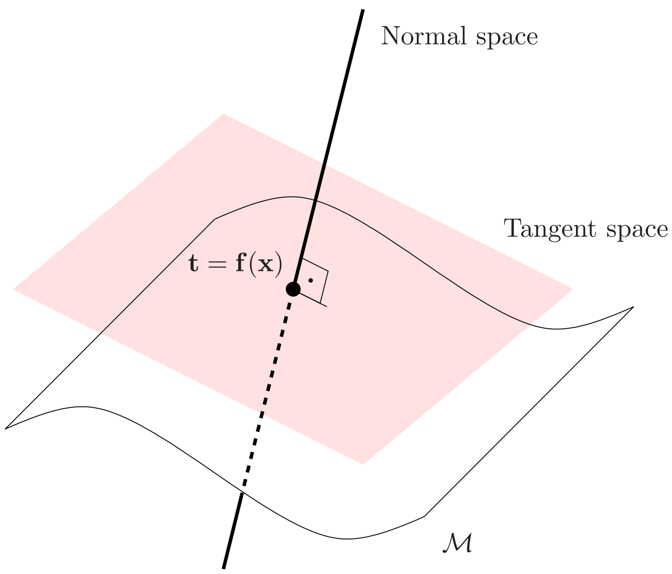


Figure 2.10: Self-consistency condition of principal curves. The graph shows a principal surface $\mathcal{M} = \mathbf{f}(\mathcal{X})$ (of dimension $L = 2$) and the normal space at $\mathbf{f}(\mathbf{x})$, in which $\mathbf{F}^{-1}(\mathbf{x})$, the set of data points projecting on a latent point $\mathbf{x}$, is contained. $\mathcal{M}$ is self-consistent if $\mathrm{E}\left\{\mathbf{t}|\mathbf{F}(\mathbf{t}) = \mathbf{x}\right\} = \mathbf{f}(\mathbf{x})$ for any $\mathbf{x} \in \mathcal{X}$.

We can see that the principal curve self-consistency condition (4.2) is more restrictive than condition (2.5) (verified by unbiased latent variable models) in that the expectation is restricted to the set $\mathbf{F}^{-1}(\mathbf{x})$ rather than to the whole data space $\mathcal{T}$. In other words, the (unbiased) latent variable model condition (2.5) means that $p(\mathbf{t}|\mathbf{x})$ is centred at $\mathbf{f}(\mathbf{x})$ for any $\mathbf{x} \in \mathcal{X}$, whereas the principal curves self-consistency condition (4.2) means that $p(\mathbf{t}|\mathbf{x})$ *restricted to the points in* $\mathcal{T}$ *projecting onto* $\mathbf{x}$ (which is a subset of the normal hyperplane to $\mathcal{M}$ at $\mathbf{f}(\mathbf{x})$) is centred at $\mathbf{f}(\mathbf{x})$ for any $\mathbf{x} \in \mathcal{X}$.

Clearly then, redefining self-consistency as condition (2.5) and considering both $\mathbf{t}$ and $\mathbf{x}$ as random variables turns the principal curve into a latent variable model. This is what Tibshirani (1992) did, seeking to eliminate the bias intrinsic to the standard definition of principal curves. He then approached the estimation problem as nonparametric estimation of a continuous mixture (section 2.5.2) and, by assuming a diagonal normal distribution for $p(\mathbf{t}|\mathbf{f}(\mathbf{x}))$, reduced it to EM estimation of a Gaussian mixture. As mentioned in section 2.5.2, this has the disadvantage of defining the principal curve $\mathbf{f}$ only through a finite collection of points in $\mathcal{X}$ rather than as a smooth function defined for all points in $\mathcal{X}$.

Under a generative view, such as that adopted for latent variable models (section 2.3), the self-consistency condition (4.2) is unnatural, since it means that for a point that has been generated in latent space and mapped onto a data space point $\mathbf{t}$, the error added to $\mathbf{t}$ must be "clever" enough to perturb the point $\mathbf{t}$ in a direction orthogonal to the manifold $\mathcal{M}$ at $\mathbf{t}$ and do so with mean zero! This only seems possible in the trivial case where the tangent manifold is constant in direction, i.e., the principal curve has curvature zero. This is the case of the principal component subspaces and normal distributions, in which case latent variable models and principal curves coincide.

2.3.2 Prior distribution in latent space: interpretability

Given any prior distribution $p_{\mathbf{x}}(\mathbf{x})$ of the L latent variables $\mathbf{x}$, it is always possible to find an invertible transformation $\mathbf{g}$ to an alternative set of L latent variables $\mathbf{y} = (y_1, \dots, y_L) = \mathbf{g}(\mathbf{x})$ having another desired distribution $p_{\mathbf{y}}(\mathbf{y})$:

$$\begin{tikzcd} \mathcal{X} \arrow[r, "\mathbf{g}"] \arrow[rr, bend right, "\mathbf{f}"'] & \mathcal{Y} \arrow[r, "\mathbf{f}'"] & \mathcal{T} \end{tikzcd}$$

The mapping from the new latent space onto the data space becomes $\mathbf{f}' = \mathbf{f} \circ \mathbf{g}^{-1}$, i.e., $\mathbf{t} = \mathbf{f}(\mathbf{x}) = \mathbf{f}(\mathbf{g}^{-1}(\mathbf{y})) = \mathbf{f}'(\mathbf{y})$ and the new prior distribution of the $\mathbf{y}$ variables becomes $p_{\mathbf{y}} = p_{\mathbf{x}} |\mathbf{J}_{\mathbf{g}}|^{-1}$, where $\mathbf{J}_{\mathbf{g}} \stackrel{\text{def}}{=} \left(\frac{\partial g_l}{\partial x_k}\right)$ is the Jacobian of the transformation $\mathbf{g}$. That is, given a space $\mathcal{X}$ with a distribution $p_{\mathbf{x}}$, to transform it into a space $\mathcal{Y}$ with a distribution $p_{\mathbf{y}}$ we apply[10] an invertible mapping $\mathbf{g} : \mathcal{X} \to \mathcal{Y}$ of Jacobian $|\mathbf{J}_{\mathbf{g}}| = \frac{p_{\mathbf{x}}}{p_{\mathbf{y}}}$. For example, if $\mathbf{x}$ are independently and normally distributed, transforming $y_l = e^{x_l}$ for $l = 1, \dots, L$ means that y_l follow a log-normal distribution.

Thus, fixing the functional form of the prior distribution in latent space is a convention rather than an assumption[11]. However, doing so requires being able to select $\mathbf{f}$ and $p_d(\mathbf{t}|\mathbf{x})$ from a broad class of functions so that eq. (2.3) still holds. GTM (section 2.6.5) is a good example: while it keeps the prior $p(\mathbf{x})$ simple (discrete uniform), its mapping $\mathbf{f}$ is a generalised linear model (which has universal approximation capabilities).

In particular, we can choose the latent variables to be independent[12] and identically distributed, $p_{\mathbf{x}}(\mathbf{x}) \stackrel{\text{def}}{=} \prod_{l=1}^{L} p(x_l)$. Unfortunately, while this is a particularly simple and symmetric choice for prior distribution in latent space, it does not necessarily simplify the calculation of the difficult L-dimensional integral (2.3), which is a major shortcoming of the latent variable modelling framework (see section 2.4).

We coincide with Bartholomew (1985) that the latent variables must be seen as constructs designed to simplify and summarise the observed variables, without having to look for an interpretation for them (which may exist in some cases anyway): all possible combinations of $p(\mathbf{x})$ and $p(\mathbf{t}|\mathbf{x})$ are equally valid as long as they satisfy eq. (2.3). Thus, we do not need to go into the issue of the interpretation of the latent variables, which has plagued the statistical literature for a very long time without reaching a general consensus.

Another issue is the topology of the true data manifold. For the one-dimensional example of figure 2.13, the data manifold is a closed curve (an ellipse). Thus, modelling it with a latent space that has the topology of an open curve (e.g. an interval of the Cartesian coordinate x) will lead to a discontinuity where both ends of

[10]Unfortunately, this is terribly complicated in practice, since solving the nonlinear system of partial differential equations $\left|\frac{\partial \mathbf{y}}{\partial \mathbf{x}}\right| = \frac{p_{\mathbf{x}}}{p_{\mathbf{y}}}$ to obtain $\mathbf{y} = \mathbf{g}(\mathbf{x})$ will be impossible except in very simple cases.

[11]The choice of a prior distribution in the latent space is completely different from, and much simpler than, the well-known problem of Bayesian analysis of choosing a noninformative prior distribution for the parameters of a model (mentioned in section 6.2.3.2)

[12]When $\mathbf{g}$ is linear and $y_1, \dots, y_L$ are independent, this is exactly the objective of independent component analysis (section 2.6.3).

the curve join: it is impossible to have a continuous mapping between spaces with different topological characteristics without having singularities, e.g. mapping a circle onto a line. A representation using a periodic latent variable is required[13] (e.g. the polar angle θ). Although some techniques exist for Gaussian mixture modelling of periodic variables (Bishop and Nabney, 1996), all the latent variable models considered in section 2.6 assume non-periodic latent variables.

A flexible and powerful representation of the prior distribution can be obtained with a mixture:

$$p(\mathbf{x}) \stackrel{\text{def}}{=} \sum_{m=1}^{M} p(m) p(\mathbf{x}|m)$$

which keeps the marginalisation in data space (2.3) analytically tractable (if the component marginals are):

$$p(\mathbf{t}) = \int_{\mathcal{X}} p(\mathbf{t}|\mathbf{x}) p(\mathbf{x})\, d\mathbf{x} = \sum_{m=1}^{M} p(m) \int_{\mathcal{X}} p(\mathbf{t}|\mathbf{x}) p(\mathbf{x}|m)\, d\mathbf{x}. \tag{2.6}$$

By making the latent space distribution more complex we can have a simpler mapping $\mathbf{f}$. The independent factor analysis model of Attias (1999), discussed in section 2.6.4, uses this idea, where $p(\mathbf{x})$ is a product of Gaussian mixtures, the mapping is linear and the noise model is normal.

2.3.3 Smooth mapping from latent onto data space: preservation of topographic structure

In section 2.3 we required the mapping $\mathbf{f} : \mathcal{X} \to \mathcal{T}$ to be smooth, that is, continuous and differentiable. There are two reasons for this:

- Continuity: this, by definition, will guarantee that points which lie close to each other in the latent space will be mapped onto points which will be close to each other in data space. In the context of Kohonen maps and similar algorithms for dimensionality reduction this is called topology or topography preservation (although quantifying this topography preservation is difficult; Bauer and Pawelzik, 1992; Martinetz and Schulten, 1994; Kohonen, 1995; Bezdek and Pal, 1995; Goodhill and Sejnowski, 1997; Villmann et al., 1997; Bauer et al., 1999). It expresses the essential requirement that a continuous trajectory followed by a point in latent space will generate a continuous trajectory in data space (without abrupt jumps). The question of whether the dimensionality reduction mapping from data space onto latent space is continuous will be dealt with in section 2.9.2.
- (Piecewise) differentiability: this is more of a practical requirement in order to be able to use derivative-based optimisation methods, such as gradient descent.

The more general the class of functions from which we can pick $\mathbf{f}$ is (expressed through its parameters), the more flexible the latent variable model is (in that more data space distributions $p(\mathbf{t})$ can be constructed for a given prior $p(\mathbf{x})$, eq. (2.3)), but also the more complex the mathematical treatment becomes and the more local optima appear. The quality of the local optima found for models which are very flexible can be dreadful, as that shown in fig. 2.13 for GTM. The problem of local optima is very serious since it affects all local optimisation methods, i.e., methods that start somewhere in parameter space and move towards a nearby optimum, such as the EM algorithm and gradient or Newton methods. The only way for such methods to find a good optimum is to start them from many different locations, but this really does not guarantee any good results.

In section 2.3 we also required the mapping $\mathbf{f}$ to be nonsingular (as defined in section 2.2.2). This is to ensure that the manifold $\mathcal{M} = \mathbf{f}(\mathcal{X})$ has the same dimension as the latent space $\mathcal{X}$; otherwise we would be wasting latent variables.

2.4 The problem of the marginalisation in high dimensions

The latent variable framework is very general, accomodating arbitrary mappings and probability distributions. However, this presents insurmountable mathematical and computational difficulties—particularly in the analytical evaluation of integral (2.3) but also when maximising the log-likelihood (2.9)—so that the actual choice

[13] It is not enough that a curve self-intersects, as it nearly happens in plot A of fig. 2.13: the almost coincident end points $\mathbf{f}(x_1)$ and $\mathbf{f}(x_K)$ of the model manifold correspond to the widely separated end latent grid points x_1 and x_K.

is limited. In fact, the only[14] tractable case in arbitrary dimensions seems to be when both the prior in latent space $p(\mathbf{x})$ and the noise model $p(\mathbf{t}|\mathbf{x})$ are Gaussian (or mixtures of Gaussians) and the mapping $\mathbf{f}$ linear; or when $p(\mathbf{x})$ is a mixture of Dirac deltas (as a result of Monte Carlo sampling) and the mapping is nonlinear; or when $p(\mathbf{t}|\mathbf{x})$ is a Dirac delta[15]. Combinations of these give the specific latent variable models of section 2.6.

Conditioning or marginalising a multivariate distribution, such as the joint distribution $p(\mathbf{t}, \mathbf{x})$ of eq. (2.3), requires the evaluation of an integral in several dimensions of the form:

$$I(\mathbf{u}) = \int_{\mathcal{V}} \mathbf{f}(\mathbf{u}, \mathbf{v}) p(\mathbf{v})\, d\mathbf{v} \qquad \mathbf{u} \in \mathcal{U} \subseteq \mathbb{R}^U, \quad \mathbf{v} \in \mathcal{V} \subseteq \mathbb{R}^V. \tag{2.7}$$

This integral cannot be evaluated analytically for most forms of the function $\mathbf{f}$ and the distribution $p(\mathbf{v})$. In this case, a conceptually simple but computationally expensive workaround is to approximate it by Monte Carlo integration, as MacKay (1995a) suggested. For the case (2.7) this means sampling K times in the space $\mathcal{V}$ from $p(\mathbf{v})$ and approximating $I(\mathbf{u}) \approx \frac{1}{K} \sum_{k=1}^K \mathbf{f}(\mathbf{u}, \mathbf{v}_k)$, with an error of order $1/\sqrt{K}$ (Press et al., 1992, section 7.6). However, the sample size K in a space of V dimensions grows exponentially with V, as does the hypervolume of the V-dimensional region (this is basically the curse of the dimensionality, discussed in section 4.3). This severely limits the practical use of Monte Carlo methods. However, it should be noted that K is not a parameter of the model (it does not take part in the estimation) and so increasing it does not produce overfitting.

For eq. (2.3) Monte Carlo sampling yields

$$p(\mathbf{t}) \approx \frac{1}{K} \sum_{k=1}^K p(\mathbf{t}|\mathbf{x}_k) \tag{2.8}$$

with $\{\mathbf{x}_k\}_{k=1}^K$ drawn from $p(\mathbf{x})$. GTM (section 2.6.5) uses this method with a uniform $p(\mathbf{x})$. For the log-likelihood gradient $\nabla_{\boldsymbol{\Theta}} \mathcal{L}$ of eq. (2.10) Monte Carlo sampling yields

$$\nabla_{\boldsymbol{\Theta}} \mathcal{L}(\boldsymbol{\Theta}) \approx \sum_{n=1}^N \frac{\sum_{k=1}^K \nabla_{\boldsymbol{\Theta}} p(\mathbf{t}_n|\mathbf{x}_k, \boldsymbol{\Theta})}{\sum_{k=1}^K p(\mathbf{t}_n|\mathbf{x}_k, \boldsymbol{\Theta})}.$$

The aim of integral (2.3) is not to obtain the numerical value of $p(\mathbf{t})$ at a given $\mathbf{t}$, but to obtain an analytical expression for $p(\mathbf{t})$ dependent on the parameters $\boldsymbol{\Theta}$ (section 2.5). This is so because to estimate the model parameters (e.g. by an EM algorithm) we generally need to be able to take the derivative of $p(\mathbf{t}|\boldsymbol{\Theta})$ with respect to $\boldsymbol{\Theta}$.

2.5 Parameter estimation

The prior in latent space $p(\mathbf{x})$, the smooth mapping $\mathbf{f}$ and the noise model $p(\mathbf{t}|\mathbf{x})$ are all equipped with parameters[16] which we collectively call $\boldsymbol{\Theta}$. Once their functional forms are fixed, we have freedom to set the parameters to those values that agree best with the data sample. These parameters are optimised, typically to maximise the likelihood of the observed data given the parameters, $p(\mathbf{t}_n|\boldsymbol{\Theta})$. This approach has the well-known problems of overfitting and model selection and could be overcome by a Bayesian treatment. In a Bayesian treatment, a prior distribution is placed on the parameters and all subsequent inferences are done by marginalising over the parameters:

$$p(\mathbf{t}, \mathbf{x}) = \int p(\mathbf{t}, \mathbf{x}|\boldsymbol{\Theta}) p(\boldsymbol{\Theta})\, d\boldsymbol{\Theta}$$

where $p(\boldsymbol{\Theta})$ is the prior parameter distribution or the posterior parameter distribution after having seen some data $\{\mathbf{t}_n\}_{n=1}^N$. However, this adds an extra degree of intractability to that of eq. (2.3) and approximations are required. For example, there is current interest in approximate Bayesian inference using Markov chain

[14] One might think that using uniform distributions may simplify the mathematics. However, while they do simplify the expression of the integrand in (2.3), they complicate the integration region, preventing further treatment for any kind of mapping (Carreira-Perpiñán, 1997).

[15] This is the zero-noise case, which is not interesting since data space points not in $\mathbf{f}(\mathcal{X})$ receive zero density (although the standard formulation of ICA has zero noise).

[16] Strictly, the dimensionality L of the latent space is also a parameter of the latent variable model, but we will consider it fixed to some value, due to the practical difficulty of optimising it jointly with the other parameters.

Monte Carlo methods (Besag and Green, 1993; Brooks, 1998; Gilks et al., 1996; Neal, 1993) and variational methods (Jordan et al., 1999), and this has been applied to some latent variable models, such as principal component analysis (Bishop, 1999) and mixtures of factor analysers (Ghahramani and Beal, 2000; Utsugi and Kumagai, 2001). However, these are still preliminary results and the potential gain of using the Bayesian treatment (notably the autodetection of the optimal number of latent variables and mixture components) may not warrant the enormous complication of the computations, at least in high dimensions. In this thesis we will only consider maximum likelihood estimation unless indicated otherwise.

The log-likelihood of the parameters given the sample $\{\mathbf{t}_n\}_{n=1}^N$ is (assuming $\mathbf{t}_1, \dots, \mathbf{t}_N$ independent and identically distributed random variables):

$$\mathcal{L}(\mathbf{\Theta}) \stackrel{\text{def}}{=} \ln p(\mathbf{t}_1, \dots, \mathbf{t}_N|\mathbf{\Theta}) = \ln \prod_{n=1}^N p(\mathbf{t}_n|\mathbf{\Theta}) = \sum_{n=1}^N \ln p(\mathbf{t}_n|\mathbf{\Theta}) \tag{2.9}$$

which is to be maximised under the maximum likelihood criterion for parameter estimation. This will provide with a set of values for the parameters, $\mathbf{\Theta}^* = \arg\max_{\mathbf{\Theta}} \mathcal{L}(\mathbf{\Theta})$, corresponding to a (local) maximum of the log-likelihood. The log-likelihood value $\mathcal{L}(\mathbf{\Theta}^*)$ allows the comparison of any two latent variable models (or, in general, any two probability models), however different these may be (although from a Bayesian point of view, in addition to their likelihood, one should take into account a prior distribution for the parameters and the evidence for the model; MacKay, 1995b).

One maximisation strategy is to find the stationary points of the log-likelihood (2.9):

$$\nabla_{\mathbf{\Theta}} \mathcal{L}(\mathbf{\Theta}) = \sum_{n=1}^N \frac{1}{p(\mathbf{t}_n|\mathbf{\Theta})} \nabla_{\mathbf{\Theta}} p(\mathbf{t}_n|\mathbf{\Theta}) = \mathbf{0} \tag{2.10}$$

but maximum likelihood optimisation is often carried out using an **EM algorithm** (Dempster et al., 1977; McLachlan and Krishnan, 1997), which is usually simpler and is guaranteed to increase the log-likelihood monotonically. In the EM approach to latent variable models, the latent variables $\{\mathbf{x}_n\}_{n=1}^N$ (one per data point) are considered missing[17]. If their values were known, estimation of the parameters (e.g. the $\mathbf{\Lambda}$ matrix in eq. (2.14)) would be straightforward by least squares. However, for a given data point $\mathbf{t}_n$ we do not know the value of $\mathbf{x}_n$ that generated it. The EM algorithm operates in two steps which are repeated alternatively until convergence:

E step computes the expectation of the complete data log-likelihood with respect to the current posterior distribution $p(\mathbf{x}_n|\mathbf{t}_n, \mathbf{\Theta}^{(\tau)})$ (i.e., using the current parameter values), traditionally notated $Q(\mathbf{\Theta}|\mathbf{\Theta}^{(\tau)})$:

$$Q(\mathbf{\Theta}|\mathbf{\Theta}^{(\tau)}) \stackrel{\text{def}}{=} \sum_{n=1}^N \mathrm{E}_{p(\mathbf{x}_n|\mathbf{t}_n,\mathbf{\Theta}^{(\tau)})} \{\mathcal{L}_{n,\text{complete}}(\mathbf{\Theta})\} \text{ where } \mathcal{L}_{n,\text{complete}}(\mathbf{\Theta}) \stackrel{\text{def}}{=} \ln p(\mathbf{t}_n, \mathbf{x}_n|\mathbf{\Theta}).$$

Thus, we average over the missing latent variables $\{\mathbf{x}_n\}_{n=1}^N$, effectively filling in their unknown values. Computing $\mathcal{L}_{n,\text{complete}}(\mathbf{\Theta})$ is possible because the joint distribution $p(\mathbf{t}, \mathbf{x}|\mathbf{\Theta})$ is known for the latent variable model in question.

M step determines new parameter values $\mathbf{\Theta}^{(\tau+1)}$ that maximise the expected complete-data log-likelihood:

$$\mathbf{\Theta}^{(\tau+1)} = \arg\max_{\mathbf{\Theta}} Q(\mathbf{\Theta}|\mathbf{\Theta}^{(\tau)}).$$

This increases the log-likelihood $\mathcal{L}(\mathbf{\Theta})$ unless it is already at a local maximum.

The standard EM algorithm has some disadvantages:

- It is a batch algorithm. However, by interpreting EM as an alternating maximisation of a negative free-energy-like function (Neal and Hinton, 1998), it is possible to derive online EM algorithms, suitable for online learning (e.g. in sequential tasks, where the data come one at a time).
- Its slow convergence after the first few steps, which are usually quite effective. Also, the greater the proportion of missing information, the slower the rate of convergence of EM (Dempster et al., 1977). However, methods for accelerating it are available; see, for example, Meng and van Dyk (1997), McLachlan and Krishnan (1997) and references therein.

[17]Depending on the model, additional missing variables may have to be introduced. For example, the component labels for mixture models.

Despite these shortcomings, EM usually remains the best choice for parameter estimation thanks to its reliability.

For a general choice of prior in latent space, mapping between latent and data space and noise model the log-likelihood surface can have many local maxima of varying height. In some cases, some or all of those maxima are equivalent, in the sense that the model produces the same distribution (and therefore the same log-likelihood value at all the maxima), i.e., the model is not identifiable (section 2.8). This is often due to symmetries of the parameter space, such as permutations (e.g. PCA) or general rotations of the parameters (e.g. factor analysis). In those cases, the procedure to follow is to find a first maximum likelihood estimate of the parameters (in general, by some suitable optimisation method, e.g. EM, although sometimes an analytical solution is available, as for PCA) and then possibly apply a transformation to them to take them to a canonical form satisfying a certain criterion (e.g. varimax rotation in factor analysis).

2.5.1 Relation of maximum likelihood with other estimation criteria

Least squares Using the least-squares reconstruction error as objective function for parameter estimation gives in general different estimates as the maximum likelihood criterion, although the latter usually results in a low reconstruction error. If we consider the unobserved values $\{\mathbf{x}_n\}_{n=1}^N$ as fixed parameters rather than random variables and assume that the noise model is normal with isotropic known variance, then a penalised maximum likelihood criterion results in

$$\sum_{n=1}^{N} \left\|\mathbf{t}_n - \mathbf{f}(\mathbf{x}_n)\right\|^2 + \text{ penalty term on } \mathbf{f}$$

which coincides with the spline-related definition of principal curves given by Hastie and Stuetzle (1989), mentioned in section 4.8.

Kullback-Leibler distance For $N \to \infty$, the normalised log-likelihood of $\boldsymbol{\Theta}$ converges in probability to its expectation by the law of large numbers:

$$\begin{aligned} \mathcal{L}_N(\boldsymbol{\Theta}) \stackrel{\text{def}}{=} \frac{1}{N}\sum_{n=1}^{N} \ln p(\mathbf{t}_n|\boldsymbol{\Theta}) \xrightarrow{\mathcal{P}} \mathcal{L}_\infty(\boldsymbol{\Theta}) &\stackrel{\text{def}}{=} \mathrm{E}_{p_\mathbf{t}}\left\{\ln p(\mathbf{t}|\boldsymbol{\Theta})\right\} \\ &= \int_{\mathcal{T}} p_\mathbf{t}(\mathbf{t}) \ln p(\mathbf{t}|\boldsymbol{\Theta})\, d\mathbf{t} = -h(p_\mathbf{t}) - \mathrm{D}\left(p_\mathbf{t} \| p(\cdot|\boldsymbol{\Theta})\right) \end{aligned} \tag{2.11}$$

for any $\boldsymbol{\Theta}$. Since the entropy of the data distribution $h(p_\mathbf{t})$ does not depend on the parameters $\boldsymbol{\Theta}$, maximising the log-likelihood is asymptotically equivalent to minimising the Kullback-Leibler distance to the data density.

2.5.2 Relation with nonparametric estimation of continuous mixtures

As mentioned in section 2.3, the fundamental equation (2.3) can be interpreted as a continuous mixture model for $\mathbf{t}$, where $\mathbf{x}$ is the mixing variable. Assume that the functional form of $p(\mathbf{t}|\mathbf{x})$ is known and depends on $\mathbf{f}(\mathbf{x})$ and on parameters $\boldsymbol{\theta}(\mathbf{x})$. The log-likelihood of this model is (call $p_\mathbf{x}$ the density of the mixing variable $\mathbf{x}$):

$$\mathcal{L}(p_\mathbf{x}, \mathbf{f}, \boldsymbol{\theta}) \stackrel{\text{def}}{=} \sum_{n=1}^{N} \ln \int_{\mathcal{X}} p(\mathbf{t}_n|\mathbf{f}(\mathbf{x}), \boldsymbol{\theta}(\mathbf{x})) p(\mathbf{x})\, d\mathbf{x}.$$

Results from the theory of mixtures (Laird, 1978; Lindsay, 1983) dictate that for fixed $\mathbf{f}$ and $\boldsymbol{\theta}$ the nonparametric maximum likelihood estimate for $p(\mathbf{x})$ uniquely exists and is discrete with at most N support points (where N is the sample size). Denote these support points by $\{\mathbf{x}_m\}_{m=1}^M \subset \mathcal{X}$ where $M \leq N$. This results then in

$$p(\mathbf{t}; \boldsymbol{\Theta}) = \sum_{m=1}^{M} p_m p(\mathbf{t}|\mathbf{f}_m, \boldsymbol{\theta}_m) \tag{2.12}$$

where $\boldsymbol{\Theta} \stackrel{\text{def}}{=} \{p_m, \mathbf{f}_m, \boldsymbol{\theta}_m\}_{m=1}^M$ contains the values of $\mathbf{f}$, $\boldsymbol{\theta}$ and $p_\mathbf{x}$ at the M unknown support points:

$$\mathbf{f}_m \stackrel{\text{def}}{=} \mathbf{f}(\mathbf{x}_m) \qquad \boldsymbol{\theta}_m \stackrel{\text{def}}{=} \boldsymbol{\theta}(\mathbf{x}_m) \qquad p_m \stackrel{\text{def}}{=} p(\mathbf{x}_m).$$

Model	*Prior in latent space* $p(\mathbf{x})$	*Mapping* $\mathbf{f}$ $\mathbf{x} \to \mathbf{t}$	*Noise model* $p(\mathbf{t}\|\mathbf{x})$	*Density in observed space* $p(\mathbf{t})$
Factor analysis (FA)	$\mathcal{N}(\mathbf{0}, \mathbf{I})$	linear	diagonal normal	constrained Gaussian
Principal component analysis (PCA)	$\mathcal{N}(\mathbf{0}, \mathbf{I})$	linear	spherical normal	constrained Gaussian
Independent component analysis (ICA)	unknown but factorised	linear	Dirac delta	depends
Independent factor analysis (IFA)	product of 1D Gaussian mixtures	linear	normal	constrained Gaussian mixture
Generative topographic mapping (GTM)	discrete uniform	generalised linear model	spherical normal	constrained Gaussian mixture

Table 2.2: Summary of specific continuous latent variable models.

So we end up with a parametric finite mixture model of M components and parameters $\mathbf{\Theta}$ which must be estimated from the sample $\{\mathbf{x}_n\}_{n=1}^N$: $\{p_m\}$ are the mixing proportions and, for each component m, $\mathbf{f}_m$ and $\boldsymbol{\theta}_m$ could be (for example) location and covariance parameters. Note that the log-likelihood is not a function of the support points $\{\mathbf{x}_m\}_{m=1}^M$, but only of $\mathbf{\Theta}$. An obvious choice of optimisation algorithm would be EM (Redner and Walker, 1984; McLachlan and Krishnan, 1997).

The nonparametric approach to parameter estimation in latent variable models has an important disadvantage: it only finds the values of $\mathbf{f}$ and $p_{\mathbf{x}}$ at the support points. To obtain their values at other, intermediate, points—which is necessary to perform dimensionality reduction and reconstruction (section 2.9)—they must be smoothed or interpolated using the known values, $\{\mathbf{f}_m\}_{m=1}^M$ for $\mathbf{f}$ and $\{p_m\}_{m=1}^M$ for $p_{\mathbf{x}}$. This is as hard a problem as the original one of estimating $\mathbf{f}$ parametrically. A less important disadvantage is that of the singularities of the log-likelihood due to the use of separate parameters for each component: if $\mathbf{f}_m$ becomes equal to some data point $\mathbf{t}_n$, then if its variance parameter $\boldsymbol{\theta}_m \to \mathbf{0}$ then $\mathcal{L} \to \infty$. This requires regularising the model.

Although the expression for $p(\mathbf{t})$ in eq. (2.12) looks similar to the one obtained by Monte Carlo sampling of the latent space, eq. (2.8), there is an important difference: the Monte Carlo method preserves a parametric form for the mapping $\mathbf{f}$, which the nonparametric method has lost.

Knowing that the number of support points is at most N is not useful in practice, since fitting a mixture of more than N components (each with separate parameters) to N data points would result in overfitting as well as being computationally expensive.

2.6 Specific latent variable models

A latent variable model is specified by the functional forms of:

- the prior in latent space $p(\mathbf{x})$
- the smooth mapping $\mathbf{f} : \mathcal{X} \to \mathcal{T}$ from latent space to data space
- the noise model in data space $p(\mathbf{t}|\mathbf{x})$

all of which are equipped with parameters (as before, we omit them for clarity). As mentioned in section 2.4, analytically tractable models can be obtained by clever combinations of (mixtures of) normal distributions with linear mappings or mixtures of Dirac deltas with nonlinear mappings.

We describe here several well-known specific latent variable models. Table 2.2 gives a summary of them. In all cases the parameters of the model may be estimated using the EM algorithm (in the case of PCA, an analytical solution is known as well). In all cases $\mathcal{T} \equiv \mathbb{R}^D$ and $\mathcal{X} \equiv \mathbb{R}^L$ except for GTM, which uses a discrete prior latent space.

Latent variable models can be classified as *linear* and *nonlinear* according to the corresponding character of the mapping $\mathbf{f}$. We call a latent variable model *normal* when both the prior in latent space and the noise model are normal. Thus, factor analysis and principal component analysis (defined below) are *linear-normal* latent variable models.

2.6.1 Factor analysis (FA)

Factor analysis[18] (Bartholomew, 1987; Everitt, 1984) uses a Gaussian distributed prior and noise model, and a linear mapping from data space to latent space. Specifically:

- The latent space prior $p(\mathbf{x})$ is unit normal:

$$\mathbf{x} \overset{\text{def}}{\sim} \mathcal{N}(\mathbf{0}, \mathbf{I}) \tag{2.13}$$

  although there exist varieties of factor analysis where these factors are correlated. The latent variables $\mathbf{x}$ are often referred to as the *factors*.

- The mapping $\mathbf{f}$ is linear:

$$\mathbf{f}(\mathbf{x}) \overset{\text{def}}{=} \mathbf{\Lambda}\mathbf{x} + \boldsymbol{\mu}. \tag{2.14}$$

  The columns of the $D \times L$ matrix $\mathbf{\Lambda}$ are referred to as the *factor loadings*. We assume $\operatorname{rank}(\mathbf{\Lambda}) = L$, i.e., linearly independent factors.

- The data space noise model is normal centred at $\mathbf{f}(\mathbf{x})$ with diagonal covariance matrix $\mathbf{\Psi}$:

$$\mathbf{t}|\mathbf{x} \overset{\text{def}}{\sim} \mathcal{N}(\mathbf{f}(\mathbf{x}), \mathbf{\Psi}). \tag{2.15}$$

  The D diagonal elements of $\mathbf{\Psi}$ are referred to as the *uniquenesses*.

The marginal distribution in data space can be computed analytically and it turns out to be normal with a constrained covariance matrix (theorems 2.12.1 and A.3.1(iv)):

$$\mathbf{t} \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{\Lambda}\mathbf{\Lambda}^T + \mathbf{\Psi}). \tag{2.16}$$

The posterior in latent space is also normal:

$$\mathbf{x}|\mathbf{t} \sim \mathcal{N}\left(\mathbf{A}(\mathbf{t} - \boldsymbol{\mu}), (\mathbf{I} + \mathbf{\Lambda}^T\mathbf{\Psi}^{-1}\mathbf{\Lambda})^{-1}\right) \tag{2.17}$$

$$\mathbf{A} = \mathbf{\Lambda}^T(\mathbf{\Lambda}\mathbf{\Lambda}^T + \mathbf{\Psi})^{-1} = (\mathbf{I} + \mathbf{\Lambda}^T\mathbf{\Psi}^{-1}\mathbf{\Lambda})^{-1}\mathbf{\Lambda}^T\mathbf{\Psi}^{-1}. \tag{2.18}$$

The reduced-dimension representative (defined in section 2.9.1) is taken as the posterior mean (coinciding with the mode) and is usually referred to as the *Thomson scores*:

$$\mathbf{F}(\mathbf{t}) \overset{\text{def}}{=} \mathrm{E}\{\mathbf{x}|\mathbf{t}\} = \mathbf{A}(\mathbf{t} - \boldsymbol{\mu}). \tag{2.19}$$

The dimensionality reduction mapping $\mathbf{F}$ is linear and therefore smooth. As discussed in section 2.9.1.1, factor analysis with Thomson scores does not satisfy the condition that $\mathbf{F} \circ \mathbf{f}$ be the identity, because $\mathbf{A}\mathbf{\Lambda} \neq \mathbf{I}$, except in the zero-noise limit.

If we apply an invertible linear transformation $\mathbf{g}$ with matrix $\mathbf{R}$ to the factors $\mathbf{x}$ to obtain a new set of factors $\mathbf{y} = \mathbf{R}\mathbf{x}$, the prior distribution $p(\mathbf{y})$ is still normal (theorem A.3.1(i)), $\mathbf{y} \sim \mathcal{N}(\mathbf{0}, \mathbf{R}\mathbf{R}^T)$, and the new mapping becomes $\mathbf{t} = \mathbf{f}'(\mathbf{y}) = \mathbf{f}(\mathbf{g}^{-1}(\mathbf{y})) = \mathbf{\Lambda}\mathbf{R}^{-1}\mathbf{y} + \boldsymbol{\mu}$ (section 2.3.2). That is, the new factor loadings become $\mathbf{\Lambda}' = \mathbf{\Lambda}\mathbf{R}^{-1}$. If $\mathbf{R}$ is an orthogonal matrix, i.e., $\mathbf{R}^{-1} = \mathbf{R}^T$, the new factors $\mathbf{y}$ will still be independent and $\mathbf{\Psi}' = \mathbf{\Psi}$ diagonal; this is called an *orthogonal rotation* of the factors in the literature of factor analysis. If $\mathbf{R}$ is an arbitrary nonsingular matrix, the new factors $\mathbf{y}$ will not be independent anymore; this is called an *oblique rotation* of the factors. Thus, from all the factor loadings matrices $\mathbf{\Lambda}$, we are free to choose that which is easiest to interpret according to some criterion, e.g. by varimax rotation[19]. However, we insist that, provided that the model $p(\mathbf{t})$ remains the same, all transformations—orthogonal or oblique—are equally valid. Section 2.8.1 further discusses this issue.

The log-likelihood of the parameters[20] $\mathbf{\Theta} = \{\mathbf{\Lambda}, \mathbf{\Psi}, \boldsymbol{\mu}\}$ is obtained as the log-likelihood of a normal distribution $\mathcal{N}(\boldsymbol{\mu}, \mathbf{\Sigma})$ of covariance $\mathbf{\Sigma} = \mathbf{\Lambda}\mathbf{\Lambda}^T + \mathbf{\Psi}$:

$$\mathcal{L}(\mathbf{\Lambda}, \mathbf{\Psi}) = -\frac{N}{2}\left(D \ln 2\pi + \ln|\mathbf{\Sigma}| + \operatorname{tr}\left(\mathbf{S}\mathbf{\Sigma}^{-1}\right)\right) \tag{2.20}$$

[18] Our Matlab implementation of factor analysis (EM algorithm, Rao's algorithm, scores, χ^2-test, etc.) and varimax rotation is freely available in the Internet (see appendix C).

[19] Varimax rotation (Kaiser, 1958) finds an orthogonal rotation of the factors such that, for each new factor, the loadings are either very large or very small (in absolute value). The resulting rotated matrix $\mathbf{\Lambda}'$ has many values clamped to (almost) 0, that is, each factor involves only a few of the original variables. This simplifies factor interpretation.

[20] The maximum likelihood estimate of the location parameter $\boldsymbol{\mu}$ is the sample mean $\bar{\mathbf{t}}$. If the covariance matrix of $p(\mathbf{t})$ in eq. (2.16) was unconstrained, the problem would be that of fitting a normal distribution $\mathcal{N}(\boldsymbol{\mu}, \mathbf{\Sigma})$ to the sample $\{\mathbf{t}_n\}_{n=1}^N$. In this case, the maximum likelihood estimates for $\boldsymbol{\mu}$ and $\mathbf{\Sigma}$ would be the natural ones: the sample mean $\bar{\mathbf{t}}$ and the sample covariance matrix $\mathbf{S}$, respectively (Mardia et al., 1979).

where $\mathbf{S} \stackrel{\text{def}}{=} \frac{1}{N}\sum_{n=1}^{N}(\mathbf{t}_n - \overline{\mathbf{t}})(\mathbf{t}_n - \overline{\mathbf{t}})^T$ is the sample covariance matrix and $\overline{\mathbf{t}} \stackrel{\text{def}}{=} \frac{1}{N}\sum_{n=1}^{N}\mathbf{t}_n$ the sample mean. The log-likelihood gradient is:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{\Lambda}} = -N(\mathbf{\Sigma}^{-1}(\mathbf{I} - \mathbf{S}\mathbf{\Sigma}^{-1})\mathbf{\Lambda}) \qquad \frac{\partial \mathcal{L}}{\partial \mathbf{\Psi}} = -\frac{N}{2}\operatorname{diag}\left(\mathbf{\Sigma}^{-1}(\mathbf{I} - \mathbf{S}\mathbf{\Sigma}^{-1})\right).$$

The log-likelihood has infinite equivalent maxima resulting from orthogonal rotation of the factors. Apart from these, it is not clear whether the log-likelihood has a unique global maximum or there exist suboptimal ones. In theory, different local maxima are possible and should be due to an underconstrained model (e.g. a small sample), but there does not seem to be any evidence in the factor analysis literature about the frequency of multiple local maxima with actual data. Rubin and Thayer (1982), Bentler and Tanaka (1983) and Rubin and Thayer (1983) give an interesting discussion about this[21].

The parameters of a factor analysis model may be estimated using an EM algorithm (Rubin and Thayer, 1982):

E step: This requires computing the moments:

$$\mathrm{E}\left\{\mathbf{x}|\mathbf{t}_n\right\} = \mathbf{A}^{(\tau)}(\mathbf{t}_n - \boldsymbol{\mu})$$
$$\mathrm{E}\left\{\mathbf{x}\mathbf{x}^T|\mathbf{t}_n\right\} = \mathbf{I} - \mathbf{A}^{(\tau)}\mathbf{\Lambda}^{(\tau)} + \mathbf{A}^{(\tau)}(\mathbf{t}_n - \boldsymbol{\mu})(\mathbf{t}_n - \boldsymbol{\mu})^T(\mathbf{A}^{(\tau)})^T$$

for each data point $\mathbf{t}_n$ given the current parameter values $\mathbf{\Lambda}^{(\tau)}$ and $\mathbf{\Psi}^{(\tau)}$.

M step: This results in the following update equations for the factor loadings $\mathbf{\Lambda}$ and uniquenesses $\mathbf{\Psi}$:

$$\mathbf{\Lambda}^{(\tau+1)} = \left(\sum_{n=1}^{N}\mathbf{t}_n \,\mathrm{E}\left\{\mathbf{x}|\mathbf{t}_n\right\}^T\right)\left(\sum_{n=1}^{N}\mathrm{E}\left\{\mathbf{x}\mathbf{x}^T|\mathbf{t}_n\right\}^T\right)^{-1}$$
$$\mathbf{\Psi}^{(\tau+1)} = \frac{1}{N}\operatorname{diag}\left(\sum_{n=1}^{N}\mathbf{t}_n\mathbf{t}_n^T - \mathbf{\Lambda}^{(\tau+1)}\,\mathrm{E}\left\{\mathbf{x}|\mathbf{t}_n\right\}\mathbf{t}_n^T\right)$$

where the updated moments are used and the "diag" operator sets all the off-diagonal elements of a matrix to zero.

The location parameter $\boldsymbol{\mu}$ is estimated by the sample mean, and does not need to take part in the EM algorithm.

Apart from EM, there are a number of other methods for maximum likelihood parameter estimation for factor analysis, such as the methods of Jöreskog (1967)[22] or Rao (Morrison, 1990, pp. 357–362). Also, in common with other probabilistic models, factor analysis may be implemented using an autoencoder network, with weights implementing a recognition and a generative model: a special case of the Helmholtz machine having two layers of linear units with Gaussian noise, and trained with the wake-sleep algorithm[23] (Neal and Dayan, 1997).

There are also estimation criteria for factor analysis other than maximum likelihood, e.g. principal factors (Harman, 1967) or minimum trace factor analysis (Jamshidian and Bentler, 1998).

2.6.2 Principal component analysis (PCA)

Principal component analysis[24] (PCA) can be seen as a maximum likelihood factor analysis in which the uniquenesses are constrained to be equal, that is, $\mathbf{\Psi} = \sigma^2\mathbf{I}$ is isotropic. This simple fact, already reported in the early factor analysis literature, seems to have gone unnoticed until Tipping and Bishop (1999b) and Roweis (1998) recently rediscovered it. Indeed, a number of textbooks and papers (e.g. Krzanowski, 1988,

[21]Rubin and Thayer (1982), reanalysing an example with 9 observed variables, 4 factors and around 36 free parameters in which Jöreskog had found a single maximum with the LISREL method, claimed to find several additional maxima of the log-likelihood using their EM algorithm. By carefully checking the gradient and the Hessian of the log-likelihood at those points, Bentler and Tanaka (1983) showed that none except Jöreskog's were really maxima. This is disquieting in view of the large number of parameters used in pattern recognition applications.

[22]The method of Jöreskog (1967) is a second-order Fletcher-Powell method. It is also applicable to *confirmatory factor analysis*, where some of the loadings have been set to fixed values (usually zero) according to the judgement of the user (Jöreskog, 1969).

[23]Strictly speaking, Neal and Dayan (1997) give no proof that wake-sleep learning works for factor analysis, only empirical support that it usually does in some simulations.

[24]Our Matlab implementation of principal component analysis is freely available in the Internet (see appendix C).

p. 502 or Hinton et al., 1997, beginning of section III) wrongly quote that "PCA does not propose a model for the data" as a disadvantage when compared with factor analysis.

The approach of considering an isotropic noise model in factor analysis had already been adopted in the Young-Whittle factor analysis model (Young, 1940; Whittle, 1952) and its maximum likelihood solution assuming σ known found analytically (Anderson, 1963; Basilevsky, 1994, pp. 361–363). Lawley (1953) and Anderson and Rubin (1956) showed that stationary points of the log-likelihood as a function of the loadings $\mathbf{\Lambda}$ and uniqueness σ^2 occur at the value of eq. (2.28), although they did not prove it to be global maximum, which Tipping and Bishop (1999b) did. In addition to this direct solution, Roweis (1998) and Tipping and Bishop (1999b) give an EM algorithm for estimating $\mathbf{\Lambda}$ and σ^2.

Consider then the factor analysis model of the previous section with an isotropic error model. There exists a unique (although possibly degenerate, if some eigenvalues are equal) maximum likelihood estimate closely related to the L principal components of the data[25]. If the sample covariance matrix is decomposed as $\mathbf{S} = \mathbf{U}\mathbf{V}\mathbf{U}^T$, with $\mathbf{V} = \operatorname{diag}(v_1, \dots, v_D)$ containing the eigenvalues (ordered decreasingly) and $\mathbf{U} = (\mathbf{u}_1, \dots, \mathbf{u}_D)$ the associated eigenvectors, then $\mathbf{\Lambda} = \mathbf{U}_L(\mathbf{V}_L - \sigma^2\mathbf{I})^{1/2}$ with $\mathbf{U}_L = (\mathbf{u}_1, \dots, \mathbf{u}_L)$, $\mathbf{V}_L = \operatorname{diag}(v_1, \dots, v_L)$ and $\sigma^2 = \frac{1}{D-L}\sum_{j=L+1}^{D} v_j$. Therefore eqs. (2.13)–(2.19) become:

$$\mathbf{x} \stackrel{\text{def}}{\sim} \mathcal{N}(\mathbf{0}, \mathbf{I}) \tag{2.21}$$

$$\mathbf{f}(\mathbf{x}) \stackrel{\text{def}}{=} \mathbf{\Lambda}\mathbf{x} + \boldsymbol{\mu} \tag{2.22}$$

$$\mathbf{t}|\mathbf{x} \stackrel{\text{def}}{\sim} \mathcal{N}(\mathbf{f}(\mathbf{x}), \sigma^2\mathbf{I}) \tag{2.23}$$

$$\mathbf{t} \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{\Lambda}\mathbf{\Lambda}^T + \sigma^2\mathbf{I}) \tag{2.24}$$

$$\mathbf{x}|\mathbf{t} \sim \mathcal{N}\left(\mathbf{A}(\mathbf{t} - \boldsymbol{\mu}), \sigma^2\mathbf{V}_L^{-1}\right) \tag{2.25}$$

$$\mathbf{A} = \mathbf{V}_L^{-1}(\mathbf{V}_L - \sigma^2\mathbf{I})^{1/2}\mathbf{U}_L^T \tag{2.26}$$

$$\mathbf{F}(\mathbf{t}) \stackrel{\text{def}}{=} \mathrm{E}\left\{\mathbf{x}|\mathbf{t}\right\} = \mathbf{A}(\mathbf{t} - \boldsymbol{\mu}) \tag{2.27}$$

with maximum likelihood estimates:

$$\mathbf{\Lambda} = \mathbf{U}_L(\mathbf{V}_L - \sigma^2\mathbf{I})^{1/2} \qquad \sigma^2 = \frac{1}{D-L}\sum_{j=L+1}^{D} v_j. \tag{2.28}$$

The Thomson scores give $\mathbf{F}(\mathbf{t}) = \mathrm{E}\left\{\mathbf{x}|\mathbf{t}\right\} = \mathbf{A}(\mathbf{t} - \boldsymbol{\mu})$, which still does not satisfy the condition that $\mathbf{F} \circ \mathbf{f}$ be the identity, except in the zero-noise limit ($\sigma \to 0$), as discussed in section 2.9.1.1.

Thus, in the latent variable model scenario PCA does define a probability model. Still, PCA is not usually constructed as a probability model but as a reconstruction and dimensionality reduction technique, since its most attractive property is that it is the linear mapping that minimises the least squares reconstruction error of a sample. We describe this aspect of PCA in section 4.5.

Since the noise model variance σ^2 is the same for all observed variables, the directions of the columns of $\mathbf{\Lambda}$ will be more influenced by the variables that have higher noise—unlike with factor analysis, which should be able to separate the linear correlations from the noise. The PCA model is then more restricted than the factor analysis one. However, in many practical situations the directions $\mathbf{\Lambda}$ found by factor analysis and PCA do not differ much.

The log-likelihood of the estimate (2.28) is[26]

$$\mathcal{L}(\mathbf{\Lambda}, \sigma^2) = -\frac{N}{2}\left(D(1 + \ln 2\pi) + \ln|\mathbf{S}| + (D - L)\ln\frac{a}{g}\right) \tag{2.29}$$

where a and g are the arithmetic and geometric means of the $D - L$ smallest eigenvalues of the sample covariance matrix $\mathbf{S}$. Although there is an EM algorithm that finds the L principal components by maximising the log-likelihood (Tipping and Bishop, 1999b; Roweis, 1998), the fastest way to perform PCA is via numerical singular value decomposition (Golub and van Loan, 1996; Press et al., 1992).

Again, as in factor analysis, it is possible to orthogonally rotate the latent variables keeping the same distribution.

[25] To differentiate it from conventional PCA, where no probabilistic model is explicitly defined and only a linear orthogonal projection is considered (section 4.5), this latent variable model is called *probabilistic PCA* by Tipping and Bishop (1999b) and *sensible PCA* by Roweis (1998). We do not deem necessary to use and additional, different name for what is essentially the same thing, so in this thesis we will only use the name PCA and the context will make clear what aspect we mean—probability model or linear mapping.

[26] As with factor analysis, the maximum likelihood estimate of the location parameter $\boldsymbol{\mu}$ is the sample mean $\bar{\mathbf{t}}$.

For a fixed data set, PCA has the property of *additivity*, insofar that the principal components obtained using a latent space of dimension L are exactly the same as the ones obtained using a latent space of dimension $L-1$ plus a new, additional principal component. However, this additivity property does not necessarily hold for either PCA followed by (varimax) rotation or for factor analysis. That is, the factors found by a factor analysis of order L are, in general, all different from those found by a factor analysis of order $L-1$. That means that one can only talk about the joint collection of L factors (or the linear subspace spanned by them).

2.6.3 Independent component analysis (ICA)

Independent component analysis (ICA) or blind source separation consists of recovering independent sources given only sensor observations that are unknown linear mixtures of the sources (Comon, 1994; Cardoso, 1998; Hyvärinen, 1999b; Hyvärinen et al., 2001). Its basic formulation is as follows. Denote the time variable by τ (continuous or discrete). Call $\mathbf{x}(\tau) \in \mathbb{R}^L$ the L time-varying *source signals*[27], assumed independent and zero-mean. Call $\mathbf{t}(\tau) \in \mathbb{R}^D$ the D *data signals*, measured without noise and with instantaneous mixing[28] (that is, there is no time delay between source l mixing into channel d). Then $\mathbf{t}(\tau) \stackrel{\text{def}}{=} \mathbf{\Lambda x}(\tau)$, where $\mathbf{\Lambda}_{D\times L}$ is the *mixing matrix* and $D \geq L$ (although we will assume $D = L$ in most of this section[29]). The goal of ICA is, given a sample $\{\mathbf{t}_n\}_{n=1}^N$ of the sensor outputs, to find a linear transformation $\mathbf{A}_{L\times D}$ (*separating matrix*) of the sensor signals $\mathbf{t}$ that makes the outputs $\mathbf{u}(\tau) = \mathbf{At}(\tau) = \mathbf{A\Lambda x}(\tau)$ as independent as possible. If $\mathbf{\Lambda}$ was known, $\mathbf{A} = \mathbf{\Lambda}^+ = (\mathbf{\Lambda}^T\mathbf{\Lambda})^{-1}\mathbf{\Lambda}^T$ (assuming $\operatorname{rank}(\mathbf{\Lambda}) = L$) would recover the sources exactly, but $\mathbf{\Lambda}$ is unknown; all we have is a finite set of realisations of the sensor outputs $\{\mathbf{t}_n\}_{n=1}^N$.

If we take the sources $\mathbf{x} = (x_1, \dots, x_L)^T$ as the latent variables and the sensor outputs $\mathbf{t} = (t_1, \dots, t_D)^T$ as the observed variables, ICA can be seen as a latent variable model with the following elements (the dependence on the time τ is omitted for clarity):

- The prior distribution in latent space $\mathcal{X} = \mathbb{R}^L$ is factorised but unknown: $p(\mathbf{x}) = \prod_{l=1}^L p_l(x_l)$.
- The mapping between latent and observed space is linear: $\mathbf{t} = \mathbf{\Lambda x}$.
- The noise model is a Dirac delta, i.e., the observed variables are generated without noise: $p(\mathbf{t}|\mathbf{x}) = \delta(\mathbf{t} - \mathbf{\Lambda x}) = \prod_{d=1}^D \delta(t_d - \sum_{l=1}^L a_{dl}x_l)$. However, we can imagine that noise is an independent source and segregate it from the other sources.

Therefore, the only free parameter in the latent variable model is the matrix $\mathbf{\Lambda}_{D\times L} = (\lambda_{dl})$ (or $\mathbf{A}_{L\times D} = \mathbf{\Lambda}^+ = (a_{ld})$), but it is also necessary to determine the unknown functions $\{p_l(x_l)\}_{l=1}^L$. How to deal with this unknown prior distribution is what makes ICA different from the other latent variable models discussed here. Ideally one would learn the functions $\{p_l\}_{l=1}^L$ nonparametrically, but this is difficult. For practical estimation, all current methods use a fixed functional form for the latent space prior distribution with or without parameters. A flexible parametric model of the prior distribution is a Gaussian mixture and this gives rise to the IFA model of section 2.6.4.

The very effective **infomax** learning rule of Bell and Sejnowski (1995) is the following online iterative algorithm:

$$\mathbf{A}^{(\tau+1)} = \mathbf{A}^{(\tau)} + \Delta\mathbf{A}^{(\tau+1)} \tag{2.30}$$

$$\Delta\mathbf{A}^{(\tau+1)} \propto ((\mathbf{A}^{(\tau)})^T)^{-1} + \mathbf{g}(\mathbf{u})\mathbf{t}^T \tag{2.31}$$

where $\mathbf{t}$ is a sensor output, $\mathbf{u} \stackrel{\text{def}}{=} \mathbf{A}^{(\tau)}\mathbf{t}$, $\mathbf{g}(\mathbf{u}) \stackrel{\text{def}}{=} (g(u_1), \dots, g(u_L))^T$ and $g(u) \stackrel{\text{def}}{=} \frac{\partial \ln |f'(u)|}{\partial u}$. The choice of the nonlinear function $f: \mathbb{R} \to \mathbb{R}$ is critical since it can be proven (see below and Bell and Sejnowski, 1995) that $p(x_l) \propto |f'(x_l)|$. That is, f is the one-dimensional cumulative distribution function in latent space and fixing its form is equivalent to assuming a prior distribution $p(x_l)$ for the latent variables (the sources). Taking $g(u) = -2u$ (or equivalently $f = \operatorname{erf}$) means assuming normally distributed sources, $\mathbf{x} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ (although with a Dirac-delta noise model), which gives second-order decorrelation only—in which case using factor analysis would have been enough. The use of a nonlinear function, whose Taylor series expansion has terms of many orders, may be more sensitive to the right higher-order statistics of the inputs. In practice, a sigmoidal function is chosen, such as the logistic function. Several forms of the nonlinearity f are summarised in table 2.3. The computationally simpler nonlinearities, such as the logistic function or the hyperbolic tangent, assume

[27]Instead of time-varying sources $\mathbf{x}(\tau)$ one can consider space-varying sources $\mathbf{x}(x, y)$, such as images.

[28]If time delays are considered, the problem is called *blind deconvolution* rather than *blind separation*.

[29]The case $D < L$ (more sources than sensors) is called *overcomplete representation* and has been advocated because of its greater robustness to noise, sparsity and flexibility in matching structure in the data (Amari, 1999; Lewicki and Sejnowski, 2000).

Nonlinearity name	$f(u)$	$f'(u)$	$g(u) \stackrel{\text{def}}{=} \frac{\partial}{\partial u} \ln \lvert f'(u) \rvert = \frac{f''(u)}{f'(u)}$	Kurtosis k_4
Logistic	$\frac{1}{1+e^{-u}}$	$f(1-f)$	$1-2f$	$+$
Generalised logistic		$f^p(1-f)^r$	$\left(\frac{p}{f} - \frac{r}{1-f}\right) f^p(1-f)^r$	$\begin{cases} +, & p,r>1 \\ -, & p,r<1 \end{cases}$
Hyperbolic tangent	$\tanh u$	$1-f^2$	$-2f$	$+$
Generalised tanh		$1-\lvert f\rvert^r$	$-r \lvert f\rvert^{r-1} \operatorname{sgn}(f)$	$\begin{cases} +, & r<2 \\ -, & r>2 \end{cases}$
		$e^{-u^2/2}\cosh u$	$-u+\tanh u$	$-$
Error function	$\operatorname{erf} u$	$\frac{2}{\sqrt{\pi}} e^{-u^2}$	$-2u$	0
Generalised erf		$e^{-\lvert u\rvert^r}$	$-r \lvert u\rvert^{r-1} \operatorname{sgn}(u)$	$\begin{cases} +, & r<2 \\ -, & r>2 \end{cases}$
Arctangent	$\arctan u$	$\frac{1}{1+u^2}$	$-\frac{2u}{1+u^2}$	$+\infty$
cdf of Student's t		$\frac{\Gamma(\frac{1}{2}(r+1))}{\sqrt{\pi r}\,\Gamma(\frac{1}{2}r)} \left(1+\frac{u^2}{r}\right)^{-\frac{r+1}{2}}$	$-\frac{(r+1)u}{u^2+r}$	$\frac{6}{r-4}$

Table 2.3: Different nonlinearities f and their associated slopes f' and functions g for the infomax rule (2.31). The kurtosis is that of the associated p.d.f. in latent space, $p(u) \propto \lvert f'(u)\rvert$. If $f(u)$ is omitted, it is defined as $\int_{-\infty}^{u} f'(v)\,dv$. $f = \operatorname{erf}$ gives the normal distribution and $f = \arctan$ the Cauchy one. In all cases $p, r > 0$.

supergaussian (positive kurtosis) source distributions and thus are not suitable for subgaussian ones. A better nonlinearity in those cases is the one given by $g(u) = -u + \tanh u$. However, Bell and Sejnowski claim that most real-world analog signals are supergaussian. Subgaussian source separation is a topic of current research (e.g. Lee et al., 1999).

The rule (2.31) is nonlocal, requiring a matrix inversion, and thus does not seem biologically plausible. Being a standard gradient descent, it is also noncovariant: different variables have different units. However, rescaling the gradient by a metric matrix $\mathbf{A}^T\mathbf{A}$, we obtain a **natural gradient** algorithm (Yang and Amari, 1997; Amari and Cichoki, 1998):

$$\Delta\mathbf{A} \propto (\mathbf{I} + \mathbf{g}(\mathbf{u})\mathbf{u}^T)\mathbf{A} \tag{2.32}$$

which is simpler and converges faster. Batch algorithms are also much faster, such as FastICA (Hyvärinen, 1999a), which is based on a fixed-point iteration.

The rule (2.31), or approximate versions of it, can be justified from several points of view:

Information maximisation In the low noise case, the *infomax principle* states that if the mutual information between the inputs $\mathbf{t}$ and the outputs $\mathbf{y}$ of a processor is maximum, then the output distribution $p(\mathbf{y})$ is factorial (Linsker, 1989; Nadal and Parga, 1994). Consider a single-layer feedforward neural network with inputs $\mathbf{t} = (t_1, \dots, t_D)^T$, net inputs $\mathbf{u} \stackrel{\text{def}}{=} \mathbf{At}$ and outputs $y_l \stackrel{\text{def}}{=} f(u_l)$, $l = 1, \dots, L$, where $\mathbf{A}_{L\times D}$ is a matrix of adjustable parameters and f a fixed invertible (thus monotonic) and differentiable nonlinearity. Maximising the mutual information $I(\mathbf{y}, \mathbf{t})$ by varying the parameters $\mathbf{A}$ is equivalent to maximising the output entropy $h(\mathbf{y})$:

$$I(\mathbf{y}, \mathbf{t}) \stackrel{\text{def}}{=} h(\mathbf{y}) - h(\mathbf{y}|\mathbf{t}) \Rightarrow \frac{\partial}{\partial\mathbf{A}} I(\mathbf{y}, \mathbf{t}) = \frac{\partial}{\partial\mathbf{A}} h(\mathbf{y})$$

because $h(\mathbf{y}|\mathbf{t})$ does not depend on $\mathbf{A}$. Taking into account that $p_{\mathbf{y}}(\mathbf{y}) = p_{\mathbf{t}}(\mathbf{t}) \lvert J(\mathbf{t}, \mathbf{y})\rvert^{-1}$ (where $J(\mathbf{t}, \mathbf{y}) = \left(\frac{\partial y_l}{\partial t_d}\right)$ is the Jacobian of the transformation), $p(u_l) = p(y_l) \lvert f'(u_l)\rvert$ and $h(\mathbf{y}) \stackrel{\text{def}}{=} -\operatorname{E}_{p_{\mathbf{y}}} \{\ln p_{\mathbf{y}}\}$, we obtain the rule (2.31) with $g(u) \stackrel{\text{def}}{=} \frac{\partial \ln \lvert f'(u)\rvert}{\partial u}$.

The maximum entropy is obtained when each y_l is distributed uniformly (assuming $\{y_l\}_{l=1}^{L}$ are amplitude-bounded random variables), in which case $p(u_l) = p(y_l) \lvert f'(u_l)\rvert \propto \lvert f'(u_l)\rvert$. That is, f is the c.d.f. of u_l.

Finally, observe that the output $\mathbf{y}$ of our single-layer neural network is used only for minimising the mutual information. We are really interested in $\mathbf{u}$, which are the recovered sources.

Maximum likelihood estimation MacKay (1996) shows for the case $L = D$ that the infomax algorithm can be derived by maximising the log-likelihood of the ICA latent variable model by gradient ascent.

Let us compute the probability distribution induced in the observed space for the particular case $L = D$ by marginalising the joint distribution as in eq. (2.3):

$$p(\mathbf{t}) = \int_{\mathcal{X}} p(\mathbf{t}|\mathbf{x})p(\mathbf{x})\,d\mathbf{x} = \int_{\mathcal{X}} \delta(\mathbf{t} - \mathbf{\Lambda}\mathbf{x})p(\mathbf{x})\,d\mathbf{x} = |\mathbf{\Lambda}|^{-1}\,p(\mathbf{\Lambda}^{-1}\mathbf{t}) = |\mathbf{\Lambda}|^{-1}\prod_{l=1}^{L} p_l\left(\sum_{d=1}^{D}\lambda_{ld}^{-1}t_d\right).$$

So we get:

$$\begin{aligned} \text{as a function of } \mathbf{\Lambda}\text{:}\ \ln p(\mathbf{t}|\mathbf{\Lambda}) &= -\ln|\mathbf{\Lambda}| + \sum_{l=1}^{L}\ln p_l\left(\sum_{d=1}^{D}\lambda_{ld}^{-1}t_d\right) \\ \text{as a function of } \mathbf{A} = \mathbf{\Lambda}^{-1}\text{:}\ \ln p(\mathbf{t}|\mathbf{A}) &= \ln|\mathbf{A}| + \sum_{l=1}^{L}\ln p_l\left(\sum_{d=1}^{D}a_{ld}t_d\right). \end{aligned}$$

Calling $\mathbf{u} \stackrel{\text{def}}{=} \mathbf{A}\mathbf{t}$ and $\mathbf{g}(\mathbf{u}) \stackrel{\text{def}}{=} \left(\frac{d\ln p_1(u_1)}{du_1}, \dots, \frac{d\ln p_L(u_L)}{du_L}\right)^T$ and using the matrix differentiation identities (A.4) we obtain the gradient of the log-likelihood:

$$\begin{aligned} \frac{\partial}{\partial\lambda_{dl}}\ln p(\mathbf{t}|\mathbf{\Lambda}) &= -a_{ld} - u_l\sum_{l'=1}^{L}a_{l'd}g_{l'}(u_{l'}) \\ \frac{\partial}{\partial a_{ld}}\ln p(\mathbf{t}|\mathbf{\Lambda}) &= \lambda_{dl} + g_l(u_l)t_d \end{aligned}$$

which coincides with the infomax one, eq. (2.31).

An alternative derivation from a maximum likelihood estimation point of view has been given by Pearlmutter and Parra (1996) and Cardoso (1997). Consider a parametric model $p_{\mathbf{\Theta}}(\mathbf{t})$ for the true sensor distribution $p_{\mathbf{t}}(\mathbf{t})$. Eq. (2.11) shows that maximising the log-likelihood of $\mathbf{\Theta}$ is equivalent to minimising the Kullback-Leibler distance to the data density $p_{\mathbf{t}}(\mathbf{t})$. Assuming $\mathbf{A}$ and $\mathbf{\Lambda}$ invertible, and (a) since $H(p_{\mathbf{t}})$ is independent of $\mathbf{A}$ and (b) the Kullback-Leibler divergence is invariant under invertible transformations ($\mathbf{t} = \mathbf{\Lambda}\mathbf{x}$ and $\mathbf{u} = \mathbf{A}\mathbf{t}$), maximum likelihood estimation produces:

$$\Delta\mathbf{A} \propto \frac{\partial\mathcal{L}(\mathbf{\Theta})}{\partial\mathbf{A}} \stackrel{\text{(a)}}{=} -\frac{\partial}{\partial\mathbf{A}}\,\mathrm{D}\left(p_{\mathbf{t}}(\mathbf{t})\|p_{\mathbf{\Theta}}(\mathbf{t})\right) \stackrel{\text{(b)}}{=} -\frac{\partial}{\partial\mathbf{A}}\,\mathrm{D}\left(p(\mathbf{x})\|p_{\mathbf{\Theta}}(\mathbf{u})\right)$$

which coincides with the infomax rule of eq. (2.31).

Negentropy maximisation and projection pursuit From a projection pursuit point of view (section 4.6), ICA looks for linear projections where the data become independent. Girolami et al. (1998) show that negentropy maximisation projection pursuit performs ICA on sub- or supergaussian sources and leads exactly to the infomax rule (2.31). Negentropy of a distribution of density p is defined as the Kullback-Leibler divergence between p and the Gaussian distribution $p_{\mathcal{N}}$ with the same mean and covariance as p: $\mathrm{D}\left(p\|p_{\mathcal{N}}\right)$. It is zero if p is normal and positive otherwise.

Cumulant expansion If $\mathbf{u}$ and $\mathbf{x}$ are symmetrically distributed and approximately normal, their mutual information can be approximated using a Gram-Charlier or Edgeworth polynomial expansion (Kendall and Stuart, 1977) of fourth order in terms of the cumulants (Comon, 1994) and used as objective function for gradient ascent. However, this method requires more computations than the infomax algorithm and gives worse separation—an order higher than 4 is necessary to improve separation.

Nonlinear PCA Karhunen and Joutsensalo (1994) show that a neural network trained by least squares to compute the function $\mathbf{A}f(\mathbf{A}\mathbf{t})$ on prewhitened data can separate signals. Girolami and Fyfe (1999) show that its cost function is approximately equivalent to that of the cumulants method.

2.6.4 Independent factor analysis (IFA)

Independent factor analysis (Attias, 1998, 1999) uses a factorial Gaussian mixture prior, a linear mapping from data space to latent space and a normal noise model[30]. Specifically:

[30] Similar approaches have also been proposed that use mixture models for the source density: Pearlmutter and Parra (1996) (mixture of logistic densities, gradient descent) and Moulines et al. (1997) and Xu et al. (1998) (Gaussian mixture, EM algorithm).

- Each latent variable is modelled independently of the others as a mixture of M_l one-dimensional Gaussians:

$$p(\mathbf{x}) \stackrel{\text{def}}{=} \prod_{l=1}^{L} p_l(x_l) \qquad p_l(x_l) \stackrel{\text{def}}{=} \sum_{m_l=1}^{M_l} p(m_l)p(x_l|m_l) \qquad x_l|m_l \stackrel{\text{def}}{\sim} \mathcal{N}(\mu_{l,m_l}, v_{l,m_l}). \tag{2.33}$$

  Thus the latent space prior $p(\mathbf{x})$ is a product of one-dimensional Gaussian mixtures, which results in a mixture of L-dimensional diagonal Gaussians but with constrained means and covariance matrices (since a general mixture of diagonal Gaussians need not be factorised):

$$p(\mathbf{x}) = \sum_{\mathbf{m}} p(\mathbf{m})p(\mathbf{x}|\mathbf{m}) \begin{cases} \mathbf{m} \stackrel{\text{def}}{=} (m_1, \dots, m_L) \\ p(\mathbf{m}) = \prod_{l=1}^{L} p(m_l) \\ \mathbf{x}|\mathbf{m} \stackrel{\text{def}}{\sim} \mathcal{N}(\boldsymbol{\mu}_{\mathbf{m}}, \mathbf{V}_{\mathbf{m}}) \text{ with } \begin{cases} \boldsymbol{\mu}_{\mathbf{m}} \stackrel{\text{def}}{=} (\mu_{1,m_1}, \dots, \mu_{L,m_L})^T \\ \mathbf{V}_{\mathbf{m}} \stackrel{\text{def}}{=} \operatorname{diag}(v_{1,m_1}, \dots, v_{L,m_L}). \end{cases} \end{cases} \tag{2.34}$$

  The summation over $\mathbf{m}$ includes all combinations of tuples $\mathbf{m} = (m_1, \dots, m_L)$ where $m_l = 1, \dots, M_l$ for each $l = 1, \dots, L$.

- The mapping $\mathbf{f}$ is linear (the data space distribution is assumed zero-mean, which can be obtained by centring the data sample before fitting the model):

$$\mathbf{f}(\mathbf{x}) \stackrel{\text{def}}{=} \boldsymbol{\Lambda}\mathbf{x}. \tag{2.35}$$

  We assume $\operatorname{rank}(\boldsymbol{\Lambda}) = L$.

- The data space noise model is normal centred at $\mathbf{f}(\mathbf{x})$ with covariance matrix $\boldsymbol{\Phi}$:

$$\mathbf{t}|\mathbf{x} \stackrel{\text{def}}{\sim} \mathcal{N}(\mathbf{f}(\mathbf{x}), \boldsymbol{\Phi}). \tag{2.36}$$

Thus, factor analysis (and PCA) is a particular case of IFA where $\boldsymbol{\Phi}$ is diagonal and each $p_l(x_l)$ is $\mathcal{N}(0, 1)$.

The marginal distribution in data space can be computed analytically and turns out to be a constrained mixture of Gaussians (eq. (2.6) and theorems 2.12.1 and A.3.1(iv)):

$$p(\mathbf{t}) = \sum_{\mathbf{m}} p(\mathbf{m})p(\mathbf{t}|\mathbf{m}) \qquad \mathbf{t}|\mathbf{m} \stackrel{\text{def}}{\sim} \mathcal{N}(\boldsymbol{\Lambda}\boldsymbol{\mu}_{\mathbf{m}}, \boldsymbol{\Lambda}\mathbf{V}_{\mathbf{m}}\boldsymbol{\Lambda}^T + \boldsymbol{\Phi}). \tag{2.37}$$

The posterior in latent space is also a mixture of Gaussians:

$$p(\mathbf{x}|\mathbf{t}) = \sum_{\mathbf{m}} p(\mathbf{m}|\mathbf{t})p(\mathbf{x}|\mathbf{m},\mathbf{t}) \begin{cases} p(\mathbf{m}|\mathbf{t}) = \frac{p(\mathbf{t}|\mathbf{m})p(\mathbf{m})}{\sum_{\mathbf{m}'} p(\mathbf{t}|\mathbf{m}')p(\mathbf{m}')} \\ (\mathbf{x}|\mathbf{m},\mathbf{t}) \stackrel{\text{def}}{\sim} \mathcal{N}(\nu_{\mathbf{m},\mathbf{t}}, \boldsymbol{\Sigma}_{\mathbf{m}}) \begin{cases} \boldsymbol{\Sigma}_{\mathbf{m}} \stackrel{\text{def}}{=} (\boldsymbol{\Lambda}^T\boldsymbol{\Phi}^{-1}\boldsymbol{\Lambda} + \mathbf{V}_{\mathbf{m}}^{-1})^{-1} \\ \nu_{\mathbf{m},\mathbf{t}} \stackrel{\text{def}}{=} \boldsymbol{\Sigma}_{\mathbf{m}}(\boldsymbol{\Lambda}^T\boldsymbol{\Phi}^{-1}\mathbf{t} + \mathbf{V}_{\mathbf{m}}^{-1}\boldsymbol{\mu}_{\mathbf{m}}). \end{cases} \end{cases} \tag{2.38}$$

The reduced-dimension representative can be taken as the posterior mean, which is simple to compute (and it need be computed in each iteration of the EM algorithm):

$$\mathbf{F}(\mathbf{t}) \stackrel{\text{def}}{=} \mathrm{E}\left\{\mathbf{x}|\mathbf{t}\right\} = \sum_{\mathbf{m}} p(\mathbf{m}|\mathbf{t})\nu_{\mathbf{m},\mathbf{t}}. \tag{2.39}$$

Since the posterior (2.38) can be asymmetric or multimodal, one could take its mode instead (which could be computed iteratively with the algorithms of chapter 8). In the limit of zero noise, the reduced-dimension representative can be taken as the projection via the pseudoinverse $\boldsymbol{\Lambda}^+ = (\boldsymbol{\Lambda}^T\boldsymbol{\Lambda})^{-1}\boldsymbol{\Lambda}^T$, as in ICA.

Attias (1999) gives an EM algorithm for IFA. Taking the limit $\boldsymbol{\Phi} \to \mathbf{0}$ in it results in an EM algorithm for PCA, as presented by Tipping and Bishop (1999b) and Roweis (1998). Attias (1999) also gives an EM and a generalised EM algorithm for noiseless IFA ($\boldsymbol{\Phi} = \mathbf{0}$). The latter results in a combination of a rule similar to the infomax rule (2.31) for updating the mixing matrix and update rules for the Gaussian mixture parameters similar to those of the standard EM algorithm for a Gaussian mixture. That is, it combines separating the sources $x_1, \dots, x_L$ with learning their densities.

Again, a major disadvantage of the IFA model is that the number of parameters grows exponentially with the dimensionality of the latent space: the prior distribution for the latent variables includes $\prod_{l=1}^{L}(2M_l - 1)$ parameters, which is $\mathcal{O}(e^L)$. Attias (1999) proposes a variational approximation of the EM algorithm that reduces the number of operations—but the number of parameters remains $\mathcal{O}(e^L)$.

2.6.5 The generative topographic mapping (GTM)

The generative topographic mapping (GTM) (Bishop, Svensén, and Williams, 1998b) is a nonlinear latent variable model which has been proposed as a principled alternative to self-organising feature maps (Kohonen, 1995). Specifically:

- The L-dimensional latent space $\mathcal{X}$ is discrete[31]. The prior in latent space, $p(\mathbf{x})$, is discrete uniform, assigning nonzero probability only to the points $\{\mathbf{x}_k\}_{k=1}^K \subset \mathbb{R}^L$, usually arranged in a regular grid (for visualisation purposes):
$$p(\mathbf{x}) \stackrel{\text{def}}{=} \frac{1}{K}\sum_{k=1}^{K} \delta(\mathbf{x}-\mathbf{x}_k). \tag{2.40}$$
This discrete prior can be seen as a fixed approximation of a continuous, uniform distribution in a hyperrectagle of $\mathbb{R}^L$ (see section 2.4 on Monte Carlo sampling).

- The mapping $\mathbf{f}$ is a generalised linear model:
$$\mathbf{f}(\mathbf{x}) \stackrel{\text{def}}{=} \mathbf{W}\boldsymbol{\phi}(\mathbf{x}), \tag{2.41}$$
where $\mathbf{W}$ is a $D \times F$ matrix and $\boldsymbol{\phi}$ an $F \times 1$ vector of fixed basis functions.

- The noise model $p(\mathbf{t}|\mathbf{x})$ is an isotropic normal centred at $\mathbf{f}(\mathbf{x})$:
$$\mathbf{t}|\mathbf{x} \stackrel{\text{def}}{\sim} \mathcal{N}(\mathbf{f}(\mathbf{x}), \sigma^2\mathbf{I}). \tag{2.42}$$

The marginal distribution in data space is a constrained mixture of Gaussians (in the sense that the Gaussian centres cannot move independently, but only by changing the mapping $\mathbf{f}$ through $\mathbf{W}$):
$$p(\mathbf{t}) = \frac{1}{K}\sum_{k=1}^{K} p(\mathbf{t}|\mathbf{x}_k), \tag{2.43}$$
and the posterior in latent space is discrete:
$$p(\mathbf{x}_k|\mathbf{t}) = \frac{p(\mathbf{t}|\mathbf{x}_k)}{\sum_{i=1}^{K} p(\mathbf{t}|\mathbf{x}_i)}. \tag{2.44}$$
The reduced-dimension representative can be taken as the posterior mean or the posterior mode, both of which can be easily computed since the posterior distribution is discrete. The mean and the mode can be quite different from each other if the posterior distribution is multimodal. The dimensionality reduction mapping $\mathbf{F}$ for the posterior mode is not continuous in general, although it will be approximately continuous if the posterior distribution (2.44) is unimodal and sharply peaked for most points in data space (section 2.9.2).

The log-likelihood of the parameters $\boldsymbol{\Theta} = \{\mathbf{W}, \sigma^2\}$ is
$$\mathcal{L}(\mathbf{W},\sigma^2) = \sum_{n=1}^{N} \ln \frac{1}{K}\sum_{k=1}^{K} p(\mathbf{t}_n|\mathbf{x}_k;\mathbf{W},\sigma^2) = \sum_{n=1}^{N} \ln \frac{1}{K(2\pi\sigma^2)^{\frac{D}{2}}} \sum_{k=1}^{K} e^{-\frac{1}{2\sigma^2}\|\mathbf{t}_n - \mathbf{W}\boldsymbol{\phi}(\mathbf{x}_k)\|^2}$$
and is known to contain a number of suboptimal maxima (see fig. 2.13 for some examples).

The parameters of a GTM model may be estimated using the EM algorithm:

E step This requires computing the *responsibility* $R_{nk} = p(\mathbf{x}_k|\mathbf{t}_n)$ of each latent space point $\mathbf{x}_k$ having generated point $\mathbf{t}_n$ using eqs. (2.40)-(2.44) with the current parameter values $\mathbf{W}^{(\tau)}$ and $\sigma^{(\tau)}$.

M step This results in the following update equations for the parameters $\mathbf{W}$ and σ, respectively:
$$\boldsymbol{\Phi}^T\mathbf{G}^{(\tau)}\boldsymbol{\Phi}(\mathbf{W}^{(\tau+1)})^T = \boldsymbol{\Phi}^T(\mathbf{R}^{(\tau)})^T\mathbf{T} \tag{2.45a}$$
$$(\sigma^{(\tau+1)})^2 = \frac{1}{ND}\sum_{k=1}^{K}\sum_{n=1}^{N} R_{nk}^{(\tau)} \|\mathbf{t}_n - \mathbf{f}(\mathbf{x}_k)\|^2 \tag{2.45b}$$
where $\boldsymbol{\Phi} \stackrel{\text{def}}{=} (\boldsymbol{\phi}_1,\dots,\boldsymbol{\phi}_K)^T$, $\mathbf{T} \stackrel{\text{def}}{=} (\mathbf{t}_1,\dots,\mathbf{t}_N)^T$, $\mathbf{R}$ is an $N \times K$ matrix with elements R_{nk} and $\mathbf{G}$ is a $K \times K$ diagonal matrix with elements $g_{kk} = \sum_{n=1}^{N} R_{nk}$. Solving for $\mathbf{W}$ requires (pseudo-)inverting the matrix $\boldsymbol{\Phi}^T\mathbf{G}\boldsymbol{\Phi}$ at each iteration.

[31] Strictly, the latent space is continuous and $p(\mathbf{x})$ is a density, but we will call it "discrete uniform" for short (see also sections 2.4 and 2.9.2).

A simple regularised version of GTM to control the mapping $\mathbf{f}$ is obtained by placing an isotropic Gaussian prior distribution of variance λ^{-1} on the mapping weights $\mathbf{W} = (w_{df})$, $\mathbf{W}|\lambda \stackrel{\text{def}}{\sim} \mathcal{N}(\mathbf{0}, \lambda^{-1}\mathbf{I})$. This leads to a maximum a posteriori (MAP) estimator:

$$\mathcal{L}_{\text{MAP}}(\mathbf{W}, \sigma^2) = \mathcal{L}(\mathbf{W}, \sigma^2) - \frac{N\lambda}{2} \|\mathbf{W}\|^2 \tag{2.46}$$

where λ is the regularisation coefficient, $\|\mathbf{W}\|^2 = \sum_{d=1}^{D} \sum_{f=1}^{F} w_{df}^2$ and an additive term independent of $\mathbf{W}$ and σ^2 has been omitted. This results in a modified M step of the EM algorithm:

$$(\mathbf{\Phi}^T \mathbf{G}^{(\tau)} \mathbf{\Phi} + \lambda\sigma^2 \mathbf{I})(\mathbf{W}^{(\tau+1)})^T = \mathbf{\Phi}^T (\mathbf{R}^{(\tau)})^T \mathbf{T}.$$

An approximate Bayesian treatment of the hyperparameter λ is given by Bishop et al. (1998a) and Utsugi (2000).

The major shortcoming of GTM is that, being based on (fixed) Monte Carlo sampling of the latent space mentioned in section 2.4, both the number of latent grid points K and the number of basis functions F required for the generalised linear model (2.41) grow exponentially with the dimension of the latent space, L. This limits the practical applicability of GTM to about 2 latent variables. Another shortcoming, shared by most global methods (i.e., that find an estimate for the whole data space) that are very flexible parameterised models, is the fact that the EM estimate very often converges to very bad local maxima, as those shown in the lower half of fig. 2.13; for twisted manifolds, a good maximum can only be reached it starting EM from a small region of the parameter space.

The **density networks** of MacKay (1995a) are a model similar to GTM: the mapping $\mathbf{f}$ is implemented by a multilayer perceptron, all the distributions are assumed isotropic Gaussian and a Monte Carlo fixed sampling is necessary to obtain $p(\mathbf{t})$. The log-likelihood is maximised with a conjugate gradients method. Like GTM, this approach suffers from the exponential complexity of the Monte Carlo sampling. MacKay applies density networks to a discrete problem, modelling a protein family. The insight of (Bishop et al., 1998b) is to implement the mapping $\mathbf{f}$ with a generalised linear model, which results in a tractable M step of the EM algorithm, since only the weights of the linear layer have to be estimated.

GTM pursues a similar goal as Kohonen's self-organising maps (to adapt a topographical arrangement of knots to a data distribution) but with the important advantage that it defines a probabilistic model for the data and dimensionality reduction and reconstruction mappings. Table 4.4 compares succintly both models.

2.6.5.1 Extensions to GTM

Bishop et al. (1998a) propose several extensions to the GTM model:

- A manifold-aligned noise model, where the covariance matrix of the noise model (2.42) is not isotropic anymore but approximately aligned (depending on the value of η below) with the manifold $\mathcal{M} = \mathbf{f}(\mathcal{X})$ locally at each mapped point $\mathbf{f}(\mathbf{x})$:

  $$\mathbf{t}|\mathbf{x} \stackrel{\text{def}}{\sim} \mathcal{N}(\mathbf{f}(\mathbf{x}), \mathbf{\Sigma}(\mathbf{x})) \qquad \mathbf{\Sigma}(\mathbf{x}) \stackrel{\text{def}}{=} \sigma^2\mathbf{I} + \eta \sum_{l=1}^{L} \frac{\partial \mathbf{f}}{\partial x_l} \left(\frac{\partial \mathbf{f}}{\partial x_l} \right)^T$$

  where the new hyperparameter η must be adjusted manually and $\mathbf{\Sigma}(\mathbf{x})$ is computed only at the points $\{\mathbf{x}_k\}_{k=1}^{K}$. The goal is to ensure that the variance of the noise distribution in directions tangential to the manifold is never significantly less than the square of the typical distance between neighbouring points $\mathbf{f}(\mathbf{x}_k)$, so that there is a smooth distribution along the manifold even when the noise variance perpendicular to the manifold becomes small.

  It would seem that this model violates the axiom of local independence (section 2.3.1), since $\mathbf{\Sigma}(\mathbf{x})$ is not diagonal and so the noise model would not be factorised anymore. However, the model can be derived from taking the model with factorised axis-aligned noise, but approximating the uniform density by a grid of Gaussians (rather than deltas) and then linearising the manifold about the centre of each Gaussian (Chris Williams, pers. comm.).

  It remains to be seen whether this approach has any advantages over using a mixture of linear-normal latent variable models (since each GTM mixture component represents now a local linear subspace).

- Modelling of discrete observed variables by defining a Bernoulli or multinomial noise model depending on $\mathbf{f}(\mathbf{x})$ via a logistic or softmax function, respectively. However, estimation of the parameters via the EM algorithm is much more difficult, requiring nonlinear optimisation in the M step.

- A semilinear model where some of the latent variables are discretised and mapped with the generalised linear model (2.41) as in the standard GTM model and the rest are continuous and mapped linearly.
- An incremental EM algorithm based on the approach of Neal and Hinton (1998).
- Approximate Bayesian inference for the regularisation parameter λ of eq. (2.46), using the Laplace approximation (MacKay, 1992a), i.e., a local Gaussian approximation at the mode of the posterior distribution of the parameters $\mathbf{W}$. The Laplace approximation breaks down when the posterior parameter distribution is multimodal or skewed—a likely situation when training data is scarce (Richardson and Green, 1997). Utsugi (2000) uses a Gibbs sampler and an ensemble learning method to approximate Bayesian inference.
- A Gaussian process formulation.

Further extensions by other authors include:

- Marrs and Webb (1999) propose an average generalised unit-speed constraint on $\mathbf{f}$ to preserve geodetic distances in data space (outlined in section 2.8.3).
- We propose a diagonal noise GTM model (dGTM), where the noise model covariance is diagonal rather than spherical: $\mathbf{t}|\mathbf{x} \sim \mathcal{N}(\mathbf{f}(\mathbf{x}), \mathbf{\Psi})$ with $\mathbf{\Psi} = \operatorname{diag}(\psi_1, \dots, \psi_D)$. In section 2.12.4 we show that the EM algorithm remains the same except for eq. (2.45b), which becomes D equations:

$$\psi_d^{(\tau+1)} = \frac{1}{N} \sum_{k=1}^{K} \sum_{n=1}^{N} R_{nk}^{(\tau)} (t_{nd} - f_d(\mathbf{x}_k))^2 \quad d = 1, \dots, D \tag{2.45b'}$$

  and the responsibilities R_{nk} depend on $\mathbf{\Psi}$ rather than on σ^2.

  A diagonal noise model is necessary to account for different scales and noise levels in the different data variables $t_1, \dots, t_D$ (which in general cannot be overcome by presphering the data, as fig. 2.8 shows).

2.7 Finite mixtures of latent variable models

Finite mixtures (Everitt and Hand, 1981) of latent variable models can be constructed in the usual way as[32]

$$p(\mathbf{t}) \stackrel{\text{def}}{=} \sum_{m=1}^{M} p(m) p(\mathbf{t}|m) \tag{2.47}$$

where:

- $p(\mathbf{t}|m)$, $m = 1, \dots, M$ are latent variable models based on latent spaces $\mathcal{X}_m$ of dimension L_m (not necessarily equal), i.e.,

$$p(\mathbf{t}|m) = \int_{\mathcal{X}_m} p(\mathbf{t}, \mathbf{x}|m)\, d\mathbf{x} = \int_{\mathcal{X}_m} p(\mathbf{t}|\mathbf{x}, m) p(\mathbf{x}|m)\, d\mathbf{x}$$

  where $p(\mathbf{x}|m)$ is the prior distribution in the latent space of the mth component, $p(\mathbf{t}|\mathbf{x}, m)$ its noise model and $\mathbf{f}_m : \mathcal{X}_m \to \mathcal{T}$ its mapping from latent space into data space.
- $p(m)$ are the mixing proportions.

The joint density is $p(\mathbf{t}, \mathbf{x}, m) = p(\mathbf{t}|\mathbf{x}, m) p(\mathbf{x}|m) p(m)$ and the finite mixture distribution can be expressed as the marginalisation of $p(\mathbf{t}, \mathbf{x}, m)$ over $\mathbf{x}$ and m:

$$p(\mathbf{t}) = \sum_{m=1}^{M} \int_{\mathcal{X}_m} p(\mathbf{t}, \mathbf{x}, m)\, d\mathbf{x} = \sum_{m=1}^{M} p(m) \int_{\mathcal{X}_m} p(\mathbf{t}|\mathbf{x}, m) p(\mathbf{x}|m)\, d\mathbf{x} = \sum_{m=1}^{M} p(m) p(\mathbf{t}|m).$$

The advantage of finite mixtures of latent variable models is that they can place different latent variable models in different regions of data space, where each latent variable model models locally the data. This allows the use of simple local models (e.g. linear-normal, like factor analysis or principal component analysis) that build a complex global model (piecewise linear-normal). In other words, finite mixtures of latent variable models combine clustering with dimensionality reduction.

[32]The IFA model results in a data space distribution (2.37) with the same form as eq. (2.47). But in the IFA model the mixture takes place in the latent space while here it takes place in the data space.

2.7.1 Parameter estimation

Once the model is formulated and the functional forms of each latent variable model are fixed, estimation of the parameters can be done by maximum likelihood. As usual with mixture models, the mixing proportions are taken as parameters $p(m) = \pi_m$ and included in the estimation process; one parameter π_m is not free due to the constraint $\sum_{m=1}^{M} \pi_m = 1$.

Maximum likelihood estimation can be conveniently accomplished with an EM algorithm, where for each data point $\mathbf{t}_n$ the missing information is not only the values of the latent variables $\mathbf{x}_n$ (as in section 2.5), but also the index of the mixture component that generated $\mathbf{t}_n$:

E step This requires computation of the *responsibility* $R_{nm} = p(m|\mathbf{t}_n)$ of each component m having generated point $\mathbf{t}_n$ using Bayes' theorem:

$$R_{nm} = \frac{p(\mathbf{t}_n|m)p(m)}{p(\mathbf{t}_n)} = \frac{\pi_m p(\mathbf{t}_n|m)}{\sum_{m=1}^{M} \pi_m p(\mathbf{t}_n|m)}$$

where $p(\mathbf{t}|m)$ is given by the latent variable model with the parameter values of the current iteration.

M step This results in several update equations for the parameters. The update equations for the mixing proportions are independent of the type of latent variable model used:

$$\pi_m^{(\tau+1)} = \frac{1}{N} \sum_{n=1}^{N} R_{nm}^{(\tau)} \pi_m^{(\tau)}.$$

The equations for the rest of the parameters (from the individual latent variable models) depend on the specific functional form of $p(\mathbf{t}|\mathbf{x}, m)$ and $p(\mathbf{x}|m)$, but often they are averages of the usual statistics weighted by the responsibilities and computed in a specific order.

2.7.2 Examples

Ghahramani and Hinton (1996) construct a **mixture of factor analysers** where each factor analyser, of L_m factors, is characterised by two kinds of parameters: the mean vector $\boldsymbol{\mu}_m \in \mathbb{R}^{L_m}$ and the loadings matrix $\boldsymbol{\Lambda}_m$ (containing L_m loading vectors), in addition to the mixing proportion π_m. All analysers share a common noise model diagonal covariance matrix $\boldsymbol{\Psi}$ for simplicity (although this implies a loss of generality). They give an EM algorithm for estimating the parameters $\{\{\pi_m, \boldsymbol{\mu}_m, \boldsymbol{\Lambda}_m\}_{m=1}^{M}, \boldsymbol{\Psi}\}$ by maximum likelihood from a sample. As with many other mixture models, the log-likelihood surface contains singularities (where the covariance matrix of a component model tends to zero and the lilkelihood tends to infinity) to which the EM algorithm can be attracted (see sections 3.2.2 and 5.4.4.1). Hinton et al. (1997) also used a mixture of factor analysers to model handwritten characters, where each factor analyser was implemented via an autoencoder. McLachlan and Peel (2000, chapter 8) give an AECM algorithm[33] for a mixture of factor analysers where each component has a different, diagonal covariance matrix $\boldsymbol{\Psi}_m$. Utsugi and Kumagai (2001) derive Bayesian estimation algorithms using a Gibbs sampler and its deterministic approximation for this same model. Ghahramani and Beal (2000) apply a variational approximation of Bayesian inference to the model of Ghahramani and Hinton (1996) to automatically determine the optimal number of components M and the local dimensionality of each component L_m.

Tipping and Bishop (1999a) define **mixtures of principal component analysers** and give for them an EM algorithm and a simpler and faster generalised EM algorithm. They show good results in image compression and handwritten digit recognition applications, although the reconstruction error attained is in general larger than that attained by a nonprobabilistic mixture of PCAs trained to minimise the reconstruction error (the VQPCA algorithm of Kambhatla and Leen, 1997, discussed in section 4.7). This is reasonable since the probabilistic mixture of PCAs is trained to maximise the log-likelihood rather than minimise the reconstruction error.

A mixture of diagonal Gaussians and a mixture of spherical Gaussians can be seen, as limit cases, as a mixture of factor analysers with zero factors per component model and a mixture of principal component analysers with zero principal components per component model, respectively. Thus, Gaussian mixtures explain the data by assuming that it its exclusively due to noise—without any underlying (linear) structure.

[33] The alternating expectation conditional-maximisation (AECM) algorithm (Meng and van Dyk, 1997) replaces the M-step of the EM algorithm by a number of computationally simpler conditional maximisation steps and allows the specification of the complete data to be different on each such step.

Bishop and Tipping (1998) use hierarchical mixtures of latent variable models with a two-dimensional latent space to visualise the structure of a data set at different levels of detail; for example, a first, top level can show the general cluster structure of the data while a second level can show the internal structure of the clusters, and so on. The tree structure of the hierarchy can be built iteratively by the user in a top-down way.

Section 4.7 mentions other local models for dimensionality reduction not based on mixtures of latent variable models.

2.7.3 Comparison with Gaussian mixtures

In general, mixtures of latent variable models whose distribution in data space $p(\mathbf{t})$ results in a Gaussian mixture (such as mixtures of factor analysers or PCAs) have two advantages over usual mixtures of Gaussian distributions:

- Each component latent variable model locally models both the (linear) mapping and the noise, rather than just the covariance.
- They use fewer parameters per component, e.g. $D(L+1)$ for a factor analyser versus $\frac{D(D+1)}{2}$ for a Gaussian (of course, L should not be too small for the model to remain good).

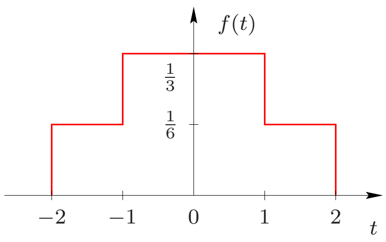
A small number of free parameters requires less computation and less training data (thus reducing the risk of overfitting when data is scarce). Using an unconstrained Gaussian mixture requires more parameters because each mixture component has its own covariance matrix. The total number of parameters can be reduced by one of the following approaches:

- Using diagonal or even spherical components (which still are universal density approximators, as full-covariance mixtures are; Titterington et al., 1985; Scott, 1992).
- Using *parameter tying* or *sharing* (Young, 1996). This method, very popular in the literature of speech recognition based on hidden Markov models, consists of allocating a pool of parameters, typically covariance matrices, to be shared by several or all the components of the mixture. Thus, several components may have the same covariance matrix but different means and mixture proportions. However, the training algorithm—often EM-based—becomes considerably complex and the task of deciding what parameters to tie is not straightforward.

Exactly what approach (full, diagonal or spherical covariance; and unconstrained, latent variable or parameter-tied mixture) is more efficient depends on each particular case; "efficient" here means to use as few parameters as possible to achieve a given performance (such as approximation error, classification error, word error rate, etc.). Saul and Rahim (2000b) give an EM algorithm for hidden Markov models whose output distribution is a mixture of factor analysers and show good performance with a reduced number of parameters when compared with other methods.

2.8 Identifiability, interpretability and visualisation

Identifiability of a class of probabilistic models refers to the existence of a unique characterisation for any model in that class, i.e., to the fact that no two different values of the model parameters give rise to the same distribution (Titterington et al., 1985; Everitt and Hand, 1981; McLachlan and Peel, 2000). For example, the class of finite mixtures of uniform distributions is not identifiable, as a simple counterexample shows. The following three mixtures produce the same distribution:



$$f_1 \stackrel{\text{def}}{\sim} \frac{1}{3}\mathcal{U}(-1,1) + \frac{2}{3}\mathcal{U}(-2,2) \tag{2.48a}$$

$$f_2 \stackrel{\text{def}}{\sim} \frac{1}{2}\mathcal{U}(-2,1) + \frac{1}{2}\mathcal{U}(-1,2) \tag{2.48b}$$

$$f_3 \stackrel{\text{def}}{\sim} \frac{1}{6}\mathcal{U}(-2,1) + \frac{2}{3}\mathcal{U}(-1,1) + \frac{1}{6}\mathcal{U}(1,2) \tag{2.48c}$$

since $f_1(t) = f_2(t) = f_3(t)$ for all $t \in (-\infty, \infty)$.

For a mixture distribution, the parameters include the number of mixture components too. Thus, identifiability is defined formally as follows for mixtures (and, as a particular case, for non-mixture models):

Definition 2.8.1. A class of finite mixtures parameterised by $\mathbf{\Theta}$ is said to be identifiable if for any two members

$$p(\mathbf{t};\mathbf{\Theta}) \stackrel{\text{def}}{=} \sum_{m=1}^{M} \pi_m p(\mathbf{t};\theta_m) \qquad p(\mathbf{t};\mathbf{\Theta}^*) \stackrel{\text{def}}{=} \sum_{m=1}^{M^*} \pi_m^* p(\mathbf{t};\theta_m^*)$$

then $p(t;\mathbf{\Theta}) = p(t;\mathbf{\Theta}^*)$ for all $\mathbf{t} \in \mathcal{T}$ if and only if $M = M^*$ and $\pi_m = \pi_m^*$ and $\theta_m = \theta_m^*$ for $m = 1,\dots,M$ (perhaps with a reordering of the indices). Trivial cases where $\pi_m = 0$ or $\theta_m = \theta_{m'}$ for some m, m' are disregarded, being exactly represented by a mixture with fewer components.

Consider the example of equations (2.48). Given a sample $\{t_n\}_{n=1}^N$ generated from f_1, we cannot tell from which of f_1, f_2 or f_3 it was generated (given the sample alone). Estimating the parameters of a uniform mixture with two components could end up (approximately) in any of f_1 or f_2 or not work at all, depending on the algorithm used, the starting point if it is iterative, etc. The sample could be interpreted as coming from two populations (in the case of f_1 and f_2) or from three (in the case of f_3). Thus, if we want (1) to be able to interpret in a unique way the parameters estimated from a sample, and (2) to avoid that the estimation procedure may break down in case of ill-posedness, then the model class being considered must be identifiable.

However, theoretical identifiability is not the whole story:

- The existence of local, suboptimal maxima of the log-likelihood (or other objective function) makes very difficult to obtain the global maximum—for example, when estimating a Gaussian mixture or a GTM model (fig. 2.13).
- Theoretical identifiability does not guarantee practical identifiability, as discussed in section 3.3.3.2.

Non-identifiability often arises with discrete distributions, for the following reason: if there are C categories we cannot set up more than $C-1$ independent equations (because $\sum p(\mathbf{c}) = 1$) and hence can only determine $C-1$ parameters at most. Fortunately, with continuous distributions—the case that concerns us—it is usually not a problem.

Identifiability is a mathematical property that depends on the choice of model and there are no general necessary and sufficient conditions for all models. The identifiability of the latent variable models of section 2.6, if known, is discussed below, while that of finite mixtures of multivariate Bernoulli distributions (which are used in chapter 5) is discussed separately in section 3.3. A further treatment of the identifiability of mixtures is given by Titterington et al. (1985).

2.8.1 Interpretability

The factor analysis literature, especially regarding social science applications, has debated the issue of the interpretability of factors for decades. The problem arises from the fact that factor analysis is non-identifiable with respect to orthogonal rotations of the factors.

As noticed in section 2.3.2, in an ideal latent variable model we could adjust freely both the latent space prior distribution $p_l(\mathbf{x})$ and the mapping $\mathbf{f}$ via an invertible transformation $\mathbf{g}$ of the latent space, so that we could have infinitely many equivalent combinations $(p_l, \mathbf{f})$, all giving rise to the same data distribution $p(\mathbf{t})$ in eq. (2.3). In other words, we could have infinitely many different coordinate systems, each one with its one interpretation, equally able to explain the data. In more practical latent variable models only some restricted kinds of indeterminacy will exist (such as orthogonal rotations in factor analysis) but the point remains: there is no **empirical** ground to prefer one model over another if both give rise to the same $p(\mathbf{t})$.

The key matter is that the only variables with a real existence are the observed variables and that the latent variables are sheer mathematical constructs subordinated to explaining the observed ones in a compact way (Bartholomew, 1987; Everitt, 1984). Therefore, one should not try to reify the latent variables[34] or look too hard for an interpretation of the estimated parameters if the model under consideration belongs to an non-identifiable class. Also, one should not attribute a causal nature to the latent variables: the generative view of section 2.3 is just a convenient conceptualisation of the probabilistic latent variable framework.

In contrast, let us mention the technique of **principal variables** (McCabe, 1984), related to principal component analysis, whose aim is to select a subset of variables that contain, in some sense, as much information as possible. The principal variables can be readily interpreted because they are observed variables—unlike the principal components, which are linear combinations of the observed variables.

[34] Although in some particular cases this may be possible. For example, Kvalheim (1992) claims that, in chemistry, latent variables are often interpretable and independently verifiable.

However, what can be interpretable is the manifold in data space spanned by the mapping from latent onto data space, $\mathcal{M} = \mathbf{f}(\mathcal{X})$. This manifold is invariant under a coordinate change, while the coordinates themselves (the latent variables) are not. As an example, consider PCA: what reason is there to choose a particular basis of the hyperplane spanned by the principal components? Besides, sample-based estimates of the principal components can vary considerably, in particular when the covariance matrix has several eigenvalues of approximately the same value, since the directions of the associated eigenvectors will be strongly influenced by the noise. But the subspace spanned by those principal components is still unique and thus identifiable and interpretable.

Another issue is that often the model assumptions are wrong for the problem under consideration, due to the nature of the relationship between variables being unknown or to the mathematical impossibility of using an appropriate but overly complex model. Consider, for example, fitting a principal component analysis to the data of figure 2.13. Interpretation of the estimated parameters in such a situation may be completely misleading. Yet another difficulty lies in whether the number of parameters in the model and the way it is estimated from a data set can lead to overfitting.

Fortunately, in machine learning applications the data is usually of a very high dimensionality and the relationships between variables is nonlinear, so that more often than not the user does not have strong preconceptions about what the latent variables should be. Besides, as a consequence of the philosophical approach of statistical machine learning, which pursues to simulate the behaviour of a system, the utility of a model is related to how well it can extract the structure of a high-dimensional data set—irrespective of whether it actually matches the true underlying system (always an idealisation anyway) or just mimicks it. Of course, overfitting remains a difficult problem.

To summarise:

- In the ideal case where we know the model from which the data comes (i.e., we know the number of latent variables and the functional forms for the prior distribution in latent space, the mapping and the noise model) and all is necessary is to fit its parameters, we need to check the issues of identifiability and overfitting.
- If we are guessing the model, apart from the identifiability and overfitting we need to assess the model's goodness, which is very difficult in high-dimensional cases when there are nonlinear relationships. For example, the noise can easily mask nonlinear relationships when using a linear model.

2.8.2 Identifiability of specific latent variable models

2.8.2.1 Factor analysis

From section 2.6.1 we know that:

- If $L > 1$ then an orthogonal rotation of the factors produces the same data distribution, so that factor analysis is non-identifiable with respect to orthogonal rotations. If $L = 1$ then the rotation becomes a sign reversal, which is irrelevant.
- If we consider a generalised view of factor analysis where the factors are distributed normally but not necessarily $\mathcal{N}(\mathbf{0}, \mathbf{I})$, then factor analysis is non-identifiable with respect to any invertible linear transformation of the factors.

In confirmatory factor analysis, where some elements of the loading matrix $\mathbf{\Lambda}$ are fixed, the rotation indeterminacy may disappear. In explanatory factor analysis, where all elements of $\mathbf{\Lambda}$ are free, one usually applies a restriction to make it identifiable. The most typical one is to choose the factors so that the first factor makes a maximum contribution to the variance of the observed variables, the second makes a maximum contribution subject to being uncorrelated with the first one, and so on. This can be shown to be equivalent to choosing $\mathbf{\Lambda}$ such that $\mathbf{\Lambda}^T \mathbf{\Psi}^{-1} \mathbf{\Lambda}$ is diagonal and has the effect of imposing $\frac{1}{2}L(L-1)$ constraints, so that the total number of free parameters becomes

$$\underbrace{D}_{\mathbf{\Psi}} + \underbrace{DL}_{\mathbf{\Lambda}} + \underbrace{\frac{1}{2}L(L-1)}_{\text{constraints}}$$

which for consistency must be smaller or equal than the number of elements in the covariance matrix, $\frac{1}{2}D(D+1)$, i.e., $\frac{1}{2}((D-L)^2 - (D+L))$ must be positive, which gives an upper bound for the number of factors to be extracted.

Since scale changes of the data ($\mathbf{t}' = \mathbf{D}\mathbf{t}$ with $\mathbf{D}$ diagonal) result in an irrelevant scale change of the factors and the uniquenesses, the same results are obtained whether using the covariance matrix or the correlation matrix.

2.8.2.2 PCA

PCA as defined in section 2.6.2 only differs from factor analysis in the noise model and is constrained to produce the principal components in decreasing order of their associated variance, i.e., $\boldsymbol{\Lambda}^T(\sigma^2\mathbf{I})^{-1}\boldsymbol{\Lambda}$ is diagonal from eq. (2.28). Thus, PCA is always identifiable except when several eigenvalues of the sample covariance matrix are equal, in which case the corresponding eigenvectors (columns of $\mathbf{U}_L$) can be rotated orthogonally inside their own subspace at will.

Unlike with factor analysis, with PCA different results are obtained whether using the covariance matrix or the correlation matrix because there is a single uniqueness parameter shared by all data variables. PCA does depend on the scale, so one has to decide whether to sphere the data or not.

2.8.2.3 ICA

Arbitrary scaling, reordering or in general invertible linear mapping of the latent variables (sources) gives rise to the same observed (sensor) distribution: $(\boldsymbol{\Lambda}, \mathbf{x})$, $(\boldsymbol{\Lambda}\mathbf{V}, \mathbf{V}^{-1}\mathbf{x})$, $(\boldsymbol{\Lambda}\mathbf{P}, \mathbf{P}^T\mathbf{x})$ and $(\boldsymbol{\Lambda}\mathbf{R}, \mathbf{R}^{-1}\mathbf{x})$ all produce the same distribution $p(\mathbf{t})$ if $\mathbf{V}$ is diagonal nonsingular, $\mathbf{P}$ is a permutation matrix and $\mathbf{R}$ is nonsingular. However, only scaling and reordering are acceptable indeterminacies, because they do not alter the "waveform of the signals" or their statistical properties (independence in particular). Thus, it can be proven that the identifiability of the ICA model (up to reordering and rescaling) is guaranteed if (Tong et al., 1991; Comon, 1994):

- At most one source is distributed normally (since the sum of two normals is itself normal, it would not possible to separate the individual components in a unique way).
- There are fewer sources than sensors: $L \leq D$.
- The mixing matrix is full-rank: $\operatorname{rank}(\boldsymbol{\Lambda}) = L$.

2.8.2.4 Other latent variable models

To our knowledge, no identifiability results are known for GTM, IFA or mixtures of factor analysers or principal component analysers—but suboptimal local maxima of the log-likelihood do exist for all of them.

2.8.3 Visualisation

A different matter from interpretability is visualisation of data. While all coordinate systems related by an invertible map (reparametrisation) are equivalent for dimensionality reduction, some may be more appropriate than others in that the structure in the data is shown more clearly. Rigid motion transformations (translation, rotation or reflection) have a trivial effect on visualisation, since our visual system can still recognise the structure in the data, but invertible transformations that alter the distances between points (distortions) can both make apparent but also completely mask clusters or other accidents in the data.

It is possible to include constraints in the latent variable model being used to ensure that the interpoint distances in data space are approximately preserved in the latent space. These constraints affect exclusively the mapping $\mathbf{f}$. However, if the prior distribution in latent space is not flexible enough to represent a large class of distributions (and it is not for factor analysis, PCA and GTM, since it is fixed) then by constraining $\mathbf{f}$ we are reducing the class of distributions that the model can approximate.

These distortion constraints are different from the usual smoothness constraints: the latter force the manifold $\mathcal{M} = \mathbf{f}(\mathcal{X})$ traced by the function $\mathbf{f}$ to be smooth but do not care about the coordinate system (the parametrisation); they usually bound (in a soft fashion) the second derivative of $\mathbf{f}$. In contrast, the distortion constraints do not care whether the manifold $\mathcal{M}$ is smooth or rough, but force the distances in data space to be preserved in latent space. How this is done depends on how the distance between two data space points is defined: as geodetic distance (i.e., following the shortest path between the two points along the data manifold, in which case it is appropriate to use a unit-speed parametrisation[35] that bounds the first derivative of $\mathbf{f}$) or as Euclidean distance (i.e., following the straight line segment joining both points freely in the data space, whether it goes out of the data manifold or not). This consideration enters the realm of distance-preserving methods and multidimensional scaling, briefly described in section 4.10.

Such distortion constraints can be incorporated as a regularisation term in the log-likelihood or, equivalently, viewed as a prior distribution on the parameters of $\mathbf{f}$. In any case they lead to the appearance of hyperparameters that control the relative importance of the fitting term (log-likelihood) and the regularisation term (distortion constraint). Determining good values for the hyperparameters is a well-known problem that we will meet several times in this thesis, but on which we shall not dwell.

For example, it has been observed that the estimates found by GTM often give a distorted view of the data due to the latent space stretching like a rubber sheet (Tenenbaum, 1998; Marrs and Webb, 1999). This metric distortion is revealed by high values of an appropriately defined magnification factor. A *magnification factor* $M(\mathbf{x})$ is a scalar function dependent on the latent space point that measures the local distortion at latent space point $\mathbf{x}$ induced by the mapping $\mathbf{f} : \mathcal{X} \rightarrow \mathcal{T}$. Bishop et al. (1997b) defined it in the usual sense of differential geometry as the ratio of hypervolumes in the data space and the latent space, equal to the Jacobian of the mapping $\mathbf{f}$: $M(\mathbf{x}; \mathbf{W}) \stackrel{\text{def}}{=} \frac{dV_{\mathcal{T}}}{dV_{\mathcal{X}}} = \mathbf{J}_{\mathbf{f}} = \sqrt{|\mathbf{K}^T \mathbf{W}^T \mathbf{W} \mathbf{K}|}$ where $(K)_{fl} \stackrel{\text{def}}{=} \frac{\partial \phi_f}{\partial x_l}$. A value of one for this magnification factor corresponds to no distortion (although Marrs and Webb (1999) have pointed out that it can also be one for some distortions; their slightly different definition of the magnification factor is given below). Marrs and Webb (1999) implement an average generalised unit-speed constraint in GTM's function $\mathbf{f}$ to preserve geodetic distances:

$$\mathrm{E}_{p(\mathbf{t})} \left\{ \mathbf{K}^T \mathbf{W}^T \mathbf{W} \mathbf{K} \right\} = \mathbf{I}_L$$

which for a one-dimensional latent space ($L = 1$) reduces to $\mathrm{E}\left\{ \left\| \frac{d\mathbf{f}}{dx} \right\|^2 \right\} = 1$, i.e., a parametrisation of average unit squared speed. This constraint results in a modified M step of the EM algorithm for GTM, eq. (2.45a). The results can be evaluated by checking that the magnification factor $M(\mathbf{x}; \mathbf{W}) \stackrel{\text{def}}{=} \left\| \mathbf{K}^T \mathbf{W}^T \mathbf{W} \mathbf{K} - \mathbf{I} \right\|^2$ (a zero value of which corresponds to no distortion) is small in all points of the latent space.

2.9 Mapping from data onto latent space

2.9.1 Dimensionality reduction and vector reconstruction

In dimensionality reduction (reviewed in chapter 4) we want, given a data point $\mathbf{t}$, to obtain a representative of it in latent space, $\mathbf{x}^* = \mathbf{F}(\mathbf{t})$, for a certain mapping $\mathbf{F} : \mathcal{T} \longrightarrow \mathcal{X}$. The latent variable modelling framework allows the definition of a natural dimensionality reduction mapping. Once the parameters[36] $\boldsymbol{\Theta}$ are fixed, Bayes' theorem gives the posterior distribution in latent space given a data vector $\mathbf{t}$, i.e., the distribution of the probability that a point $\mathbf{x}$ in latent space was responsible for generating $\mathbf{t}$:

$$p(\mathbf{x}|\mathbf{t}) = \frac{p(\mathbf{t}|\mathbf{x})p(\mathbf{x})}{p(\mathbf{t})} = \frac{p(\mathbf{t}|\mathbf{x})p(\mathbf{x})}{\int_{\mathcal{X}} p(\mathbf{t}|\mathbf{x})p(\mathbf{x})\, d\mathbf{x}}. \tag{2.49}$$

The latent variable model gives us all this information about $\mathbf{x}$ for fixed $\mathbf{t}$. Summarising this distribution $\mathbf{x}|\mathbf{t}$ in a single latent space point $\mathbf{x}^*$ results in a **reduced-dimension representative**[37] of $\mathbf{t}$. This defines a corresponding mapping $\mathbf{F}$ from data space onto latent space, so that every data point $\mathbf{t}$ is assigned a representative in latent space, $\mathbf{x}^* = \mathbf{F}(\mathbf{t})$. Thus, it can be considered as an *inverse mapping* of $\mathbf{f}$.

---

[35] For a curve $\mathbf{f} : x \in \mathbb{R} \rightarrow \mathbb{R}^D$, a unit-speed parametrisation verifies $\left\| \frac{d\mathbf{f}}{dx} \right\|^2 = \sum_{d=1}^D \left(\frac{df_d}{dx} \right)^2 = 1$. Equivalently, the arc length between two points $\mathbf{f}(x_1)$ and $\mathbf{f}(x_2)$ on the curve is

$$\left| \int_{x_1}^{x_2} \sqrt{\sum_{d=1}^D \left(\frac{df_d}{dx} \right)^2}\, dx \right| = |x_2 - x_1|$$

so that this parametrisation is also called *arc-length parametrisation*. It means that a unit step in latent space produces a unit step along the curve in data space.

[36] In what follows, we omit the parameters from the formulae for clarity, i.e., $p(\mathbf{x}|\mathbf{t})$ really means $p(\mathbf{x}|\mathbf{t}, \boldsymbol{\Theta})$, and so on.

[37] Admittedly, *reduced-dimension representative* of an observed point $\mathbf{t}$ is a pedantic denomination, but it points to the two things we are interested in: that it (1) represents the observed point (2) in latent space. *Latent space representative* is also acceptable, although less general. Other, more compact terms have been proposed but we find them unsatisfactory. For example, *cause* of an observed point $\mathbf{t}$ may attribute to the reduced-dimension representative more than it really is worth (and ring unwanted bells concerning its interpretation). And the term *scores*, very popular in statistics, sounds too vague.

When the posterior distribution $p(\mathbf{x}|\mathbf{t})$ is unimodal, defining $\mathbf{F}$ as the **posterior mean**:

$$\mathbf{F}(\mathbf{t}) \stackrel{\text{def}}{=} \mathrm{E}\left\{\mathbf{x}|\mathbf{t}\right\} = \mathrm{E}_{p(\mathbf{x}|\mathbf{t})}\left\{\mathbf{x}\right\}$$

is the obvious choice since it is optimal in the least-squares sense (see section 7.3.3). But if $p(\mathbf{x}|\mathbf{t})$ may be multimodal for some points $\mathbf{t}$ (as happens with IFA and GTM), then the posterior mean (while still being the least-squares optimum and defining a continuous mapping) may be inappropriate, since the mean of a multimodal distribution can be a low-probability point. Defining $\mathbf{F}$ as one of the **posterior modes** (perhaps the global one):

$$\mathbf{F}(\mathbf{t}) \stackrel{\text{def}}{=} \arg\max_{\mathbf{x}\in\mathcal{X}} p(\mathbf{x}|\mathbf{t})$$

ensures that the reduced-dimension representative $\mathbf{F}(\mathbf{t})$ is a high-probability point, but then which mode to choose is a problem, and besides $\mathbf{F}$ may become discontinuous. These issues are discussed at length in sections 2.9.2 and 7.3.

If $\mathbf{f}$ is injective (the manifold $\mathcal{M} = \mathbf{f}(\mathcal{X})$ does not self-intersect) then $\mathbf{f}$ is invertible[38] on $\mathcal{M}$; i.e., there exists a function $\mathbf{f}^{-1} : \mathcal{M} \to \mathcal{X}$ such that $\mathbf{f}^{-1}(\mathbf{t}) = \mathbf{x}$ if $\mathbf{t} = \mathbf{f}(\mathbf{x}) \in \mathcal{M}$ for $\mathbf{x} \in \mathcal{X}$. But we are interested in defining a dimensionality reduction mapping $\mathbf{F}$ not only for data points in $\mathcal{M}$, but in the whole data space $\mathcal{T}$, which we can attain thanks to the noise model, that assigns nonzero probability to a neighbourhood of every point in $\mathcal{M}$ (remarkably, this very essence of the probabilistic modelling implies that in general $\mathbf{F}\circ\mathbf{f} \neq$ identity, as discussed below). Since $\dim\mathcal{M} < \dim\mathcal{T}$ (for dimensionality reduction to make sense) this will mean that a whole manifold of dimension $\dim\mathcal{T} - \dim\mathcal{X} = D - L$ will be mapped onto the same latent space point $\mathbf{x}$: $\mathbf{F}^{-1}(\mathbf{x}) \stackrel{\text{def}}{=} \{\mathbf{t} \in \mathcal{T} : \mathbf{F}(\mathbf{t}) = \mathbf{x}\}$. For example, if $\mathbf{F}$ is linear with matrix $\mathbf{A}$ (assumed full-rank), $\mathbf{F}(\mathbf{t}) = \mathbf{A}(\mathbf{t}-\boldsymbol{\mu})$, then $\mathbf{F}^{-1}(\mathbf{x}) = \{\mathbf{t} \in \mathcal{T} : \mathbf{F}(\mathbf{t}) = \mathbf{A}(\mathbf{t} - \boldsymbol{\mu}) = \mathbf{x}\} = \{\boldsymbol{\mu} + \mathbf{B}\mathbf{x}\} + \ker\mathbf{A}$ where $\mathbf{B}$ is any matrix that satisfies $\mathbf{B}\mathbf{A} = \mathbf{I}$ (such as the pseudoinverse of $\mathbf{A}$, $\mathbf{A}^+ = \mathbf{A}^T(\mathbf{A}\mathbf{A}^T)^{-1}$, but there may be others, as section 2.9.1.1 discusses) and $\ker\mathbf{A} \stackrel{\text{def}}{=} \{\mathbf{t} \in \mathcal{T} : \mathbf{A}\mathbf{t} = \mathbf{0}\}$ is the kernel or null-space of the matrix $\mathbf{A}$. $\mathbf{F}^{-1}(\mathbf{x})$ has dimension $D - L$ since $\dim\ker\mathbf{F} + \dim\operatorname{im}\mathbf{F} = \dim\mathcal{T}$ and $\dim\operatorname{im}\mathbf{F} = \dim\mathcal{X}$.

Applying the mapping $\mathbf{f}$ to the reduced-dimension representative $\mathbf{x} = \mathbf{F}(\mathbf{t})$ we obtain the **reconstructed** data vector $\mathbf{t}^* = \mathbf{f}(\mathbf{x}^*)$. Then, the reconstruction error for that point $\mathbf{t}$ is defined as $d(\mathbf{t}, \mathbf{t}^*)$ (or a function of it) for some suitable distance d in data space and the average reconstruction error for the sample is defined as $E_d = \frac{1}{N}\sum_{n=1}^N d(\mathbf{t}_n, \mathbf{t}_n^*)$. For example, taking the square of the Euclidean distance, $d(\mathbf{t}, \mathbf{t}^*) = \|\mathbf{t} - \mathbf{t}^*\|_2^2$, results in the usual mean squared error criterion $E_2 = \frac{1}{N}\sum_{n=1}^N \|\mathbf{t}_n - \mathbf{t}_n^*\|_2^2$. It is unknown what the relationship between the maximum likelihood criterion and a distance criterion is. While in general they are different, leading to different estimates of the parameters, in practice maximum likelihood estimation often produces very good estimates in terms of reconstruction error.

A desirable feature of the dimension reduction mapping $\mathbf{F}$ would be to satisfy that $\mathbf{F}\circ\mathbf{f}$ be the identity, so that $\mathbf{F}(\mathbf{f}(\mathbf{x})) = \mathbf{x}$ for any latent space point. That is, that $\mathbf{F}$ be the inverse of $\mathbf{f}$ in the manifold $\mathcal{M} = \mathbf{f}(\mathcal{X}) = \operatorname{im}\mathbf{f}$ (the image, or range, space of $\mathbf{f}$). This implies perfect reconstruction of data points in $\mathcal{M}$, since if $\mathbf{t} = \mathbf{f}(\mathbf{x}) \in \mathcal{M}$, then the reconstructed point of $\mathbf{t}$ is $\mathbf{t}^* = \mathbf{f}(\mathbf{F}(\mathbf{t})) = \mathbf{f}(\mathbf{F}(\mathbf{f}(\mathbf{x}))) = \mathbf{f}(\mathbf{x}) = \mathbf{t}$. In general, this condition is not satisfied (as sections 2.9.1.1–2.9.1.3 show) except in the zero-noise limit. In the latter case, the data space points in $\mathcal{T} \setminus \mathbf{f}(\mathcal{X})$ are unreachable under the model (in the sense that $p(\mathbf{t}) = 0$ for such points) and the mapping $\mathbf{f} : \mathcal{X} \to \mathcal{M}$ is invertible on the image space of $\mathcal{X}$ by $\mathbf{F}$ (assuming that $\mathbf{f}$ is injective, i.e., $\mathcal{M} = \mathbf{f}(\mathcal{X})$ does not self-intersect).

The following sections analyse the dimensionality reduction mappings of each specific latent variable model.

2.9.1.1 Linear-normal models (factor analysis and PCA): scores matrix

Consider a general linear-normal model as in the statement of theorem 2.12.1, i.e., $\mathbf{x} \stackrel{\text{def}}{\sim} \mathcal{N}_L(\boldsymbol{\mu}_X, \boldsymbol{\Sigma}_X)$ in $\mathbb{R}^L$, $\mathbf{f}(\mathbf{x}) \stackrel{\text{def}}{=} \boldsymbol{\Lambda}\mathbf{x}+\boldsymbol{\mu}_T$ and $\mathbf{t}|\mathbf{x} \stackrel{\text{def}}{\sim} \mathcal{N}_D(\mathbf{f}(\mathbf{x}), \boldsymbol{\Sigma}_T)$ in $\mathbb{R}^D$ (remember that for factor analysis $\boldsymbol{\mu}_X = \mathbf{0}$, $\boldsymbol{\Sigma}_X = \mathbf{I}$ and $\boldsymbol{\Sigma}_T = \boldsymbol{\Psi}$ is diagonal and for PCA $\boldsymbol{\mu}_X = \mathbf{0}$, $\boldsymbol{\Sigma}_X = \mathbf{I}$ and $\boldsymbol{\Sigma}_T = \sigma^2\mathbf{I}$ is isotropic). Consider the dimensionality reduction mapping defined by the posterior mean (identical to the posterior mode), $\mathbf{F}(\mathbf{t}) \stackrel{\text{def}}{=} \mathrm{E}\left\{\mathbf{x}|\mathbf{t}\right\} = \mathbf{A}(\mathbf{t} - \boldsymbol{\mu}) + \boldsymbol{\mu}_X$. The matrix $\mathbf{A}$ of eq. (2.58) is called the **Thomson scores** in the factor analysis literature (eq. (2.18)). Theorem 2.12.3 shows that the posterior distribution is always narrower than the prior distribution, the narrower the smaller the noise is.

From eq. (2.57), the condition $\mathbf{F}\circ\mathbf{f} \equiv$ identity is equivalent to:

$$\mathbf{F}(\mathbf{f}(\mathbf{x})) = \mathbf{A}(\mathbf{f}(\mathbf{x}) - \boldsymbol{\mu}_T) + \hat{\boldsymbol{\Sigma}}_X^{-1}\boldsymbol{\Sigma}_X^{-1}\boldsymbol{\mu}_X = \mathbf{A}\boldsymbol{\Lambda}\mathbf{x} + \hat{\boldsymbol{\Sigma}}_X^{-1}\boldsymbol{\Sigma}_X^{-1}\boldsymbol{\mu}_X = \mathbf{x} \qquad \forall\mathbf{x} \in \mathbb{R}^L$$

[38]And even if $\mathbf{f}$ is not injective, local invertibility is assured by the inverse function theorem A.6.1, since $\mathbf{f}$ is smooth.

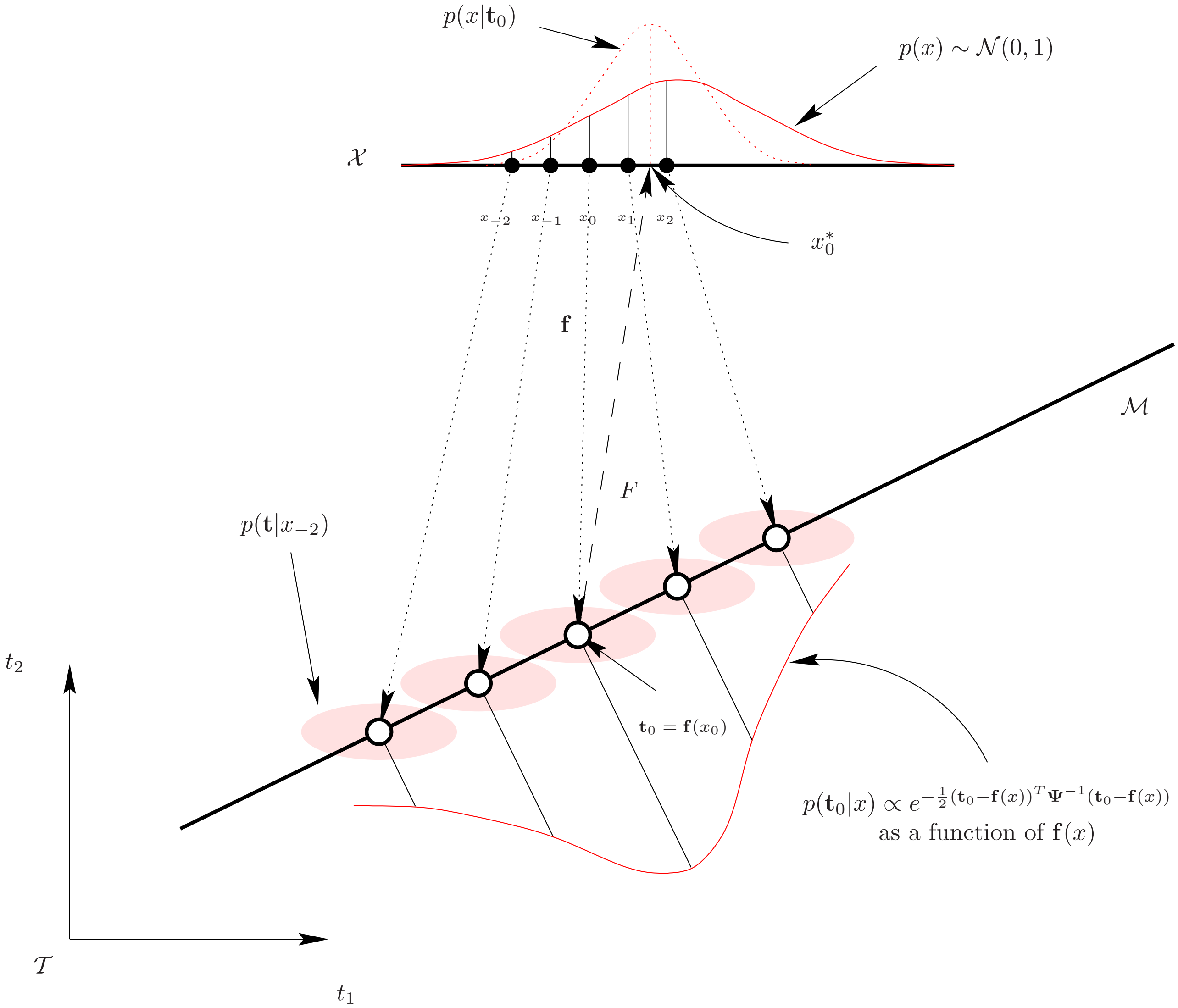


Figure 2.11: Offset of the reduced-dimension representative when using the posterior mean. This example demonstrates how mapping a latent space point onto data space and back using the posterior mean leads to a reduced-dimension representative which is different from the original latent space point. The latent variable model is a factor analysis with one-dimensional latent space and two-dimensional observed space. The latent space shows its prior distribution $p(x)$ (solid line) centred symmetrically around its mean. A point x_0 is mapped onto data space point $\mathbf{t}_0 = \mathbf{f}(x_0)$. Given the noise model $p(\mathbf{t}|x) \sim \mathcal{N}(\mathbf{f}(x), \mathbf{\Psi})$ alone, other nearby latent points x_{-1}, x_1, etc. could also have generated $\mathbf{t}_0$, but less likely than x_0 (and symmetrically so as we move away from $\mathbf{t}_0$, as marked, because of the symmetry of $p(\mathbf{t}_0|x)$). Each shaded ellipse represents $p(\mathbf{t}_0|x_i)$ for $i \in \{-2, \dots, 2\}$. But given the prior distribution in latent space, latent points to the nearby right side of x_0 are more likely a priori than latent points to the left: $p(x_{-2}) < p(x_{-1}) < p(x_0) < p(x_1) < p(x_2)$. Therefore, the posterior distribution in latent space $p(x|\mathbf{t}_0) \propto p(\mathbf{t}_0|x)p(x)$ (dotted line) is offset towards latent points which are more likely a priori (towards the prior mean in this example) and thus its centre does not match the original latent point x_0. This offset is inherent to the choice of prior distribution in latent space and noise model and is therefore unavoidable.

which implies two conditions:

$$\boldsymbol{\mu}_X = \mathbf{0} \tag{2.50a}$$
$$\mathbf{A}\boldsymbol{\Lambda} = \mathbf{I}. \tag{2.50b}$$

Condition (2.50a) is readily satisfied by both factor analysis and PCA, and we assume it holds from now on. As for condition (2.50b), using eqs. (2.58) and (2.59) it becomes:

$$\mathbf{I} = \mathbf{A}\boldsymbol{\Lambda} \overset{(2.58)}{=} \hat{\boldsymbol{\Sigma}}_X^{-1}\boldsymbol{\Lambda}^T\boldsymbol{\Sigma}_T^{-1}\boldsymbol{\Lambda} \overset{(2.59)}{=} (\boldsymbol{\Sigma}_X^{-1} + \boldsymbol{\Lambda}^T\boldsymbol{\Sigma}_T^{-1}\boldsymbol{\Lambda})^{-1}\boldsymbol{\Lambda}^T\boldsymbol{\Sigma}_T^{-1}\boldsymbol{\Lambda} \Leftrightarrow \boldsymbol{\Sigma}_X^{-1} + \boldsymbol{\Lambda}^T\boldsymbol{\Sigma}_T^{-1}\boldsymbol{\Lambda} = \boldsymbol{\Lambda}^T\boldsymbol{\Sigma}_T^{-1}\boldsymbol{\Lambda} \tag{2.51}$$

which is impossible. Therefore *no linear-normal latent variable model exists that satisfies* $\mathrm{E}\left\{\mathbf{x}|\mathbf{f}(\mathbf{x}_0)\right\} = \mathbf{x}_0$, in particular factor analysis and PCA. Figure 2.11 explains intuitively this.

Let us analyse when condition (2.50b) holds approximately. This will happen when $\boldsymbol{\Lambda}^T\boldsymbol{\Sigma}_T^{-1}\boldsymbol{\Lambda} \gg \boldsymbol{\Sigma}_X^{-1}$ or equivalently when $\mathbf{M}^T\boldsymbol{\Sigma}_T^{-1}\mathbf{M} \gg \mathbf{I}$ with $\mathbf{M} \overset{\text{def}}{=} \boldsymbol{\Lambda}\boldsymbol{\Sigma}_X^{1/2}$. Since $\mathbf{M}\mathbf{M}^T = \boldsymbol{\Sigma} - \boldsymbol{\Sigma}_T$ is the covariance associated with the latent variables and $\boldsymbol{\Sigma}_T$ the covariance associated to the noise, then $\mathbf{F}\circ\mathbf{f} \equiv$ identity will hold approximately when the covariance of the noise is much smaller than the covariance due to the latent variables. We consider two limit cases:

- The **low noise limit**, where

  $$\mathbf{A} = \hat{\boldsymbol{\Sigma}}_X^{-1}\boldsymbol{\Lambda}^T\boldsymbol{\Sigma}_T^{-1} = (\boldsymbol{\Sigma}_X^{-1} + \boldsymbol{\Lambda}^T\boldsymbol{\Sigma}_T^{-1}\boldsymbol{\Lambda})^{-1}\boldsymbol{\Lambda}^T\boldsymbol{\Sigma}_T^{-1} \approx \hat{\mathbf{A}} \overset{\text{def}}{=} (\boldsymbol{\Lambda}^T\boldsymbol{\Sigma}_T^{-1}\boldsymbol{\Lambda})^{-1}\boldsymbol{\Lambda}^T\boldsymbol{\Sigma}_T^{-1}.$$

  The matrix $\hat{\mathbf{A}}$ is called the **Bartlett scores** in the factor analysis literature. It can also be derived from a distribution-free argument as follows: if the mapping from latent to data space is represented as $\mathbf{t} = \boldsymbol{\Lambda}\mathbf{x} + \boldsymbol{\mu}_T + \mathbf{e}$ where $\mathbf{e} \overset{\text{def}}{\sim} \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_T)$ and the mapping from latent to data space as $\mathbf{x}^* = \mathbf{B}(\mathbf{t} - \boldsymbol{\mu}) \overset{(2.55)}{=} \mathbf{B}\boldsymbol{\Lambda}(\mathbf{x} - \boldsymbol{\mu}_X) + \mathbf{B}\mathbf{e}$, then it can be proven (Bartholomew, 1987, pp. 66–69) that:

  $$\hat{\mathbf{A}} = \arg\min_{\mathbf{B}\boldsymbol{\Lambda}=\mathbf{I}} \mathbf{B}\boldsymbol{\Sigma}_T\mathbf{B}^T.$$

  That is, $\hat{\mathbf{A}}$ is the matrix $\mathbf{B}$ satisfying $\mathbf{B}\boldsymbol{\Lambda} = \mathbf{I}$ that minimises the residual variance of the reduced-dimension representative, $\mathrm{var}\left\{\mathbf{e}\right\} = \mathbf{B}\boldsymbol{\Sigma}_T\mathbf{B}^T$.

  In practice there tends to be little difference between Thomson and Bartlett scores, since $\boldsymbol{\Lambda}^T\boldsymbol{\Psi}^{-1}\boldsymbol{\Lambda}$ is often approximately diagonal and the difference is just a rescaling of the $\mathbf{x}$ variables.

- The **isotropic zero noise limit**, where from corollary 2.12.2 and assuming $\boldsymbol{\Lambda}$ full-rank:

  $$\boldsymbol{\Sigma}_T = k\mathbf{I} \text{ and } k \to 0^+ \Rightarrow \begin{cases} \mathbf{A} \to \boldsymbol{\Lambda}^+ = (\boldsymbol{\Lambda}^T\boldsymbol{\Lambda})^{-1}\boldsymbol{\Lambda}^T \\ \mathbf{x}|\mathbf{t} \to \delta(\boldsymbol{\Lambda}^+(\mathbf{t} - \boldsymbol{\mu}_T)). \end{cases}$$

  The matrix $\boldsymbol{\Lambda}^+$ is the **pseudoinverse** of $\boldsymbol{\Lambda}$. The pseudoinverse is the matrix $\boldsymbol{\Lambda}^*$ that minimises $\|\mathbf{I} - \boldsymbol{\Lambda}^*\boldsymbol{\Lambda}\|_2^2$ (Boullion and Odell, 1971); thus, it is the matrix that achieves the least squares reconstruction error. The Thomson scores are also optimal in the least squares sense, being the mean of the posterior distribution in latent space. The difference is that the pseudoinverse gives equal importance to each point $\mathbf{x}$ in the latent space, while the Thomson scores weight each value $\mathbf{x}$ according to the normal distribution $\mathrm{E}\left\{\mathbf{x}|\mathbf{t}\right\}$. Thus, it "pulls" points towards the mean, since $p(\mathbf{x}|\mathbf{t})$, being normal, decreases when going away from its mean. The disagreement of both scores is then due to the Thomson scores using a joint probability model for $\mathbf{x}$ and $\mathbf{t}$ and the pseudoinverse using no model.

  If the noise tends to zero but not isotropically, then $\mathbf{A} \nrightarrow \boldsymbol{\Lambda}^+$ necessarily; e.g. in fig. 2.12 if $\boldsymbol{\Sigma}_T = k\left(\begin{smallmatrix} S_1 & 0 \\ 0 & S_2 \end{smallmatrix}\right)$ with $S_1 \neq S_2$ and $k \to 0^+$ then $\mathbf{A}, \hat{\mathbf{A}} \to \mathbf{A}_0 \overset{\text{def}}{=} \frac{1}{S_2\lambda_1^2 + S_1\lambda_2^2}(S_2\lambda_1 \;\; S_1\lambda_2)$ which is different from $\boldsymbol{\Lambda}^+ = \frac{1}{\lambda_1^2+\lambda_2^2}(\lambda_1 \;\; \lambda_2)$. However, $\mathbf{A}_0$ and $\boldsymbol{\Lambda}^+$ only differ in how they map points $\mathbf{t} \notin \mathcal{M}$ because for $\mathbf{t} \in \mathcal{M}$ they are equivalent (since $\hat{\mathbf{A}}\boldsymbol{\Lambda} = \boldsymbol{\Lambda}^+\boldsymbol{\Lambda} = \mathbf{I}$). Since the points $\mathbf{t} \notin \mathcal{M}$ are unreachable in the zero-limit noise, the difference is irrelevant.

$\mathbf{A}$ (Thomson scores) never coincides with $\boldsymbol{\Lambda}^+$ (pseudoinverse) since it does not satisfy any of the Penrose conditions (A.1). $\hat{\mathbf{A}}$ (Bartlett scores) satisfies all of the Penrose conditions except (A.1a) because $\boldsymbol{\Lambda}\hat{\mathbf{A}}$ is not symmetric in general and so it does not coincide with the pseudoinverse either (although it is a left weak generalised inverse of $\boldsymbol{\Lambda}$; Boullion and Odell, 1971); but $\hat{\mathbf{A}}$ does coincide with the pseudoinverse in

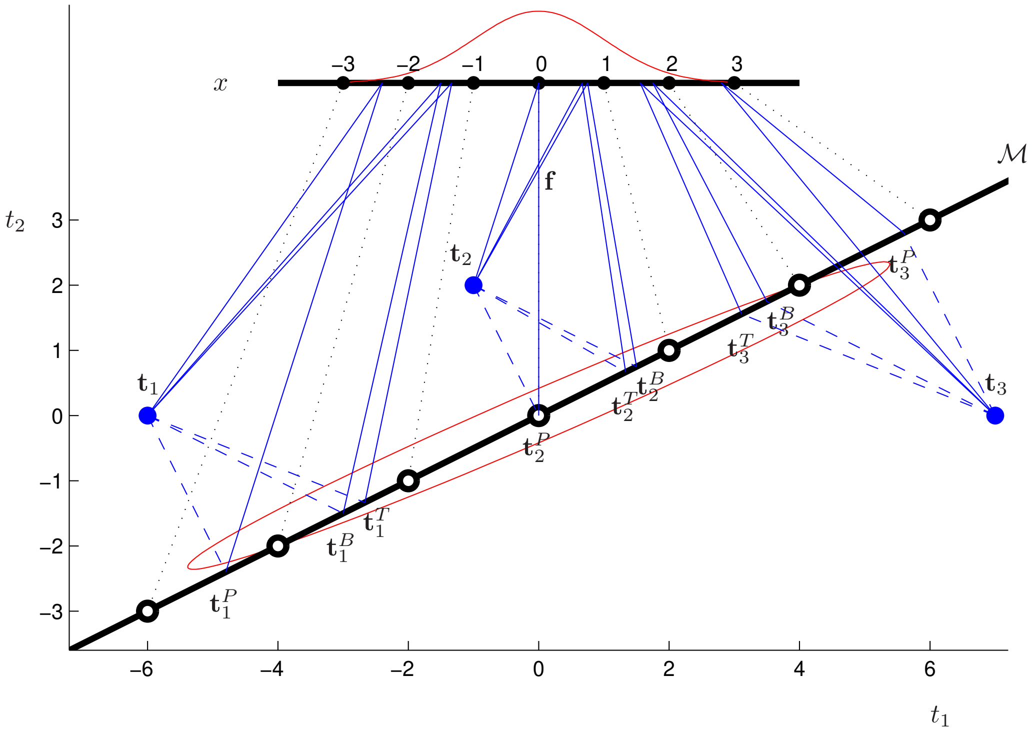


Figure 2.12: Different dimensionality reduction mappings for linear-normal models. Using the symbols of theorem 2.12.1, the numerical values are: $\mu_X = \mathbf{0}$, $\mathbf{\Sigma}_X = \mathbf{I}$, $\mathbf{\Lambda} = \left(\begin{smallmatrix} 2 \\ 1 \end{smallmatrix}\right)$, $\boldsymbol{\mu}_T = \mathbf{0}$, $\mathbf{\Sigma}_T = \left(\begin{smallmatrix} 1 & 0 \\ 0 & \frac{1}{4} \end{smallmatrix}\right)$. Therefore $\boldsymbol{\mu} = \mathbf{0}$, $\mathbf{\Sigma} = \left(\begin{smallmatrix} 5 & 2 \\ 2 & \frac{5}{4} \end{smallmatrix}\right)$, $\mathbf{A} = \frac{1}{9}(2\ 4)$, $\hat{\mathbf{A}} = \frac{1}{8}(2\ 4)$ and $\mathbf{\Lambda}^+ = \frac{1}{5}(2\ 1)$. The graph shows schematically how the latent space $\mathcal{X} = \mathbb{R}$, with a normal prior distribution $\mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu}_X, \mathbf{\Sigma}_X)$ is mapped onto $\mathcal{M}$, inducing a normal distribution $\mathbf{t} \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{\Sigma})$ (indicated by its unit Mahalanobis ellipse) in $\mathcal{T} = \mathbb{R}^2$. $\mathbf{t}_n^T$, $\mathbf{t}_n^B$ and $\mathbf{t}_n^P$ represent the reconstructed points using $\mathbf{A}$ (Thomson scores), $\hat{\mathbf{A}}$ (Bartlett scores) and $\mathbf{\Lambda}^+$ (pseudoinverse), respectively, for data point $\mathbf{t}_n$ (where $n = 1, 2, 3$). The dimension-reduced representatives are marked on the latent space by the corner of the solid lines; the dashed lines join the original and reconstructed points. The dashed lines for the pseudoinverse are parallel with respect to each other and orthogonal to $\mathcal{M}$: $\mathbf{\Lambda\Lambda}^+$ is an orthogonal projection. The dashed lines for the Bartlett scores are parallel with respect to each other but not orthogonal to $\mathcal{M}$: $\mathbf{\Lambda}\hat{\mathbf{A}}$ is an oblique projection. The dashed lines for the Thomson scores are not parallel with respect to each other: $\mathbf{\Lambda A}$ is not a projection.

the particular case where $\mathbf{\Sigma}_T$ is isotropic (precisely the case of PCA). As mentioned before, in the isotropic zero-noise limit all three scores coincide.

Observe that $\mathbf{f} \circ \mathbf{F}$ maps an arbitrary point in data space onto the manifold $\mathcal{M}$. However, of $\mathbf{A}$, $\hat{\mathbf{A}}$ and $\mathbf{\Lambda}^+$ only $\hat{\mathbf{A}}$ and $\mathbf{\Lambda}^+$ give rise to projection matrices: oblique $\mathbf{\Lambda}\hat{\mathbf{A}}$ and orthogonal $\mathbf{\Lambda\Lambda}^+$, respectively[39].

Table 2.4 summarises the three types of scores and figure 2.12 illustrates the typical situation when all of them differ.

2.9.1.2 Independent component analysis

Given the discussion of section 2.9.1.1, since in the standard ICA model there is no noise and $\mathbf{\Lambda}$ is full rank, the dimensionality reduction mapping to use is $\mathbf{F}(\mathbf{t}) \stackrel{\text{def}}{=} \mathbf{\Lambda}^+\mathbf{t} = (\mathbf{\Lambda}^T\mathbf{\Lambda})^{-1}\mathbf{\Lambda}^T\mathbf{t}$ (i.e., the unmixing matrix $\mathbf{A} = \mathbf{\Lambda}^+$ of section 2.6.3), which satisfies $\mathbf{F} \circ \mathbf{f} \equiv$ identity.

The goal of ICA is not really dimensionality reduction but separation of linearly mixed sources, which is attained by the unmixing matrix. Besides, the literature of ICA has mostly been concerned with the case where the number of sensors is equal to the number of sources, although research of more general situations

[39] A matrix $\mathbf{P}$ which verifies symmetry ($\mathbf{P} = \mathbf{P}^T$) and idempotence ($\mathbf{P}^2 = \mathbf{P}$) is an **orthogonal projection** matrix. If it only verifies idempotence but not symmetry then it is an **oblique projection** matrix.

Scores name	Matrix expression	Justification
Thomson	$\mathbf{A} \stackrel{\text{def}}{=} (\boldsymbol{\Sigma}_X^{-1} + \boldsymbol{\Lambda}^T \boldsymbol{\Sigma}_T^{-1} \boldsymbol{\Lambda})^{-1} \boldsymbol{\Lambda}^T \boldsymbol{\Sigma}_T^{-1}$	Least-squares for linear normal model
Bartlett	$\hat{\mathbf{A}} \stackrel{\text{def}}{=} (\boldsymbol{\Lambda}^T \boldsymbol{\Sigma}_T^{-1} \boldsymbol{\Lambda})^{-1} \boldsymbol{\Lambda}^T \boldsymbol{\Sigma}_T^{-1}$	Low-noise limit of $\mathbf{A}$; also distribution-free
Pseudoinverse	$\boldsymbol{\Lambda}^+ = (\boldsymbol{\Lambda}^T \boldsymbol{\Lambda})^{-1} \boldsymbol{\Lambda}^T$	Isotropic zero-noise limit of $\mathbf{A}$; also model-free least-squares

Table 2.4: Dimensionality reduction matrices for linear-normal models with the posterior mean.

is progressing. ICA has been successfully applied to a number of blind separation problems, including elimination of artifacts from electroencephalographic (EEG) data, separation of speech sources (the cocktail party problem), processing of arrays of radar and sonar signals and restoration of images (see references in Hyvärinen and Oja, 2000; Hyvärinen et al., 2001).

2.9.1.3 Independent factor analysis

The posterior distribution in latent space for IFA (2.38) is a convex combination of factor analysis-like posterior probabilities (2.54), so we expect $\mathbf{F} \circ \mathbf{f} \neq$ identity except in the zero-noise limit.

2.9.1.4 GTM

We consider two cases for the definition of $\mathbf{F}$:

- Posterior mean, $\mathbf{F}(\mathbf{t}) = \mathrm{E}\left\{\mathbf{x}|\mathbf{t}\right\}$. Then:

$$\mathbf{F}(\mathbf{t}) = \mathrm{E}\left\{\mathbf{x}|\mathbf{t}\right\} = \sum_{k=1}^{K} \mathbf{x}_k p(\mathbf{x}_k|\mathbf{t}) = \sum_{k=1}^{K} \mathbf{x}_k \rho_k(\mathbf{t})$$

  where

$$\rho_k(\mathbf{t}) = p(\mathbf{x}_k|\mathbf{t}) = \frac{p(\mathbf{t}|\mathbf{x}_k)p(\mathbf{x}_k)}{p(\mathbf{t})} = \frac{p(\mathbf{t}|\mathbf{x}_k)}{\sum_{k=1}^{K} p(\mathbf{t}|\mathbf{x}_k)} = \frac{e^{-\frac{1}{2\sigma^2}\|\mathbf{t}-\mathbf{f}(\mathbf{x}_k)\|^2}}{\sum_{k=1}^{K} e^{-\frac{1}{2\sigma^2}\|\mathbf{t}-\mathbf{f}(\mathbf{x}_k)\|^2}} \in (0,1).$$

  Thus, $\mathbf{F}(\mathbf{t})$ is a point in the convex hull of $\{\mathbf{x}_k\}_{k=1}^K$. Observe that for any $k' \in \{1,\dots,K\}$, $\rho_k(\mathbf{f}(\mathbf{x}_{k'})) > 0$ for all $k = 1,\dots,K$. So $\mathbf{F}(\mathbf{f}(\mathbf{x}_{k'})) \neq \mathbf{x}_{k'}$. However, $\mathbf{F}(\mathbf{f}(\mathbf{x}_{k'}))$ will be very close to $\mathbf{x}_{k'}$ when the Gaussian components in data space, centred at $\{\mathbf{f}(\mathbf{x}_k)\}_{k=1}^K$, are widely separated with respect to the noise standard deviation σ, since the tails of the Gaussian fall rapidly and then $\rho_k(\mathbf{f}(\mathbf{x}_{k'})) \approx \delta_{k'k}$ (again the zero-noise limit).

- Posterior mode, $\mathbf{F}(\mathbf{t}) = \arg\max_{\mathbf{x}\in\mathcal{X}} p(\mathbf{x}|\mathbf{t})$: it clearly satisfies $\mathbf{F}(\mathbf{f}(\mathbf{x})) = \mathbf{x}$ for $\{\mathbf{x}_k\}_{k=1}^K$ but not for any other $\mathbf{x} \in \mathbb{R}^L \setminus \{\mathbf{x}_k\}_{k=1}^K$ (which get mapped onto some $\mathbf{x}_k$).

  Also, since $p(\mathbf{t})$ is independent of $\mathbf{x}$ and $p(\mathbf{x}_k) = \frac{1}{K}$, then $\arg\max_{\mathbf{x}\in\mathcal{X}} p(\mathbf{x}|\mathbf{t}) = \arg\max_{k=1,\dots,K} p(\mathbf{t}|\mathbf{x}_k) = \arg\min_{k=1,\dots,K} \|\mathbf{t} - \mathbf{f}(\mathbf{x}_k)\|$, i.e., the latent grid point whose image is closest to $\mathbf{t}$. Thus, the dimensionality reduction mapping behaves like vector quantisation in the observed space based on the Euclidean distance using $\{\mathbf{f}(\mathbf{x}_k)\}_{k=1}^K$ as codebook.

2.9.2 Continuity of the dimensionality reduction mapping and regularisation

If the latent variable model has captured the structure of the data appropriately, one would not expect a multimodal distribution $p(\mathbf{x}|\mathbf{t})$, except perhaps if the data manifold is so twisted that it intersects itself or nearly so. Unfortunately, in practice the manifold $\mathcal{M}$ induced in data space can be badly twisted even when the data manifold is relatively smooth, as fig. 2.13 shows, due to a suboptimal local maximum of the log-likelihood. In this case, the induced manifold $\mathcal{M}$ can retrace itself and give rise to multimodal posterior distributions in latent space, since the same area of the data manifold is covered by different areas of the latent space. Regularising the mapping $\mathbf{f}$ (from latent to data space) so that highly twisted mappings are penalised is a possible solution. From a Bayesian point of view this requires introducing a prior distribution on the mapping parameters controlled by a hyperparameter (as in GTM), but determining good hyperparameter values is a hard problem (MacKay, 1999).

The continuity of the dimensionality reduction mapping $\mathbf{x} = \mathbf{F}(\mathbf{t})$ depends on several factors. If we define $\mathbf{F}$ as the mean of the posterior distribution $p(\mathbf{x}|\mathbf{t})$ of eq. (2.49) then $\mathbf{F}$ will be continuous by the first fundamental theorem of calculus (Spivak, 1967, p. 240, Th. 1 and also p. 230, Th. 8), because the mean is defined as an integral; further, $\mathbf{F}$ will be differentiable at every point where $p(\mathbf{x}|\mathbf{t})$ is continuous itself (as a function of $\mathbf{t}$). For example, the dimensionality reduction mappings for factor analysis and PCA are continuous, as is obvious from the linearity of the function (2.19). But if we define $\mathbf{F}$ as the (global) mode of $p(\mathbf{x}|\mathbf{t})$, then it may not be continuous if $\mathbf{F}$ can be multimodal, as fig. 2.14 shows. Section 9.2 shows more examples of the discontinuity of the $\arg\max(\cdot)$ function.

For GTM, $\mathbf{F}(\mathbf{t})$ defined as the posterior mode can be discontinuous, as would happen in the case of figure 2.13 (right). However, its continuity is likely in practical situations where the posterior distribution is unimodal and sharply peaked for the majority of the training set data points.

For latent variable models (like GTM) that sample the latent space, the concept of continuity in the latent space—which is now discretised— becomes blurred if the posterior mode is used (the posterior mean remains continuous, although it will produce points not in the latent space grid). In this case, discontinuities of the dimensionality reduction mapping $\mathbf{F}$ will manifest themselves as abrupt jumps in the grid (larger than the distance between two neighbouring latent grid points) when the point in data space changes slightly.

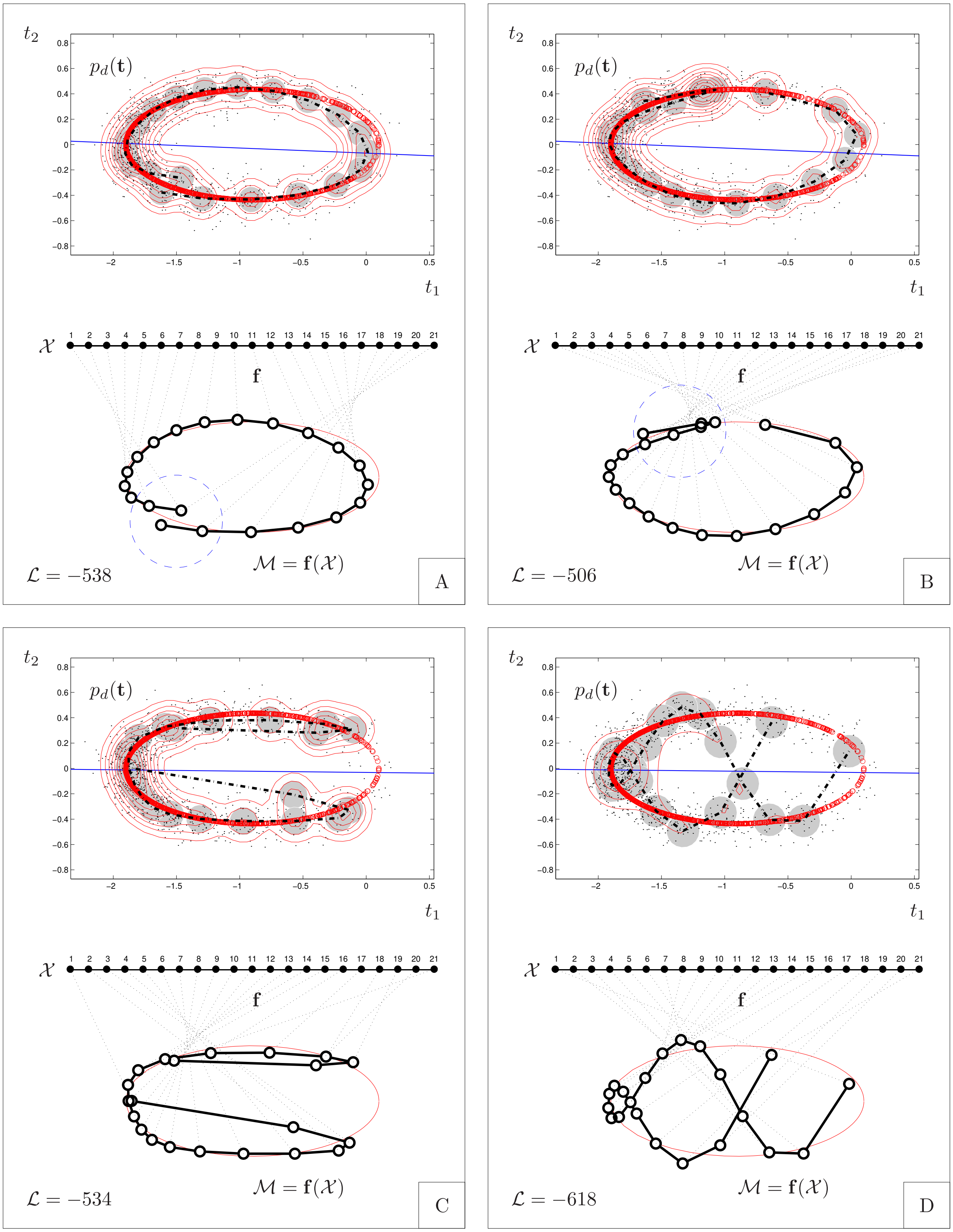
Finally, let us consider the case of a dimensionality reduction mapping defined as the point in the manifold $\mathcal{M}$ which is closest to the data point $\mathbf{t}$; that is, orthogonal projection ot $\mathbf{t}$ onto $\mathcal{M}$. This is the approach followed by the principal curves method. As mentioned in section 4.8, this definition leads to a discontinuous dimensionality reduction mapping if the manifold $\mathcal{M}$ is nonlinear. But, as discussed earlier, the presence of a noise model and a prior distribution in latent space preclude the use of an orthogonal projection.

2.9.3 Finite mixtures of latent variable models

We turn now to the subject of dimensionality reduction in finite mixtures of latent variable models. If the dimension of all the latent spaces was the same, one could think of using $\mathrm{E}\left\{\mathbf{x}|\mathbf{t}\right\}$ as the latent space representative of data point $\mathbf{t}$:

$$p(\mathbf{x}|\mathbf{t}) = \sum_{m=1}^{M} p(\mathbf{x}, m|\mathbf{t}) = \sum_{m=1}^{M} p(m|\mathbf{t}) p(\mathbf{x}|m, \mathbf{t}) \Longrightarrow$$
$$\mathrm{E}\left\{\mathbf{x}|\mathbf{t}\right\} = \int \mathbf{x} p(\mathbf{x}|\mathbf{t})\, d\mathbf{x} = \int \mathbf{x} \left(\sum_{m=1}^{M} p(m|\mathbf{t}) p(\mathbf{x}|m, \mathbf{t}) \right) d\mathbf{x} = \sum_{m=1}^{M} p(m|\mathbf{t})\, \mathrm{E}\left\{\mathbf{x}|m, \mathbf{t}\right\}$$

Figure 2.13 *(following page)*: Twisted manifolds and continuity of the dimensionality reduction mapping. Each of the plots A to D shows an estimate for a training set of $N = 1\,000$ points (t_1, t_2) generated (with additive normal noise of variance 0.01) from the Keplerian movement distribution of fig. 2.5 (first row), discussed in section 2.2.3: a one-dimensional manifold embedded in the plane. In each plot, an unregularised GTM model was fitted that used $K = 21$ latent grid points and $F = 5$ radial basis functions with a standard deviation $s = 1$ times their separation. Two different training sets were used, one for plots A, B and another one for plots C, D. The EM starting point was random for plots A, C and the line of the first principal component for plots B, D. For each plot, the upper graph shows: the GTM manifold (thick dashed line), the true, elliptical data manifold (small circles, whose density mimicks the distribution along the data manifold), the K Gaussian components associated to the latent grid points (eqs. (2.42) and (2.43), with a radius σ; the big, grey circles), the first principal component of the training set (solid straight line, almost coinciding with the $t_2 = 0$ axis), a contour plot of the GTM data space distribution $p_d(\mathbf{t})$ and the training set (scattered dots). The lower graph shows explicitly the way the one-dimensional latent space is twisted in two dimensions: the latent space grid is the horizontal top line, which is mapped onto the GTM data space manifold $\mathcal{M} = \mathbf{f}(\mathcal{X})$ (thick solid line), to be compared with the true data manifold (thin ellipse). All models have approximately the same log-likelihood $\mathcal{L}$ (except D, which is much worse), but the way the GTM manifold twists itself in data space affects differently the continuity of the dimensionality reduction mapping $\mathbf{F}$ when the posterior mode is used. Model B has discontinuities in the area marked by the dashed circle in the lower graph due to the latent grid points x_{20}–x_{21} retracing over x_{16}–x_{19}. Retracing is worse in model C (points x_{20}–x_{21} retracing over x_{15}–x_{19} and points x_1–x_2 over x_3–x_{12}) and much worse in model D (with multiple branch-switching). A further discontinuity appears in all four models (marked by the dashed circle in the lower graph of plot A) due to the mismatch between the latent space topology (open curve) and the data manifold topology (closed curve).



t_2
$p_d(\mathbf{t})$
t_1
$\mathcal{X}$
$\mathbf{f}$
$\mathcal{M} = \mathbf{f}(\mathcal{X})$
$\mathcal{L} = -538$
A
$\mathcal{L} = -506$
B
$\mathcal{L} = -534$
C
$\mathcal{L} = -618$
D

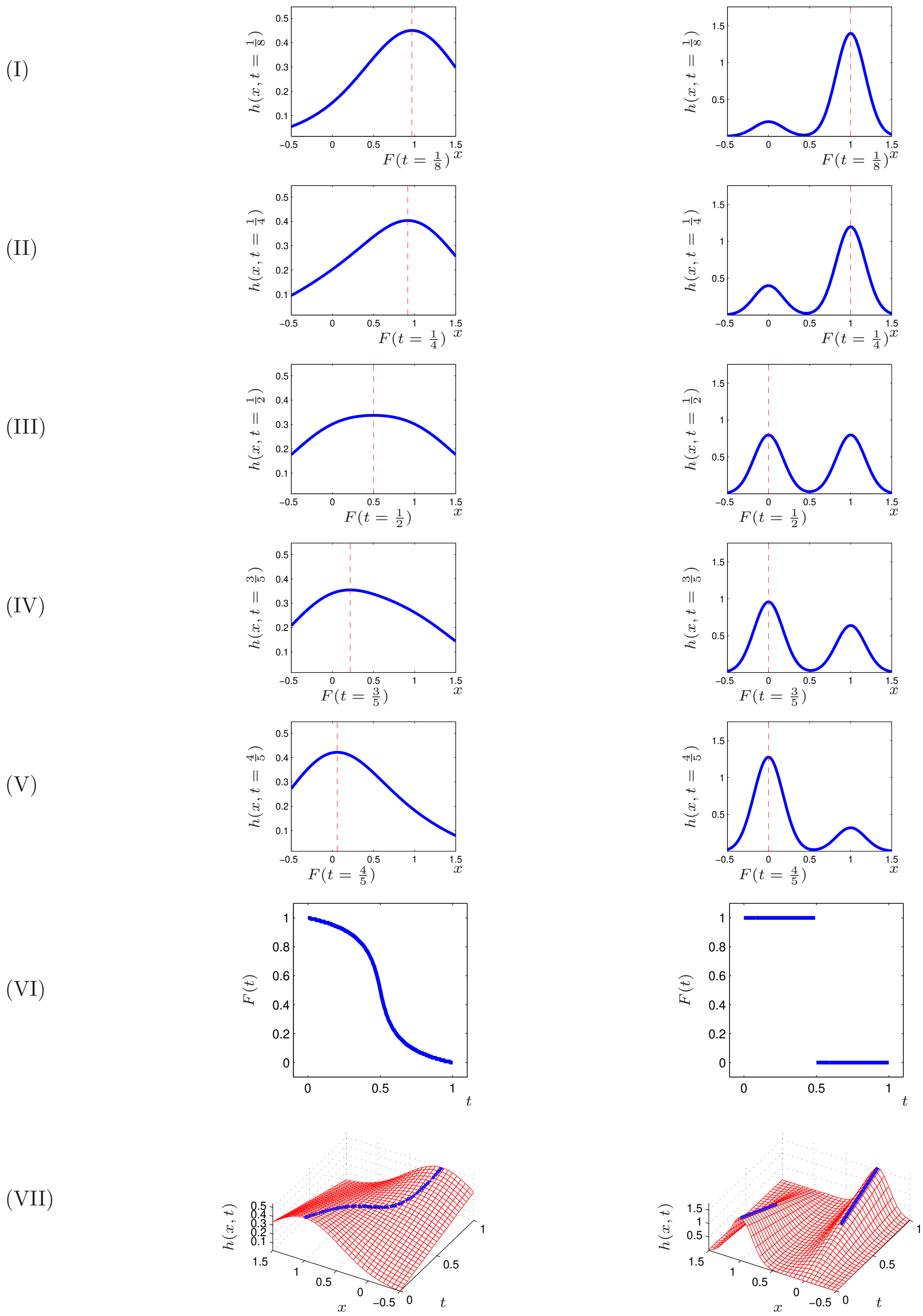


Figure 2.14: Discontinuity of the $\arg\max(\cdot)$ or mode function. The figure shows plots of the function $F(t) = \arg\max_x h(x,t)$ where $h(x,t) \stackrel{\text{def}}{=} (1-t)\mathcal{N}(x;\mu_1,\sigma^2) + t\mathcal{N}(x;\mu_2,\sigma^2)$ for $t \in [0,1]$ and $x \in (-\infty,\infty)$, with $\mu_1 = 0$ and $\mu_2 = 1$. The left column corresponds to $\sigma = 0.8$, so that the two normal functions overlap heavily and $h(x;t)$ is unimodal for any given $t \in [0,1]$: $F(t)$ is continuous. The right column corresponds to $\sigma = 0.25$, so that the two normal functions are widely separated and $h(x;t)$ is bimodal for (almost) any given $t \in [0,1]$: $F(t)$ is discontinuous. Rows (I)–(V) show, for several values of t, the function $h(x;t)$ and the location of $F(t)$. Row (VI) shows $F(t)$. Row (VII) shows a 3D view of $h(x,t)$ and, superimposed as a thick line, $F(t)$.

where $\mathrm{E}\{\mathbf{x}|m,\mathbf{t}\}$ is the representative in mth latent space proposed by component m, with responsibility $R_m = p(m|\mathbf{t})$ for having generated $\mathbf{t}$.

However, even if all the latent spaces had the same dimension, they will not necessarily correspond to the same coordinate systems (e.g. consider the effect of rotating differently several factor spaces of dimension L). Therefore, when dealing with mixtures of latent variable models, it makes no sense to talk about a single representative, but about M of them, one for each of the M latent spaces, and averaging these representatives is meaningless. Alternatively, a reduced-dimension representative can be obtained as the reduced-dimension representative of the mixture component with the highest responsibility: $\mathbf{x}^* = \mathbf{F}(\mathbf{t}) = \mathbf{x}^*_{m^*}$ such that $m^* = \arg\max_m p(m|\mathbf{t})$.

2.9.3.1 Reconstruction

While averaging the reduced-dimension representatives is not possible, it intuitively makes sense to average the reconstructed vectors (with respect to the responsibilities), because the data space coordinates are the same for any component:

$$\mathbf{t}^* = \sum_{m=1}^{M} p(m|\mathbf{t})\mathbf{f}_m(\mathbf{F}_m(\mathbf{t})) = \sum_{m=1}^{M} p(m|\mathbf{t})\mathbf{t}^*_m \tag{2.52}$$

where $\mathbf{t}^*_m = \mathbf{f}_m(\mathbf{F}_m(\mathbf{t}))$ is the reconstructed vector by the latent variable model of component m. This can be seen as a nonlinear projection of the original vector $\mathbf{t}$ onto the convex hull of $\{\mathbf{t}^*_m\}_{m=1}^M$, which is a subset of the linear subspace spanned by $\{\mathbf{t}^*_m\}_{m=1}^M$. The projection is nonlinear because even if $\mathbf{f}_m \circ \mathbf{F}_m$ is linear for all m, the $p(m|\mathbf{t})$ are not. The vectors $\{\mathbf{t}^*_m\}_{m=1}^M$ are not fixed, but depend on the particular input vector $\mathbf{t}$.

2.9.3.2 Classification

For classification purposes, one could assign the data vector $\mathbf{t}$ to the most responsible component $m^* = \arg\max_m p(m|\mathbf{t})$ and reconstruct the vector according to that component alone (cf. eq. (2.52)):

$$\mathbf{t}^* = \mathbf{t}^*_{m^*}. \tag{2.53}$$

As with latent variable models, the representatives could be computed as some suitable summary value of the posterior distribution $p(\mathbf{x}|m^*,\mathbf{t})$, e.g. the mean, $\mathrm{E}\{\mathbf{x}|m^*,\mathbf{t}\}$, or the mode, $\arg\max_m p(\mathbf{x}|m^*,\mathbf{t})$.

As with general mixture models, if for a given data vector no posterior probability is big enough, no component of the mixture will explain it properly. This may be due to the mixture model being insufficient, to the data vector lying on the boundary of two (or more) components, or to the data vector itself being an outlier (or a novel point). In the latter case the point could be rejected if $p(\mathbf{t}) < \theta$ for a suitable threshold $\theta > 0$ (which needs to be determined heuristically).

2.9.3.3 Reconstruction in general finite mixtures

We consider again equation (2.47), but where now the components $p(\mathbf{t}|m)$ are not latent variable models but certain arbitrary densities or probability mass functions. For the purposes of reconstruction, each component must provide with a reconstructed vector as a function of the original data vector $\mathbf{t}$. For a general probability distribution (without an underlying latent variable model) the best we can do is to choose a value that summarises the distribution, such as the mean, median or mode. We will call this the **prototype** of the distribution, $\mathbf{t}^*_m$. Then, the vector reconstructed by the finite mixture distribution will be $\mathbf{t}^* = \sum_{m=1}^M \pi_m \mathbf{t}^*_m$, as in the finite mixture of latent variable models. Since the $\{\mathbf{t}^*_m\}_{m=1}^M$ are fixed (unlike in reconstruction in mixtures of latent variable models), this amounts to a sort of weighted vector quantisation using the $\{\mathbf{t}^*_m\}_{m=1}^M$ as codebook vectors (although M will usually be small), which gives poor results. Effectively, this means that the finite mixture is performing cluster analysis rather than reconstruction or dimensionality reduction.

For example, for a normal distribution $\mathcal{N}(\boldsymbol{\mu},\boldsymbol{\Sigma})$, the prototype is the parameter $\boldsymbol{\mu}$, which is the mean, median and mode. For a multivariate Bernoulli distribution $\mathcal{B}(\mathbf{p})$, with $p(\mathbf{t}) = \prod_{d=1}^D p_d^{t_d}(1-p_d)^{1-t_d}$, the prototype can be given by the mean $\mathbf{p}$ (which will not be a binary vector in general) or the mode $\lfloor \mathbf{p} + \frac{1}{2}\mathbf{1} \rfloor$ (which is binary). For binary data, a finite mixture of multivariate Bernoulli distributions (with M components) is called *latent class analysis* (with M classes), as mentioned in section 2.1.

For fixed $\{\mathbf{t}^*_m\}_{m=1}^M$, the set

$$\left\{ \sum_{m=1}^M \pi_m \mathbf{t}^*_m : \sum_{m=1}^M \pi_m = 1, \quad 0 \le \pi_m \le 1, \quad m = 1,\dots,M \right\}$$

is the convex hull of $\{\mathbf{t}^*_m\}_{m=1}^M$, which is a subset of the linear subspace spanned by $\{\mathbf{t}^*_m\}_{m=1}^M$. Hence, this method of reconstruction is more limited than unrestricted linear reconstruction—and therefore more restricted than PCA using M components, whatever the individual component distributions are. But even if the original data vector $\mathbf{t}$ lies in the convex hull of $\{\mathbf{t}^*_m\}_{m=1}^M$, the reconstructed vector $\mathbf{t}^*$ will not necessarily be equal to $\mathbf{t}$.

In a *responsibility plot*, the responsibility vector $\mathbf{R}(\mathbf{t}) = (p(1|\mathbf{t}), \dots, p(M|\mathbf{t}))^T$ is plotted in the $[0,1]^M$ hypercube, with each axis corresponding to one component. For the mixture to model the data well, the points should accumulate near the axes, indicating that for each point, there is always one component clearly responsible for having generated it. In other words, the distribution $p(m|\mathbf{t})$ should be almost degenerate, concentrating most of the mass in one value of m. Because $\sum_{m=1}^M p(m|\mathbf{t}) = 1$ is the equation of a hyperplane in $\mathbb{R}^M$, we can plot the points in the intersection between the hypercube $[0,1]^M$ and that hyperplane, which meets the coordinate axes at distance $+1$ from the origin. This region is a line for $M = 2$ and an equilateral triangle for $M = 3$. The mixture will perform well if the projected points fall near the vertices of that region.

2.10 Applications for dimensionality reduction

The traditional types of continuous latent variable models (factor analysis and PCA) have been extensively used for dimensionality reduction and related problems such as feature extraction or covariance structure analysis. Examples of such applications can be found in textbooks, e.g. Bartholomew (1987) or Everitt (1984) for factor analysis and Jolliffe (1986) or Diamantaras and Kung (1996) for PCA. Application of other types of continuous latent variable models has only started recently. ICA has been widely applied to signal separation problems, but in general not with the goal of dimensionality reduction, as noted in section 2.9.1.2. A handful of applications of GTM exist, often as a replacement of a Kohonen self-organising map; we study in detail our own application of GTM to a dimensionality reduction problem of speech in chapter 5. Also, we are currently working on the application of continuous latent variable models to a computational neuroscience problem, cortical map modelling (Swindale, 1996), from the point of view of dimensionality reduction. No applications of independent factor analysis exist yet.

2.11 A worked example

To concrete some of the abstract concepts discussed in this chapter, we conclude with a textbook-style example. The example also demonstrates several facts about latent variable models:

- That the marginalisation of the joint probability distribution $p(\mathbf{t}, \mathbf{x})$ can be analytically very difficult.
- The effect of varying noise levels in the induced probability distribution in data space $p(\mathbf{t})$, which is always between two limits:
  - No noise: $p(\mathbf{t}|\mathbf{x}) \to \delta(\mathbf{f}(\mathbf{x})) \Rightarrow p(\mathbf{t}) = \int_{\mathcal{X}} p(\mathbf{t}|\mathbf{x}) p(\mathbf{x})\, d\mathbf{x} \to \begin{cases} 0 & \mathbf{t} \notin \mathcal{M} = \mathbf{f}(\mathcal{X}) \\ p(\mathbf{x}) & \mathbf{t} \in \mathcal{M} = \mathbf{f}(\mathcal{X}), \mathbf{t} = \mathbf{f}(\mathbf{x}) \end{cases}$.
  - Large noise: $p(\mathbf{t}|\mathbf{x}) \approx$ constant $\forall \mathbf{x} \Rightarrow p(\mathbf{t}) \approx p(\mathbf{t}|\mathbf{x})$.

  In both limits the utility of the latent space is lost.

We consider a latent variable model with the following characteristics[40]:

- Latent space of dimension $L = 1$ and piecewise linear prior distribution $p_l(x) = \frac{2}{a^2+b^2} |x|$ on the interval $[-a, b]$ for $a, b \geq 0$.
- Linear mapping from latent onto data space $\mathbf{f}(x) = \mathbf{u}x + \mathbf{v}$.
- D-dimensional data space with Gaussian noise model $p_n(\mathbf{t}|x) = \mathcal{N}(\mathbf{f}(x), \mathbf{\Sigma})$.

Thus, the induced distribution in data space $p_d(\mathbf{t})$ can be obtained as follows:

$$p_d(\mathbf{t}) = \int_{-a}^{b} p_n(\mathbf{t}|x) p_l(x)\, dx = \int_{-a}^{b} \frac{1}{\sqrt{|2\pi\mathbf{\Sigma}|}} e^{-\frac{1}{2}(\mathbf{t}-(\mathbf{u}x+\mathbf{v}))^T \mathbf{\Sigma}^{-1} (\mathbf{t}-(\mathbf{u}x+\mathbf{v}))} \frac{2}{a^2+b^2} |x|\, dx.$$

[40] We notate the prior, noise and data density functions as p_l, p_n and p_d, respectively.

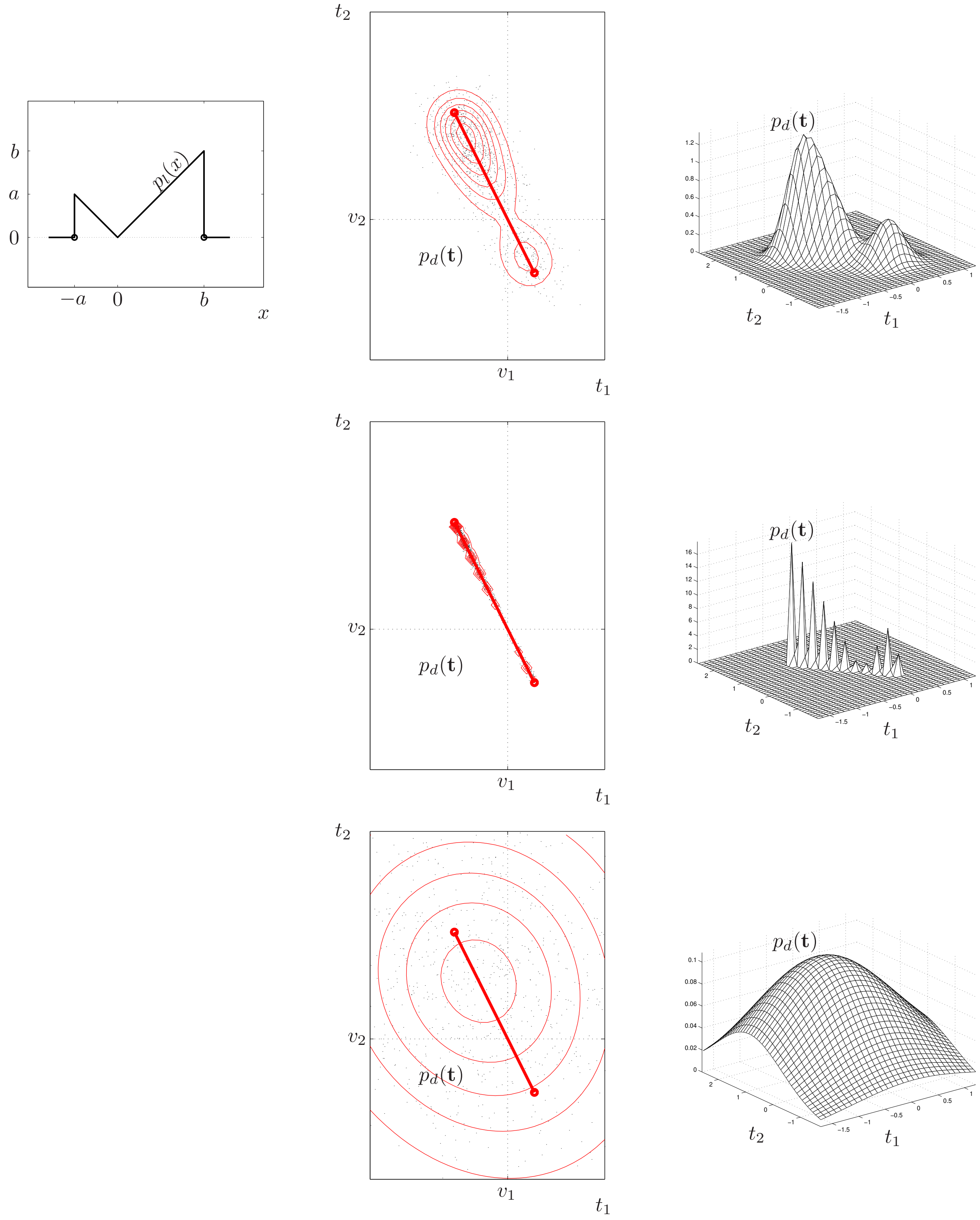


Figure 2.15: Distributions for the example of section 2.11 and effect of varying noise. The top row shows the prior $p_l(x)$ in latent space (left), a sample of 1000 points and a contour plot of the induced distribution $p_d(\mathbf{t})$ in data space (centre) and a surface plot of $p_d(\mathbf{t})$ (right). The manifold $\mathcal{M} = \mathbf{f}([-a, b])$ is linear because the mapping $\mathbf{f}$ is linear, and is represented by the thick solid segment in the centre graphs. Observe how the maxima of the induced density are close, but do not coincide, with the ends of the segment (represented by the circular points), due to the noise added. For this example, the actual parameters had the values: $a = \frac{1}{3}$, $b = \frac{2}{3}$, $\mathbf{u} = (-1, 2)^T$, $\mathbf{v} = (0, 0)^T$ and $\mathbf{\Sigma} = \frac{1}{20}\,\mathbf{I}$ (where $\mathbf{I}$ is the identity matrix). The second and third rows show the effect of very low noise ($\mathbf{\Sigma} = \frac{1}{2000}\,\mathbf{I}$) and very high noise ($\mathbf{\Sigma} = \frac{5}{4}\,\mathbf{I}$), respectively. The "pyramids" in the second row are visual artifacts produced by the mesh being too coarse.

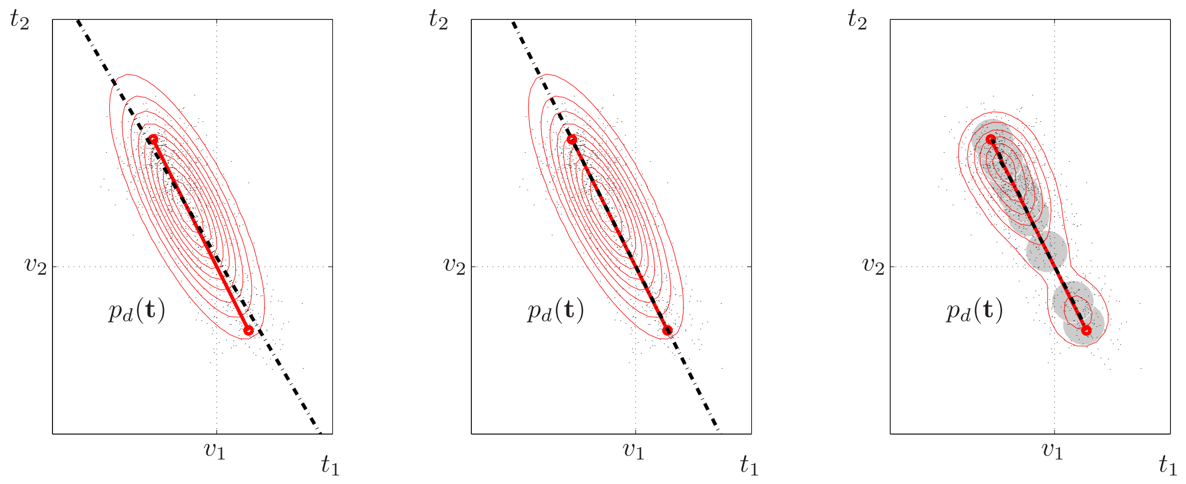


Figure 2.16: Models for the example of section 2.11 (from left to right): factor analysis ($L = 1$ factor), principal component analysis ($L = 1$ principal component) and GTM ($K = 11$ latent grid points, $F = 5$ radial basis functions with a standard deviation $s = 1$ times their separation, EM starting point: the line of the first principal component). For each model, the thick dashed line corresponds to the manifold $\mathcal{M} = \mathbf{f}([-a, b])$ (which in GTM's case overlaps the true one, represented by the thick solid segment). The grey circles in GTM's graph represent the K Gaussian components associated to the latent grid points (eqs. (2.42) and (2.43), with a radius σ). The training set contained $N = 10\,000$ points.

Changing variables to $A = \mathbf{u}^T\mathbf{\Sigma}^{-1}\mathbf{u}$, $B = -2(\mathbf{t}-\mathbf{v})^T\mathbf{\Sigma}^{-1}\mathbf{u}$, $C = (\mathbf{t}-\mathbf{v})^T\mathbf{\Sigma}^{-1}(\mathbf{t}-\mathbf{v})$, $m = \frac{-B}{2A}$ and $\sigma = A^{-1/2}$ we obtain:

$$p_d(\mathbf{t}) = \frac{2}{a^2+b^2}\frac{1}{\sqrt{|2\pi\mathbf{\Sigma}|}}\int_{-a}^{b}|x|\,e^{-\frac{1}{2}(Ax^2+Bx+C)}\,dx = \frac{2}{a^2+b^2}\frac{1}{\sqrt{|2\pi\mathbf{\Sigma}|}}\int_{-a}^{b}|x|\,e^{-\frac{1}{2}\left[\left(\frac{x-m}{\sigma}\right)^2+\left(\frac{m}{\sigma}\right)^2-C\right]}\,dx$$

and, changing again to $\alpha = \frac{a+m}{\sigma\sqrt{2}}$, $\beta = \frac{b-m}{\sigma\sqrt{2}}$ and $\gamma = \frac{m}{\sigma\sqrt{2}}$, we finally obtain:

$$p_d(\mathbf{t}) = \frac{2}{a^2+b^2}\frac{\sigma^2}{\sqrt{|2\pi\mathbf{\Sigma}|}}e^{\gamma^2-\frac{C}{2}}\left\{-e^{-\alpha^2}-e^{-\beta^2}+2e^{-\gamma^2}+\gamma\sqrt{\pi}\left(\operatorname{erf}(\beta)-\operatorname{erf}(\alpha)+2\operatorname{erf}(\gamma)\right)\right\},$$

where $\operatorname{erf}(x) = \frac{2}{\sqrt{\pi}}\int_0^x e^{-t^2}\,dt$ is the error function, which does not have a closed-form expression. α, β and γ are functions of $\mathbf{t}$ and σ depends on the noise and mapping but not on $\mathbf{t}$.

Figure 2.15 shows the prior in latent space $p_l(x)$ and the induced distribution in data space $p_d(\mathbf{t})$ for the true distribution, as well as the effects of low and high noise. Figure 2.16 shows the data space distribution for factor analysis, principal component analysis and GTM. Figure 2.17 comparatively shows the distribution in data space along the manifold segment.

Even though for this problem the true manifold in data space is linear, the distribution $p_d(\mathbf{t})$ in data space is not normal because the prior distribution $p_l(x)$ in latent space is not normal. Thus, the linear-normal models (factor analysis and principal component analysis) produce a bad model while GTM models the true distribution very well. It is interesting to observe that both factor analysis and PCA produce exactly the same normal distribution $p_d(\mathbf{t})$, but while the PCA manifold $\mathcal{M}_{\mathrm{PCA}}$ is aligned with the Gaussian's principal axis, the factor analysis manifold $\mathcal{M}_{\mathrm{FA}}$ is not. The reasons are:

- In this example, where $D = 2$ and $L = 1$, the number of free parameters for factor analysis is $D(L+1) = 4$ and for principal component analysis $DL+1 = 3$. Since the covariance matrix of a bidimensional normal distribution is determined by only $\frac{D(D+1)}{2} = 3$ parameters, both models will account exactly for (and coincide with) the sample covariance matrix—which is the best they can do. Furthermore, since the factor analysis model has more parameters than needed—rotation constraints are not possible because the latent space is one-dimensional—it is undertermined (i.e., non-identifiable): there is an infinite number of equivalent combinations for the values of $\mathbf{\Lambda}$ and $\mathbf{\Psi}$ (one of them being the PCA solution).

- For factor analysis, the uniqueness matrix $\mathbf{\Psi}$ is not isotropic in general (except when it coincides with the PCA solution), as happens for the maximum likelihood estimate found in fig. 2.16, where $\psi_1 < \psi_2$. Thus

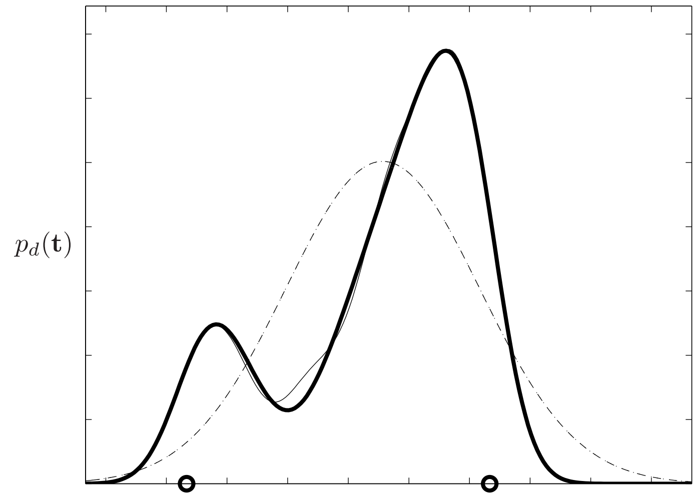


Figure 2.17: Distribution in data space along the manifold segment for the true distribution (thick, solid line) and for each model: factor analysis (dashed line), principal component analysis (dotted line, overlapping with the factor analysis one) and GTM (solid line, almost overlapping with the true one). The circles correspond to the location of the segment ends.

the principal axis of the Gaussian is not parallel to the $\mathbf{\Lambda}$ vector of eq. (2.14). For principal component analysis, the uniqueness matrix $\sigma^2\mathbf{I}$ is isotropic and therefore the principal axis of the Gaussian and the $\mathbf{U}_L$ vector of section 2.6.2 are parallel.

In high-dimensional situations, where factor analysis is identifiable (up to orthogonal rotations), factor analysis will be a better model than PCA, as has been discussed by various researchers (e.g. Hinton et al. 1997; Neal and Dayan 1997). Factor analysis attempts to model the underlying mapping and separately for each variable the noise, while PCA forces the mapping to be aligned with the sample covariance—but the principal component is not directed along the mapping if the noise level is high in another direction.

2.12 Mathematical appendix

2.12.1 Linear-normal models

Theorem 2.12.1. *Consider random variables* $\mathbf{x} \overset{\text{def}}{\sim} \mathcal{N}_L(\boldsymbol{\mu}_X, \mathbf{\Sigma}_X)$ *in* $\mathbb{R}^L$ *and* $\mathbf{t}|\mathbf{x} \overset{\text{def}}{\sim} \mathcal{N}_D(\mathbf{\Lambda}\mathbf{x} + \boldsymbol{\mu}_T, \mathbf{\Sigma}_T)$ *in* $\mathbb{R}^D$, *for symmetric positive definite matrices* $\mathbf{\Sigma}_X$, $\mathbf{\Sigma}_T$, *a* $D \times L$ *matrix* $\mathbf{\Lambda}$ *and vectors* $\boldsymbol{\mu}_X \in \mathbb{R}^L$, $\boldsymbol{\mu}_T \in \mathbb{R}^D$. *Then:*

$$\begin{pmatrix}\mathbf{t}\\ \mathbf{x}\end{pmatrix} \sim \mathcal{N}_{D+L}\left(\begin{pmatrix}\boldsymbol{\mu}\\ \boldsymbol{\mu}_X\end{pmatrix}, \begin{pmatrix}\mathbf{\Sigma} & \mathbf{\Lambda}\mathbf{\Sigma}_X\\ \mathbf{\Sigma}_X\mathbf{\Lambda}^T & \mathbf{\Sigma}_X\end{pmatrix}\right) \qquad \mathbf{t} \sim \mathcal{N}_D(\boldsymbol{\mu}, \mathbf{\Sigma}) \qquad \mathbf{x}|\mathbf{t} \sim \mathcal{N}_L(\hat{\boldsymbol{\mu}}_X, \hat{\mathbf{\Sigma}}_X^{-1}) \tag{2.54}$$

where

$$\boldsymbol{\mu} = \mathbf{\Lambda}\boldsymbol{\mu}_X + \boldsymbol{\mu}_T \tag{2.55}$$
$$\mathbf{\Sigma} = \mathbf{\Sigma}_T + \mathbf{\Lambda}\mathbf{\Sigma}_X\mathbf{\Lambda}^T = \mathbf{\Sigma}_T(\mathbf{\Sigma}_T - \mathbf{\Lambda}\hat{\mathbf{\Sigma}}_X^{-1}\mathbf{\Lambda}^T)^{-1}\mathbf{\Sigma}_T \tag{2.56}$$
$$\hat{\boldsymbol{\mu}}_X = \mathbf{A}(\mathbf{t} - \boldsymbol{\mu}) + \boldsymbol{\mu}_X = \mathbf{A}(\mathbf{t} - \boldsymbol{\mu}_T) + \hat{\mathbf{\Sigma}}_X^{-1}\mathbf{\Sigma}_X^{-1}\boldsymbol{\mu}_X \tag{2.57}$$
$$\mathbf{A} = \mathbf{\Sigma}_X\mathbf{\Lambda}^T\mathbf{\Sigma}^{-1} = \hat{\mathbf{\Sigma}}_X^{-1}\mathbf{\Lambda}^T\mathbf{\Sigma}_T^{-1} \tag{2.58}$$
$$\hat{\mathbf{\Sigma}}_X = \mathbf{\Sigma}_X^{-1} + \mathbf{\Lambda}^T\mathbf{\Sigma}_T^{-1}\mathbf{\Lambda} = \mathbf{\Sigma}_X^{-1}(\mathbf{I}_L - \mathbf{\Sigma}_X\mathbf{\Lambda}^T\mathbf{\Sigma}^{-1}\mathbf{\Lambda})^{-1}. \tag{2.59}$$

Proof. From $p(\mathbf{t}, \mathbf{x}) = p(\mathbf{t}|\mathbf{x})p(\mathbf{x})$ and

$$p(\mathbf{t}|\mathbf{x}) = |2\pi\mathbf{\Sigma}_T|^{-1/2}\, e^{-\frac{1}{2}(\mathbf{t}-\mathbf{\Lambda}\mathbf{x}-\boldsymbol{\mu}_T)^T\mathbf{\Sigma}_T^{-1}(\mathbf{t}-\mathbf{\Lambda}\mathbf{x}-\boldsymbol{\mu}_T)}$$
$$p(\mathbf{x}) = |2\pi\mathbf{\Sigma}_X|^{-1/2}\, e^{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_X)^T\mathbf{\Sigma}_X^{-1}(\mathbf{x}-\boldsymbol{\mu}_X)}$$

algebraic manipulation gives:

$$(\mathbf{t}-\mathbf{\Lambda x}-\boldsymbol{\mu}_T)^T\mathbf{\Sigma}_T^{-1}(\mathbf{t}-\mathbf{\Lambda x}-\boldsymbol{\mu}_T)+(\mathbf{x}-\boldsymbol{\mu}_X)^T\mathbf{\Sigma}_X^{-1}(\mathbf{x}-\boldsymbol{\mu}_X)=\begin{pmatrix}\mathbf{t}-\boldsymbol{\mu}\\ \boldsymbol{\mu}_X\end{pmatrix}^T\begin{pmatrix}\mathbf{\Sigma}_T^{-1} & -\mathbf{\Sigma}_T^{-1}\mathbf{\Lambda}\\ -\mathbf{\Lambda}^T\mathbf{\Sigma}_T^{-1} & \mathbf{\Sigma}_X^{-1}+\mathbf{\Lambda}^T\mathbf{\Sigma}_T^{-1}\mathbf{\Lambda}\end{pmatrix}\begin{pmatrix}\mathbf{t}-\boldsymbol{\mu}\\ \boldsymbol{\mu}_X\end{pmatrix}$$

with $\boldsymbol{\mu}=\mathbf{\Lambda}\boldsymbol{\mu}_X+\boldsymbol{\mu}_T$. Theorem A.1.1(ii) proves

$$\begin{pmatrix}\mathbf{\Sigma}_T^{-1} & -\mathbf{\Sigma}_T^{-1}\mathbf{\Lambda}\\ -\mathbf{\Lambda}^T\mathbf{\Sigma}_T^{-1} & \mathbf{\Sigma}_X^{-1}+\mathbf{\Lambda}^T\mathbf{\Sigma}_T^{-1}\mathbf{\Lambda}\end{pmatrix}^{-1}=\begin{pmatrix}\mathbf{\Sigma} & \mathbf{\Lambda\Sigma}_X\\ \mathbf{\Sigma}_X\mathbf{\Lambda}^T & \mathbf{\Sigma}_X\end{pmatrix}$$

with $\mathbf{\Sigma}=\mathbf{\Sigma}_T(\mathbf{\Sigma}_T-\mathbf{\Lambda}\hat{\mathbf{\Sigma}}_X^{-1}\mathbf{\Lambda}^T)^{-1}\mathbf{\Sigma}_T$, $\hat{\mathbf{\Sigma}}_X=\mathbf{\Sigma}_X^{-1}+\mathbf{\Lambda}^T\mathbf{\Sigma}_T^{-1}\mathbf{\Lambda}$ and $\mathbf{\Sigma}_X\mathbf{\Lambda}^T\mathbf{\Sigma}^{-1}=\hat{\mathbf{\Sigma}}_X^{-1}\mathbf{\Lambda}^T\mathbf{\Sigma}_T^{-1}$. The Sherman-Morrison-Woodbury formula (A.2) proves first that $\mathbf{\Sigma}=\mathbf{\Sigma}_T+\mathbf{\Lambda\Sigma}_X\mathbf{\Lambda}^T$ and then that $\hat{\mathbf{\Sigma}}_X=\mathbf{\Sigma}_X^{-1}(\mathbf{I}_L-\mathbf{\Sigma}_X\mathbf{\Lambda}^T\mathbf{\Sigma}^{-1}\mathbf{\Lambda})^{-1}$. Theorem A.1.1(i) proves that

$$\begin{vmatrix}\mathbf{\Sigma}_T^{-1} & -\mathbf{\Sigma}_T^{-1}\mathbf{\Lambda}\\ -\mathbf{\Lambda}^T\mathbf{\Sigma}_T^{-1} & \mathbf{\Sigma}_X^{-1}+\mathbf{\Lambda}^T\mathbf{\Sigma}_T^{-1}\mathbf{\Lambda}\end{vmatrix}=|\mathbf{\Sigma}_X|^{-1}|\mathbf{\Sigma}_T|^{-1}\quad\text{and}\quad\begin{vmatrix}\mathbf{\Sigma} & \mathbf{\Lambda\Sigma}_X\\ \mathbf{\Sigma}_X\mathbf{\Lambda}^T & \mathbf{\Sigma}_X\end{vmatrix}=|\mathbf{\Sigma}|\,|\hat{\mathbf{\Sigma}}_X|^{-1}=|\mathbf{\Sigma}_X|\,|\mathbf{\Sigma}_T|\,.$$

Finally, theorem A.3.1(iv) and the previous results prove that $\mathbf{x}|\mathbf{t}\sim\mathcal{N}(\hat{\boldsymbol{\mu}}_X,\hat{\mathbf{\Sigma}}_X^{-1})$ with $\hat{\boldsymbol{\mu}}_X$ as defined above. □

Corollary 2.12.2 (Isotropic zero-noise limit of theorem 2.12.1). *In the same conditions of theorem 2.12.1, when $\mathbf{\Sigma}_T=k\mathbf{I}$ and $k\to 0^+$ then:*

$$\begin{pmatrix}\mathbf{t}\\ \mathbf{x}\end{pmatrix}\to\mathcal{N}_{D+L}\left(\begin{pmatrix}\boldsymbol{\mu}\\ \boldsymbol{\mu}_X\end{pmatrix},\begin{pmatrix}\mathbf{\Sigma} & \mathbf{\Lambda\Sigma}_X\\ \mathbf{\Sigma}_X\mathbf{\Lambda}^T & \mathbf{\Sigma}_X\end{pmatrix}\right)\qquad\mathbf{t}\to\mathcal{N}_D(\boldsymbol{\mu},\mathbf{\Sigma})\qquad\mathbf{x}|\mathbf{t}\to\delta_L(\hat{\boldsymbol{\mu}}_X)$$

where

$$\boldsymbol{\mu}=\mathbf{\Lambda}\boldsymbol{\mu}_X+\boldsymbol{\mu}_T \tag{2.60}$$
$$\mathbf{\Sigma}=\mathbf{\Lambda\Sigma}_X\mathbf{\Lambda}^T \tag{2.61}$$
$$\hat{\boldsymbol{\mu}}_X=\mathbf{A}(\mathbf{t}-\boldsymbol{\mu}_T) \tag{2.62}$$
$$\mathbf{A}=\mathbf{A}^+=(\mathbf{\Lambda}^T\mathbf{\Lambda})^{-1}\mathbf{\Lambda} \tag{2.63}$$
$$\hat{\mathbf{\Sigma}}_X\to\infty. \tag{2.64}$$

The normal distributions above are degenerate in the space of $\mathbf{t}$: only those data points $\mathbf{t}\in\operatorname{span}\{\mathbf{\Lambda}\}=\operatorname{im}\mathbf{f}$ have nonzero density.

The following theorem proves that, for linear-normal models, the posterior distribution in latent space $\mathbf{x}|\mathbf{t}\sim\mathcal{N}_L(\hat{\boldsymbol{\mu}}_X,\hat{\mathbf{\Sigma}}_X^{-1})$ is always narrower than the prior distribution $\mathbf{x}\sim\mathcal{N}_L(\boldsymbol{\mu}_X,\mathbf{\Sigma}_X)$, the narrower the smaller the noise is: $\mathbf{\Sigma}_T\to\mathbf{0}\Rightarrow\hat{\mathbf{\Sigma}}_X^{-1}\to\mathbf{0}$.

Theorem 2.12.3. *In the same conditions of theorem 2.12.1, $\left|\hat{\mathbf{\Sigma}}_X^{-1}\right|<|\mathbf{\Sigma}_X|$.*

Proof. We have that:

- $\mathbf{\Lambda}^T\mathbf{\Sigma}_T^{-1}\mathbf{\Lambda}$ is positive definite because, for any $\mathbf{x}\neq\mathbf{0}$, $\mathbf{x}^T\mathbf{\Lambda}^T\mathbf{\Sigma}_T^{-1}\mathbf{\Lambda x}=\mathbf{y}^T\mathbf{\Sigma}_T^{-1}\mathbf{y}>0$, since $\mathbf{\Sigma}_T$ is positive definite and $\mathbf{y}\stackrel{\text{def}}{=}\mathbf{\Lambda x}\neq\mathbf{0}$ (since $\mathbf{\Lambda}$ is full-rank).
- $\mathbf{\Lambda}^T\mathbf{\Sigma}_T^{-1}\mathbf{\Lambda\Sigma}_X$ is positive definite because the product of positive definite matrices is positive definite.
- If $\mathbf{B}$ is positive definite then $|\mathbf{I}+\mathbf{B}|>1$, since decomposing spectrally $\mathbf{B}=\mathbf{UVU}^T$ with $\mathbf{V}>0$ diagonal and $\mathbf{U}$ orthogonal: $|\mathbf{I}+\mathbf{B}|=\left|\mathbf{UU}^T+\mathbf{UVU}^T\right|=|\mathbf{U}|\,|\mathbf{I}+\mathbf{V}|\,\left|\mathbf{U}^T\right|=|\mathbf{I}+\mathbf{V}|=\prod_{l=1}^L(1+v_l)>1$.

Hence $|\hat{\mathbf{\Sigma}}_X|\,|\mathbf{\Sigma}_X|=|\hat{\mathbf{\Sigma}}_X\mathbf{\Sigma}_X|=\left|\mathbf{I}+\mathbf{\Lambda}^T\mathbf{\Sigma}_T^{-1}\mathbf{\Lambda\Sigma}_X\right|>1\Rightarrow\left|\hat{\mathbf{\Sigma}}_X^{-1}\right|<|\mathbf{\Sigma}_X|$. □

2.12.2 Independence relations

Theorem 2.12.4 (Pairwise local independence). $p(t_i t_j|\mathbf{x}) = p(t_i|\mathbf{x})p(t_j|\mathbf{x})\ \forall i,j \in \{1,\dots,D\}$.

Proof. Call $\mathcal{I} = \{1,\dots,D\} \setminus \{i,j\}$. Then:

$$p(t_i t_j|\mathbf{x}) = \int p(\mathbf{t}|x)\,d\mathbf{t}_{\mathcal{I}} = \int \prod_{d\in\{1,\dots,D\}} p(t_d|\mathbf{x})\,d\mathbf{t}_{\mathcal{I}} = p(t_i|\mathbf{x})p(t_j|\mathbf{x}) \int p(\mathbf{t}_{\mathcal{I}}|\mathbf{x})\,d\mathbf{t}_{\mathcal{I}} = p(t_i|\mathbf{x})p(t_j|\mathbf{x})$$

where we have used the axiom of local independence (2.4). □

Definition 2.12.1 and theorems 2.12.5 and 2.12.6 are from Cover and Thomas (1991).

Definition 2.12.1. The random variables X, Y and Z form a Markov chain $X \to Y \to Z$ in that order if the conditional distribution of Z depends only on Y and is conditionally independent of X, i.e., $p(x,y,z) = p(x)p(y|x)p(z|y) = p(x,y)p(z|y)$.

Theorem 2.12.5. *$X \to Y \to Z$ if and only if X and Z are conditionally independent given Y.*

Proof. $p(x,z|y) = \frac{p(x,y,z)}{p(y)} = \frac{p(x,y)p(z|y)}{p(y)} = p(x|y)p(z|y)$. □

Theorem 2.12.6. *If $X \to Y \to Z$ then:*

(i) $X \to Y \to Z \Rightarrow Z \to Y \to X$.

(ii) $I(X;Z) \le I(X;Y)$: no clever manipulation of the data Y (deterministic or random) can improve the inferences that can be made from the data, i.e., the information that Y contains about X (data processing inequality).

(iii) $I(X;Y|Z) \le I(X;Y)$: the dependency of X and Y is decreased, or remains unchanged, by observation of a downstream random variable Z. Note, though, that $I(X;Y|Z) > I(X;Y)$ may happen if X, Y and Z do not form a Markov chain.

Theorem 2.12.7 (Pairwise Markov chains). *$t_i \to \mathbf{x} \to t_j\ \forall i,j \in \{1,\dots,D\}$, i.e., there is a Markov chain between any two observed variables via the latent ones.*

Proof. From theorems 2.12.4 and 2.12.5. □

2.12.3 Latent variable models and entropy

The following results are easily proven using theorem 2.12.4 and the information theory results from section A.4.

Theorem 2.12.8.

- $I(t_i;t_j|\mathbf{x}) = 0$.
- $h(t_i|t_j,\mathbf{x}) = h(t_i|\mathbf{x})\ \forall i,j \in \{1,\dots,D\}$.
- $h(t_i|\mathbf{x}) + h(t_j|\mathbf{x}) = h(t_i,t_j|\mathbf{x})\ \forall i,j \in \{1,\dots,D\}$.
- $h(\mathbf{t}|\mathbf{x}) = \sum_{d=1}^{D} h(t_d|\mathbf{x})$.
- $h(\mathbf{t},\mathbf{x}) = h(\mathbf{t}) + h(\mathbf{x}|\mathbf{t}) = h(\mathbf{x}) + h(\mathbf{t}|\mathbf{x}) = h(\mathbf{x}) + \sum_{d=1}^{D} h(t_d|\mathbf{x})$. *If the latent variables are mutually independent,* $h(\mathbf{t},\mathbf{x}) = \sum_{l=1}^{L} h(x_l) + \sum_{d=1}^{D} h(t_d|\mathbf{x})$.
- $0 \le I(t_i;\mathbf{x}|t_j) \le I(t_i;t_j) \le \min\left(I(t_i;\mathbf{x}), I(t_j;\mathbf{x})\right) \le \min\left(h(\mathbf{x}), h(t_i), h(t_j)\right)\ \forall i,j \in \{1,\dots,D\}$.

For models where the prior distribution in latent space has been sampled, $p(\mathbf{x}) = \sum_{k=1}^{K} p(\mathbf{x}_k)\delta(\mathbf{x} - \mathbf{x}_k)$, the distributions $p(\mathbf{t})$, $p(\mathbf{x}|\mathbf{t})$ and $p(\mathbf{t}_{\mathcal{I}}|\mathbf{t}_{\mathcal{J}})$ are finite mixtures, whose entropy cannot be computed analytically. However, we can give bounds for it.

Theorem 2.12.9. *Whether the prior distribution in latent space is continuous or has been sampled, if $\mathbf{\Psi}$ is independent of $\mathbf{x}$ and $\mathbf{t}|\mathbf{x} \sim \mathcal{N}(\mathbf{f}(\mathbf{x}), \mathbf{\Psi})$, then $h(\mathbf{t}|\mathbf{x}) = \frac{1}{2}\ln|2\pi e\mathbf{\Psi}|$.*

Proof.

- Continuous $p(\mathbf{x})$: $h(\mathbf{t}|\mathbf{x}) \stackrel{\text{def}}{=} -\int_{\mathbb{R}^L} p(\mathbf{x}) \int_{\mathbb{R}^D} p(\mathbf{t}|\mathbf{x}) \ln p(\mathbf{t}|\mathbf{x})\, d\mathbf{t}\, d\mathbf{x} = \int_{\mathbb{R}^L} p(\mathbf{x}) \frac{1}{2} \ln |2\pi e \boldsymbol{\Psi}|\, d\mathbf{x} = \frac{1}{2} \ln |2\pi e \boldsymbol{\Psi}|$.
- Sampled $p(\mathbf{x})$: $h(\mathbf{t}|\mathbf{x}) \stackrel{\text{def}}{=} -\sum_{k=1}^K p(\mathbf{x}_k) \int_{\mathbb{R}^D} p(\mathbf{t}|\mathbf{x}) \ln p(\mathbf{t}|\mathbf{x})\, d\mathbf{t} = \sum_{k=1}^K p(\mathbf{x}_k) \frac{1}{2} \ln |2\pi e \boldsymbol{\Psi}| = \frac{1}{2} \ln |2\pi e \boldsymbol{\Psi}|$. □

Remark. Although the axiom of local independence prescribes factorised noise models, theorem 2.12.9 holds even if $\boldsymbol{\Psi}$ is not diagonal.

Theorem 2.12.10 (Entropy for linear-normal models). *Consider random variables $\mathbf{x} \sim \mathcal{N}_L(\boldsymbol{\mu}_X, \boldsymbol{\Sigma}_X)$ in $\mathbb{R}^L$ and $\mathbf{t}|\mathbf{x} \sim \mathcal{N}_D(\boldsymbol{\Lambda}\mathbf{x} + \boldsymbol{\mu}_T, \boldsymbol{\Sigma}_T)$ in $\mathbb{R}^D$, for positive definite matrices $\boldsymbol{\Sigma}_X$, $\boldsymbol{\Sigma}_T$, a $D \times L$ matrix $\boldsymbol{\Lambda}$ and vectors $\boldsymbol{\mu}_X \in \mathbb{R}^L$, $\boldsymbol{\mu}_T \in \mathbb{R}^D$. Then, with $\boldsymbol{\Sigma}$ and $\hat{\boldsymbol{\Sigma}}_X$ given as in theorem 2.12.1:*

- $h(\mathbf{x}) = \frac{1}{2} \ln |2\pi e \boldsymbol{\Sigma}_X|$
- $h(\mathbf{t}|\mathbf{x}) = \frac{1}{2} \ln |2\pi e \boldsymbol{\Sigma}_T|$
- $h(\mathbf{t}, \mathbf{x}) = \frac{1}{2} \ln \left(|2\pi e \boldsymbol{\Sigma}_X|\, |2\pi e \boldsymbol{\Sigma}_T|\right)$
- $h(\mathbf{t}) = \frac{1}{2} \ln |2\pi e \boldsymbol{\Sigma}| = \frac{1}{2} \ln \left| 2\pi e (\boldsymbol{\Sigma}_T + \boldsymbol{\Lambda}\boldsymbol{\Sigma}_X\boldsymbol{\Lambda}^T) \right|$
- $h(\mathbf{x}|\mathbf{t}) = \frac{1}{2} \ln \left| 2\pi e \hat{\boldsymbol{\Sigma}}_X^{-1} \right| = \frac{1}{2} \ln \left(|2\pi e \boldsymbol{\Sigma}_X|\, |2\pi e \boldsymbol{\Sigma}_T|\, |2\pi e \boldsymbol{\Sigma}|^{-1} \right)$.

Theorem 2.12.11 (Entropy for factor analysis). *If $\mathbf{x} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_L)$ and $\mathbf{t}|\mathbf{x} \sim \mathcal{N}(\boldsymbol{\Lambda}\mathbf{x} + \boldsymbol{\mu}, \boldsymbol{\Psi})$ with $\boldsymbol{\Psi}$ diagonal and independent of $\mathbf{x}$, then:*

- $h(\mathbf{x}) = \frac{L}{2} \ln (2\pi e)$
- $h(\mathbf{t}|\mathbf{x}) = \frac{1}{2} \ln |2\pi e \boldsymbol{\Psi}|$
- $h(\mathbf{t}, \mathbf{x}) = \frac{1}{2} \ln \left((2\pi e)^{D+L} |\boldsymbol{\Psi}| \right)$
- $h(\mathbf{t}) = \frac{1}{2} \ln \left((2\pi e)^D \left| \boldsymbol{\Lambda}\boldsymbol{\Lambda}^T + \boldsymbol{\Psi} \right| \right)$
- $h(\mathbf{x}|\mathbf{t}) = \frac{1}{2} \ln \left((2\pi e)^L \left| \boldsymbol{\Lambda}^T \boldsymbol{\Psi}^{-1} \boldsymbol{\Lambda} + \mathbf{I}_L \right|^{-1} \right)$.

Theorem 2.12.12 (Entropy for principal component analysis). *If $\mathbf{x} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_L)$ and $\mathbf{t}|\mathbf{x} \sim \mathcal{N}(\boldsymbol{\Lambda}\mathbf{x} + \boldsymbol{\mu}, \sigma^2 \mathbf{I}_D)$ with σ independent of $\mathbf{x}$, then:*

- $h(\mathbf{x}) = \frac{L}{2} \ln (2\pi e)$
- $h(\mathbf{t}|\mathbf{x}) = \frac{D}{2} \ln (2\pi e \sigma^2)$
- $h(\mathbf{t}, \mathbf{x}) = \frac{1}{2} \ln \left((2\pi e)^{D+L} \sigma^{2D} \right)$
- $h(\mathbf{t}) = \frac{1}{2} \ln \left((2\pi e)^D \left| \boldsymbol{\Lambda}\boldsymbol{\Lambda}^T + \sigma^2 \mathbf{I}_D \right| \right)$
- $h(\mathbf{x}|\mathbf{t}) = \frac{1}{2} \ln \left((2\pi e \sigma^2)^L \left| \boldsymbol{\Lambda}^T \boldsymbol{\Lambda} + \sigma^2 \mathbf{I}_L \right|^{-1} \right)$.

Theorem 2.12.13 (Entropy for GTM). *If $p(\mathbf{x}) = \frac{1}{K} \sum_{k=1}^K \delta(\mathbf{x} - \mathbf{x}_k)$ and $\mathbf{t}|\mathbf{x} \sim \mathcal{N}(\mathbf{f}(\mathbf{x}), \sigma^2 \mathbf{I}_D)$ with σ independent of $\mathbf{x}$ and $\mathbf{f}$ a generalised linear model, then:*

- $h(\mathbf{x}) = \ln K$
- $h(\mathbf{t}|\mathbf{x}) = \frac{D}{2} \ln (2\pi e \sigma^2)$
- $h(\mathbf{t}, \mathbf{x}) = \frac{1}{2} \ln \left(K^2 (2\pi e \sigma^2)^D \right)$.

Theorems 2.12.9–2.12.13 show that, for normal noise models with fixed covariance $\boldsymbol{\Psi}$, the entropy of the observed variables, $h(\mathbf{t})$, is bounded below by the entropy of the normal distribution of covariance $\boldsymbol{\Psi}$. This means that we cannot have a distribution $p(\mathbf{t})$ in observed space which is, loosely speaking, more peaked than a $\mathcal{N}(\cdot, \boldsymbol{\Psi})$. Since it is to be expected that real distributions will have variable "peakiness" depending on the region considered, the optimal $\boldsymbol{\Psi}$ will be a compromise between the covariance in areas of large noise and in areas of small noise. In GTM this could be overcome by having many points in latent space (high K) and a small covariance $\boldsymbol{\Psi} = \sigma^2 \mathbf{I}_D$. However, allowing the covariance $\boldsymbol{\Psi}$ to depend on $\mathbf{x}$ (like the mean of the noise model does, $\mathbf{f}(\mathbf{x})$) would be the obvious workaround.

2.12.4 Diagonal GTM (dGTM)

We give here the details for dGTM, the diagonal noise model for GTM that we proposed in section 2.6.5.1. First let us fully generalise GTM so that it has diagonal noise dependent on each latent grid point and both the values of the noise covariance matrix and the values of the prior distribution in latent space are trainable. Thus

$$\mathbf{t}|\mathbf{x} \stackrel{\text{def}}{\sim} \mathcal{N}(\mathbf{f}(\mathbf{x}), \mathbf{\Psi}(\mathbf{x})) \tag{2.42'}$$

with $\mathbf{\Psi} : \mathcal{X} \rightarrow (\mathbb{R}^+)^D$ and

$$p(\mathbf{t}) = \sum_{k=1}^{K} \pi_k p(\mathbf{t}|\mathbf{x}_k)$$

with $\pi_k \stackrel{\text{def}}{=} p(\mathbf{x}_k)$ and $\mathbf{\Psi}_k \stackrel{\text{def}}{=} \mathbf{\Psi}(\mathbf{x}_k) = \operatorname{diag}(\psi_{k1}, \dots, \psi_{kD})$. The dependence of $p(\mathbf{t}|\mathbf{x}_k)$ on all the relevant parameters is not explicitly written for clarity of notation. The EM equations are derived by minimising

$$\sum_{n=1}^{N} \ln p(\mathbf{t}_n) - \lambda \left(\sum_{k=1}^{K} \pi_k - 1 \right)$$

where λ is a Lagrange multiplier that ensures that the values of the prior distribution in latent space add to one. This gives

$$\pi_k^{(\tau+1)} = \frac{1}{N} \sum_{n=1}^{N} p(\mathbf{x}_k|\mathbf{t}_n) = \frac{1}{N} \sum_{n=1}^{N} R_{nk}^{(\tau)}$$

for the prior distribution in latent space, using the responsibilities $R_{nk} = p(\mathbf{x}_k|\mathbf{t}_n)$, and for the rest of the parameters we obtain

$$\sum_{n=1}^{N} \sum_{k=1}^{K} R_{nk} \frac{\partial}{\partial \theta} \ln p(\mathbf{t}_n|\mathbf{x}_k) = 0 \tag{2.65}$$

where θ represents a parameter w_{df} or ψ_{kd}. Recalling that $f_d(\mathbf{x}_k) = \sum_{f=1}^{F} w_{df} \phi_{kf}$ where $\phi_{kf} = \phi_f(\mathbf{x}_k)$ and plugging

$$\frac{\partial}{\partial w_{df}} \ln p(\mathbf{t}_n|\mathbf{x}_k) = \frac{\phi_{kf}}{\psi_{kd}} \left(t_{nd} - \sum_{f'=1}^{F} w_{df'} \phi_{kf'} \right) \qquad d = 1, \dots, D \quad f = 1, \dots, F$$

$$\frac{\partial}{\partial \psi_{kd}} \ln p(\mathbf{t}_n|\mathbf{x}_k) = -\frac{1}{2} \left(\frac{1}{\psi_{kd}} - \frac{(t_{nd} - f_d(\mathbf{x}_k))^2}{\psi_{kd}^2} \right) \qquad k = 1, \dots, K \quad d = 1, \dots, D$$

$$\frac{\partial}{\partial \psi_{kd}} \ln p(\mathbf{t}_n|\mathbf{x}_{k'}) = 0 \text{ if } k \neq k'$$

into (2.65), we obtain a system of $D(F + K)$ equations that, together with the equations for $\{\pi_k\}_{k=1}^{K}$, define the M step:

$$\sum_{k=1}^{K} \frac{\phi_{kf}}{\psi_{kd}} g_{kk} \sum_{f'=1}^{F} \phi_{kf'} w_{df'} = \sum_{n=1}^{N} \sum_{k=1}^{K} \frac{\phi_{kf}}{\psi_{kd}} R_{nk} t_{nd} \qquad d = 1, \dots, D \quad f = 1, \dots, F$$

$$\psi_{kd} = \frac{1}{g_{kk}} \sum_{n=1}^{N} R_{nk} (t_{nd} - f_d(\mathbf{x}_k))^2 \qquad k = 1, \dots, K \quad d = 1, \dots, D$$

where $g_{kk} \stackrel{\text{def}}{=} \sum_{n=1}^{N} R_{nk}$ as in section 2.6.5. Solving for $\{w_{df}\}_{d,f=1}^{D,F}$ is cumbersome, since the first group of equations cannot be put into a nice matrix equation form. The second group of equations shows that $\mathbf{\Psi}_k$ is a weighted average of the squared componentwise deviations of the data points from the "reference point" $\mathbf{f}(\mathbf{x}_k)$, where the weights are given by the responsibilities (normalised over all reference points).

Compared to the standard GTM model, this extended version has $K(D+1)-2$ more parameters: $\{\pi_k\}_{k=1}^{K-1}$ and $\{\mathbf{\Psi}_k\}_{k=1}^{K}$. Thus, we lose the attractive property that using a large number of latent points K (exponentially dependent on the latent space dimension L) does not increase the number of parameters. Even if the EM algorithm was straightforward, such a model would require a large training set for stable statistical estimation[41]

[41]Note, however, that the standard GTM model does depend on F, the number of radial basis functions needed for the mapping $\mathbf{f}$, which also depends exponentially on L (but can be kept quite smaller than K without losing approximation power).

and would be prone to singularity problems ($\mathbf{\Psi}_k \to \mathbf{0}$ for some k), as mentioned in section 2.5.2, among other places. Defining $\mathbf{\Psi}(\mathbf{x})$ as a generalised linear model (as $\mathbf{f}$ is, eq. (2.41), and reusing the $\boldsymbol{\phi}$ function) eliminates the direct dependence on K and ensures a smooth variability of the noise variance over the data manifold, but the resulting M step is unsolvable, as can be readily checked. Also, since $\mathbf{f}$ is a universal approximator, it is not really necessary to make the prior distribution in latent space trainable as well.

Taking into account these considerations, for all $k = 1, \dots, K$ we keep $p(\mathbf{x}_k) = \frac{1}{K}$ constant as in the standard GTM model and the noise model covariance matrix $\mathbf{\Psi}_k = \mathbf{\Psi}$ diagonal constant too. As can easily be seen by recalculating $\frac{\partial}{\partial \psi_d} \ln p(\mathbf{t}_n|\mathbf{x}_k)$, the M step becomes exactly solvable and leads to the following update equations for the parameters $\mathbf{W}$ and $\mathbf{\Psi}$, respectively:

$$\mathbf{\Phi}^T \mathbf{G}^{(\tau)} \mathbf{\Phi} (\mathbf{W}^{(\tau+1)})^T = \mathbf{\Phi}^T (\mathbf{R}^{(\tau)})^T \mathbf{T} \tag{2.45a}$$

$$\psi_d^{(\tau+1)} = \frac{1}{N} \sum_{k=1}^{K} \sum_{n=1}^{N} R_{nk}^{(\tau)} (t_{nd} - f_d(\mathbf{x}_k))^2 \quad d = 1, \dots, D \tag{2.45b'}$$

the first of which is the same one as for the standard GTM model and the second of which has the same interpretation as before as weighted average of the squared componentwise deviations of the data points from $\mathbf{f}(\mathbf{x}_k)$. Further modifications can still be done, such as a Gaussian regularisation term on $\mathbf{W}$, eq. (2.46), or a unit-speed constraint on $\mathbf{f}$, section 2.8.3.

Chapter 3

Some properties of finite mixtures of multivariate Bernoulli distributions

The objective of this chapter is to prove some properties of the class of finite mixtures of multivariate Bernoulli distributions that are used elsewhere in this thesis and to discuss the identifiability of this class of mixtures and the practical maximum likelihood estimation of its parameters by an EM algorithm.

Finite mixtures of multivariate Bernoulli distributions can be seen as a latent variable model in that the discrete variable that indexes the components is latent (as opposed to the continuous latent variables considered in chapter 2). It corresponds to a latent class model (section 2.1). As such, they have been extensively used in diverse fields (such as bacterial taxonomy) to model a population of binary, multivariate measurements in terms of a few latent classes. Everitt and Hand (1981) and Gyllenberg et al. (1994) give a number of examples.

3.1 Definition and moments

3.1.1 Multivariate Bernoulli distribution

A D-variate Bernoulli distribution of parameter $\mathbf{p} = (p_1, \dots, p_D)^T \in [0,1]^D$, $\mathcal{B}_D(\mathbf{p})$, is defined as:

$$p(\mathbf{t};\mathbf{p}) = \prod_{d=1}^{D} p_d^{t_d}(1-p_d)^{1-t_d} = \prod_{d=1}^{D} p(t_d|\mathcal{B}(p_d)) \tag{3.1}$$

where $\mathcal{B}(p_d)$ is a Bernoulli distribution of parameter p_d, $d = 1, \dots, D$. Thus, the D-variate Bernoulli distribution is equivalent to D independent Bernoulli distributions.

Applying the relation $\operatorname{cov}\{\mathbf{t}\} = \mathrm{E}\left\{(\mathbf{t} - \mathrm{E}\{\mathbf{t}\})(\mathbf{t} - \mathrm{E}\{\mathbf{t}\})^T\right\} = \mathrm{E}\left\{\mathbf{t}\mathbf{t}^T\right\} - \mathrm{E}\{\mathbf{t}\}\,\mathrm{E}\{\mathbf{t}\}^T$, one obtains the moments of the D-variate Bernoulli distribution of parameter $\mathbf{p}$:

$$\text{mean:} \quad \boldsymbol{\mu} = \mathbf{p} \tag{3.2}$$

$$\text{covariance:} \quad \boldsymbol{\Sigma} = \operatorname{diag}(p_d(1-p_d)) \tag{3.3}$$

because the matrix $\mathrm{E}\left\{\mathbf{t}\mathbf{t}^T\right\}$ has elements $p_d p_e$ in the off-diagonal position (d,e) and p_d in the diagonal position (d,d).

3.1.2 Finite mixture of multivariate Bernoulli distributions

A mixture distribution of M D-variate Bernoulli distributions $\mathcal{B}_D(\mathbf{p}_1), \dots, \mathcal{B}_D(\mathbf{p}_M)$ is defined as:

$$p\left(\mathbf{t}; \{\pi_m, \mathbf{p}_m\}_{m=1}^M\right) = \sum_{m=1}^{M} \pi_m p(\mathbf{t}|m) \tag{3.4}$$

where the mixing proportions π_m satisfy $0 < \pi_m < 1$ for $m = 1, \dots, M$ and $\sum_{m=1}^M \pi_m = 1$ (in the case $M = 1$, take $\pi_1 = 1$) and the component distributions are D-variate Bernoulli distributions, $\mathbf{t}|m \sim \mathcal{B}_D(\mathbf{p}_m)$. The parameters $\{\mathbf{p}_m\}_{m=1}^M$ are often called *prototypes*.

This chapter is mainly based on reference Carreira-Perpiñán and Renals (2000).

Applying the relation $\mathrm{E}_{p(\mathbf{t})}\{\mathbf{f}(\mathbf{t})\} = \sum_{m=1}^{M} \pi_m \, \mathrm{E}_{p(\mathbf{t}|m)}\{\mathbf{f}(\mathbf{t})\}$ one obtains the moments for the mixture of M D-variate Bernoulli distributions:

$$\text{mean:} \quad \boldsymbol{\mu} = \sum_{m=1}^{M} \pi_m \boldsymbol{\mu}_m \tag{3.5}$$

$$\text{covariance:} \quad \boldsymbol{\Sigma} = \sum_{m=1}^{M} \pi_m \, \mathrm{E}_{p(\mathbf{t}|m)}\{\mathbf{t}\mathbf{t}^T\} - \boldsymbol{\mu}\boldsymbol{\mu}^T = \sum_{m=1}^{M} \pi_m (\boldsymbol{\Sigma}_m + \boldsymbol{\mu}_m \boldsymbol{\mu}_m^T) - \boldsymbol{\mu}\boldsymbol{\mu}^T \tag{3.6}$$

where for $m = 1, \dots, M$, $\boldsymbol{\mu}_m = \mathbf{p}_m$ and $\boldsymbol{\Sigma}_m = \operatorname{diag}(p_{md}(1 - p_{md}))$ are the component means and covariance matrices, respectively. Expanding $\boldsymbol{\Sigma}$ one obtains:

$$\begin{aligned} (\boldsymbol{\Sigma})_{de} &= \sum_{n>m} \pi_m \pi_n (p_{md} - p_{nd})(p_{me} - p_{ne}) \\ (\boldsymbol{\Sigma})_{dd} &= \mu_d(1 - \mu_d). \end{aligned}$$

Observe that $\boldsymbol{\Sigma}$ is no longer diagonal. Thus a mixture of multivariate Bernoulli distributions can account for correlations between variables.

3.2 Maximum likelihood parameter estimation

This section shows that the EM algorithm for finite mixtures of multivariate Bernoulli distributions is guaranteed to converge to a proper maximum likelihood estimate, owing to a property of the log-likelihood surface.

3.2.1 Lack of proper minima of the log-likelihood surface

We show here that the log-likelihood surface of a finite mixture distribution (*not necessarily a mixture of multivariate Bernoulli distributions*) has no proper minima.

For $M > 1$ integer, let

$$p(\mathbf{t}) = \sum_{m=1}^{M} \pi_m p(\mathbf{t}|m)$$

define a finite mixture distribution of M components (Everitt and Hand, 1981). For each $m = 1, \dots, M$, π_m are the mixing proportions and $p(\mathbf{t}|m)$ the component probability distributions (defined in a D-dimensional space, continuous or discrete), which are parameterised. The mixing proportions verify $0 < \pi_m < 1$ for all $m = 1, \dots, M$ and $\sum_{m=1}^{M} \pi_m = 1$. Let $\boldsymbol{\Theta}$ refer collectively to all the parameters of the mixture, including the mixing proportions. Given a sample $\{\mathbf{t}_n\}_{n=1}^{N}$, the log-likelihood of the parameters $\boldsymbol{\Theta}$ is defined as:

$$\mathcal{L}(\boldsymbol{\Theta}) = \sum_{n=1}^{N} \ln p(\mathbf{t}_n; \boldsymbol{\Theta}) = \sum_{n=1}^{N} \ln \left(\sum_{m=1}^{M} \pi_m p(\mathbf{t}_n|m) \right). \tag{3.7}$$

Proposition 3.2.1. *The log-likelihood surface of a finite mixture distribution has no proper minima.*

Proof. Omitting the dependence on the parameters $\boldsymbol{\Theta}$ for clarity, it is easily seen that:

$$\frac{\partial \mathcal{L}}{\partial \pi_m} = \frac{1}{\pi_m} \sum_{n=1}^{N} p(m|\mathbf{t}_n) - N \qquad m = 1, \dots, M \tag{3.8}$$

and

$$\frac{\partial^2 \mathcal{L}}{\partial \pi_m \partial \pi_{m'}} = -\frac{1}{\pi_m \pi_{m'}} \sum_{n=1}^{N} p(m|\mathbf{t}_n) p(m'|\mathbf{t}_n) \leq 0 \qquad m, m' = 1, \dots, M \tag{3.9}$$

where

$$p(m|\mathbf{t}_n) = \frac{p(\mathbf{t}_n|m)p(m)}{\sum_{m'=1}^{M} p(\mathbf{t}_n|m')p(m')}$$

by Bayes' theorem and the term "$-N$" in eq. (3.8) results from the constraint $\sum_{m=1}^{M} \pi_m = 1$ introduced in the log-likelihood via a Lagrange multiplier.

Therefore, from eq. (3.9), the Hessian $\mathbf{H}$ has negative or null components in its diagonal; call one of these h_{ii}. Then, if $\mathbf{e}_i$ is the ith canonical vector (having all components null except the ith, which is equal to one), we have $\mathbf{e}_i^T\mathbf{H}\mathbf{e}_i = h_{ii} \le 0$ and the Hessian is not definite positive (irrespectively of the sign of the derivatives with respect to the other parameters of the mixture).

The case where $p(m|\mathbf{t}_n) = 0$ for all n and m implies $p(\mathbf{t}_n|m) = 0$ for all n and m, by Bayes' theorem. Therefore $p(\mathbf{t}_n) = \sum_{m=1}^M \pi_m p(\mathbf{t}_n|m) = 0$ for all n and $\mathcal{L}(\mathbf{\Theta}) = -\infty$, which is an improper minimum. It can only happen for distributions that allow $p(\mathbf{t}|m)$ to be zero for certain boundary values of the parameters, such as $p = 0$ or $p = 1$ for a Bernoulli distribution of parameter p.

Hence, no stationary point of the log-likelihood is a (proper) minimum. $\square$

Example. Consider a mixture of $M = 2$ univariate Bernoulli distributions ($D = 1$), as in eq. (3.4). Therefore

$$p(t|\pi_1, p_1, p_2) = \pi_1 p_1^t(1-p_1)^{1-t} + (1-\pi_1)p_2^t(1-p_2)^{1-t} = \begin{cases} \pi_1(1-p_1) + (1-\pi_1)(1-p_2) & \text{if } t = 0 \\ \pi_1 p_1 + (1-\pi_1)p_2 & \text{if } t = 1 \end{cases}$$

with mean

$$\bar{t} \stackrel{\text{def}}{=} \mathrm{E}_{p(t)}\{t\} = p(t = 1|\pi_1, p_1, p_2) = \pi_1 p_1 + (1-\pi_1)p_2.$$

From eq.(3.7), the log-likelihood for a sample of N points ($N - N_1$ zeroes and N_1 ones) can be written as:

$$\mathcal{L}(\pi_1, p_1, p_2) = N\left(\left(1 - \frac{N_1}{N}\right)\ln(1-\bar{t}) + \frac{N_1}{N}\ln\bar{t}\right) \tag{3.10}$$

where $\frac{N_1}{N}$ is the sample mean and the model mean $\bar{t}$ is a function of the parameters (π_1, p_1 and p_2). Thus, the log-likelihood function (3.10) will be constant on the surface $\bar{t} = \pi_1 p_1 + (1-\pi_1)p_2 = k$ for any $k \in [0,1]$. Any point on that surface corresponds to a model with mean $\bar{t} = k$ which produces the same distribution, given by $p(t = 0) = 1 - k$ and $p(t = 1) = k$, due to the non-identifiability of the class of finite mixtures of multivariate Bernoulli distributions (see section 3.3). By taking the derivative of the log-likelihood function (3.10) with respect to $\bar{t}$ we find that its stationary points are the points of the surface $t = \pi_1 p_1 + (1-\pi_1)p_2 = \frac{N_1}{N}$ for $\pi_1, p_1, p_2 \in [0,1]$. By taking the second derivative, we find that they are all maxima, in accordance with proposition 3.2.1.

Figure 3.1 shows sections of the log-likelihood for constant $\pi_1 = 1 - \pi_2$ in the space $p_1 \times p_2$. The maximum likelihood surface $\pi_1 p_1 + (1-\pi_1)p_2 = \frac{N_1}{N}$ becomes a line segment in the square $[0,1] \times [0,1]$ and is plotted as a thick dotted line. Thus, each graph shows a segment of maxima plus two improper minima in the corners $p_1 = p_2 = 0$ or $1 \Leftrightarrow \pi_1 p_1 + (1-\pi_1)p_2 = 0$ or 1 (or the whole lines $p_m = \frac{N1}{N}$ for the degenerate case $\pi_m = 1$), where the log-likelihood goes to $-\infty$.

3.2.2 EM parameter estimation

We assume a fixed number of components M. Maximum likelihood estimation can be achieved by an EM algorithm. Define $\boldsymbol{\pi} = (\pi_1, \dots, \pi_M)^T$ and $\mathbf{P} = (\mathbf{p}_1, \dots, \mathbf{p}_M)$. As in eq. (3.7), the log-likelihood of the parameters $\{\boldsymbol{\pi}, \mathbf{P}\}$ given a sample $\{\mathbf{t}_n\}_{n=1}^N$ is

$$\mathcal{L}(\boldsymbol{\pi}, \mathbf{P}) = \sum_{n=1}^N \ln p(\mathbf{t}_n; \boldsymbol{\pi}, \mathbf{P}) = \sum_{n=1}^N \ln\left(\sum_{m=1}^M \pi_m \prod_{d=1}^D p_{md}^{t_{nd}}(1-p_{md})^{1-t_{nd}}\right) \tag{3.11}$$

and its gradient is easily seen to be

$$\frac{\partial\mathcal{L}}{\partial\pi_m} = \frac{1}{\pi_m}\sum_{n=1}^N p(m|\mathbf{t}_n; \boldsymbol{\pi}, \mathbf{P}) - N \qquad m = 1, \dots, M \tag{3.12}$$

$$\frac{\partial\mathcal{L}}{\partial p_{md}} = \frac{1}{p_{md}(1-p_{md})}\sum_{n=1}^N p(m|\mathbf{t}_n; \boldsymbol{\pi}, \mathbf{P})(t_{nd} - p_{md}) \qquad m = 1, \dots, M \quad d = 1, \dots, D \tag{3.13}$$

where

$$p(m|\mathbf{t}_n; \boldsymbol{\pi}, \mathbf{P}) = \frac{p(\mathbf{t}_n|m; \boldsymbol{\pi}, \mathbf{P})p(m)}{\sum_{m'=1}^M p(\mathbf{t}_n|m'; \boldsymbol{\pi}, \mathbf{P})p(m')} = \frac{\pi_m \prod_{d=1}^D p_{md}^{t_{nd}}(1-p_{md})^{1-t_{nd}}}{\sum_{m'=1}^M \pi_{m'} \prod_{d=1}^D p_{m'd}^{t_{nd}}(1-p_{m'd})^{1-t_{nd}}} \tag{3.14}$$

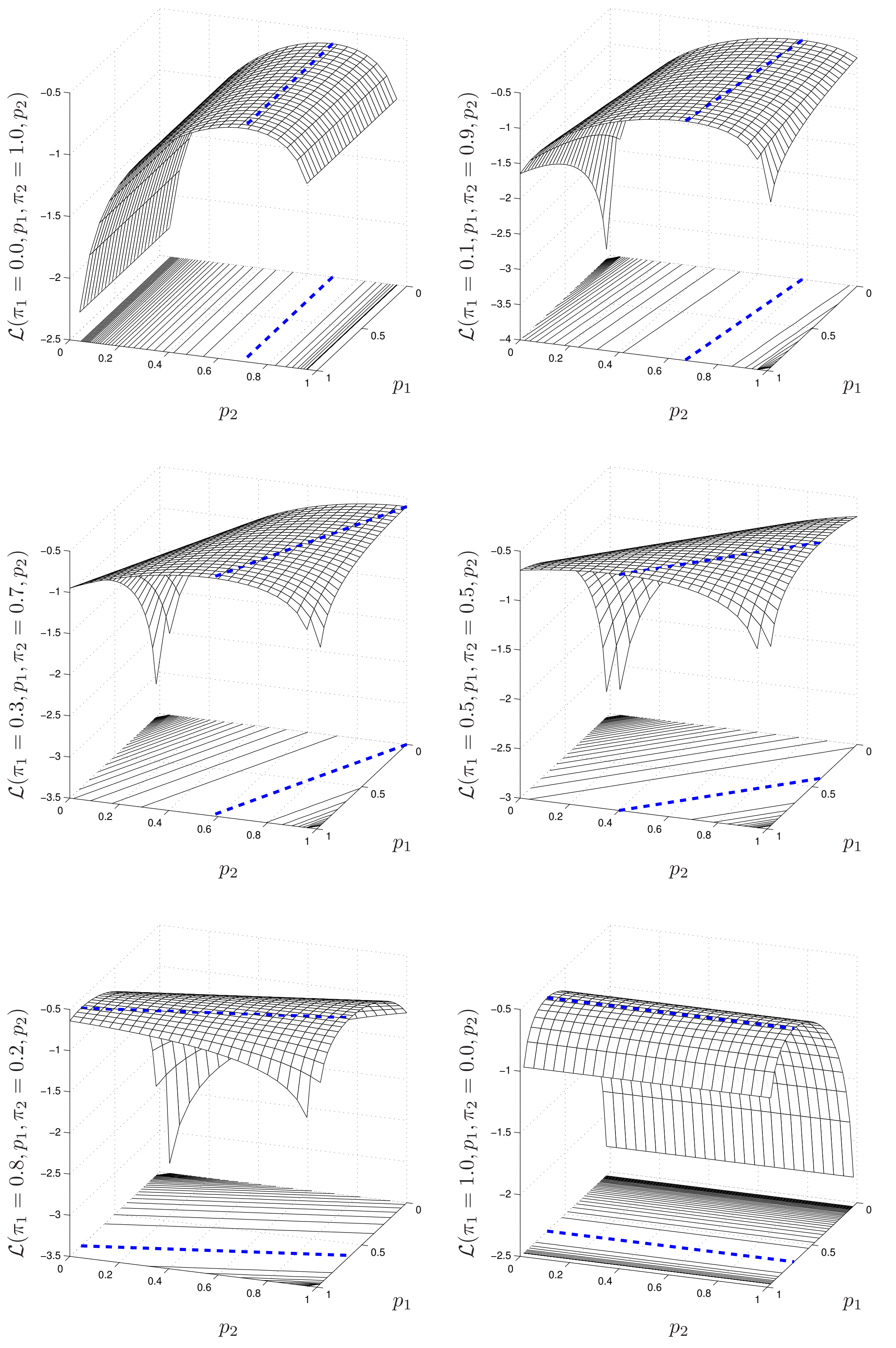


Figure 3.1: Sections of the log-likelihood hypersurface for constant $\pi_1 = 1 - \pi_2$ for a sample of mean 0.7 and a mixture of $M = 2$ univariate Bernoulli distributions. Note the contour lines in the (p_1, p_2) plane which are straight lines parallel to the line of maxima, $\pi_1 p_1 + \pi_2 p_2 =$ constant (thick dotted line). From left to right and top to bottom, the value of π_1 is 0, 0.1, 0.3, 0.5, 0.8 and 1.

are the posterior probabilities (or responsibilities) that component m generated data point $\mathbf{t}_n$. The term "$-N$" in eq. (3.12) results from the constraint $\sum_{m=1}^{M} \pi_m = 1$ introduced in the log-likelihood via a Lagrange multiplier.

Derivation of the EM algorithm[1] for finite mixtures of multivariate Bernoulli distributions is straightforward (Wolfe, 1970; Everitt and Hand, 1981). We give here the basic equations:

- E step: computation of the responsibilities using equation (3.14) from the current parameter estimates $\{\boldsymbol{\pi}^{(\tau)}, \mathbf{P}^{(\tau)}\}$ at iteration τ, $p(m|\mathbf{t}_n; \boldsymbol{\pi}^{(\tau)}, \mathbf{P}^{(\tau)})$.
- M step: reestimation of $\{\boldsymbol{\pi}^{(\tau+1)}, \mathbf{P}^{(\tau+1)}\}$:

$$\pi_m^{(\tau+1)} = \frac{1}{N} \sum_{n=1}^{N} p(m|\mathbf{t}_n; \boldsymbol{\pi}^{(\tau)}, \mathbf{P}^{(\tau)}) \qquad \mathbf{p}_m^{(\tau+1)} = \frac{1}{N\pi_m^{(\tau+1)}} \sum_{n=1}^{N} p(m|\mathbf{t}_n; \boldsymbol{\pi}^{(\tau)}, \mathbf{P}^{(\tau)})\mathbf{t}_n. \tag{3.15}$$

The sequence of parameters obtained for $\tau = 0, 1, 2, \dots$ by iterating between the E and M steps from any starting point $\{\boldsymbol{\pi}^{(0)}, \mathbf{P}^{(0)}\}$ produces a monotonically increasing sequence of values for the log-likelihood (Dempster et al., 1977).

A common problem of estimation in mixture distributions is that of singularities, that is, points in parameter space whose log-likelihood tends to positive infinity (e.g. a mixture of Gaussians in which one of the components is located on a data point and its variance tends to zero, thereby becoming a Dirac delta). Such singularities are undesirable because they give rise to degenerate distributions. Fortunately, the log-likelihood surface of a finite mixture of multivariate Bernoulli distributions has no singularities of value $+\infty$ (although it does have singularities of value $-\infty$). The reason is that both the log-likelihood (3.11) and its gradient (3.12)–(3.13) are bounded above in the whole parameter space, including its boundaries[2]. This means that estimation by the above EM algorithm from any nonpathological starting point, which is always possible by choosing p_{md} in $(0, 1)$, will always lead to a proper stationary point of the log-likelihood.

3.2.3 Stationary points of the log-likelihood

From proposition 3.2.1, no stationary point of the log-likelihood is a minimum. Besides, at any stationary point of the log-likelihood, equations (3.15) hold, so that we have

$$\mathrm{E}_{p(\mathbf{t})}\{\mathbf{t}\} = \sum_{m=1}^{M} \pi_m \, \mathrm{E}_{p(\mathbf{t}|m)}\{\mathbf{t}\} = \sum_{m=1}^{M} \pi_m \mathbf{p}_m = \sum_{m=1}^{M} \frac{1}{N} \sum_{n=1}^{N} p(m|\mathbf{t}_n)\mathbf{t}_n = \frac{1}{N} \sum_{n=1}^{N} \mathbf{t}_n = \overline{\mathbf{t}}$$

and so the mean of the mixture coincides with the sample mean at any stationary point. The converse does not hold generally.

The point in parameter space where $\mathbf{p}_m = \overline{\mathbf{t}}$ for all $m = 1, \dots, M$ and any distribution of the mixing proportions π_m is a stationary point of the log-likelihood, because $p(\mathbf{t}_n|m) = \prod_{d=1}^{D} \overline{t}_d^{\,t_{nd}} (1 - \overline{t}_d\,)^{(1-t_{nd})}$ is independent of m and therefore $p(m|\mathbf{t}_n) = \pi_m$ and the gradient is zero there. This point is equivalent to a single multivariate Bernoulli distribution, and should be avoided because experience shows that its log-likelihood,

$$\mathcal{L}(\{\pi_m, \mathbf{p}_m = \overline{\mathbf{t}}\}_{m=1}^{M}) = N \ln \prod_{d=1}^{D} \overline{t}_d^{\,\overline{t}_d} (1 - \overline{t}_d\,)^{1-\overline{t}_d}$$

is much smaller than that of other local maxima (an intuitive fact since the mixture is trivial). A starting point of the EM algorithm in which $\mathbf{p}_m$ is the same for all components (e.g. the apparently innocuous starting point $p_{md} = 1/2$ for all m and d) will lead to the mentioned trivial mixture after one EM iteration for any original distribution of the π_m. Our experiments showed that random starting points in $(0, 1)$ were less prone to leading to trivial mixtures.

---

[1]Our Matlab implementation of EM training of Bernoulli mixtures is freely available in the Internet (see appendix C).

[2]When $p_{md} \to 0$ for some m, d, the log-likelihood gradient in equation (3.13) remains bounded above, because for each n, either $t_{nd} - p_{md} \to k_1 p_{md}$ (if $t_{nd} = 0$) or $p(m|\mathbf{t}_n) \propto p(\mathbf{t}_n|m) \to k_2 p_{md}$ (if $t_{nd} = 1$), where k_1 and k_2 are constants. Similarly happens with the case $p_{md} \to 1$. Therefore the log-likelihood is differentiable for $p_{md} \in [0, 1]$, except in pathological situations where $|p_{md} - t_{nd}| = 1$ for all m and fixed n: in these cases $p(\mathbf{t}_n|m) = 0$ for all m and $\mathcal{L}(\{\pi_m, \mathbf{p}_m\}_{m=1}^{M}) \to -\infty$. Since the EM algorithm always climbs the log-likelihood surface, it will not be attracted by such singularities.

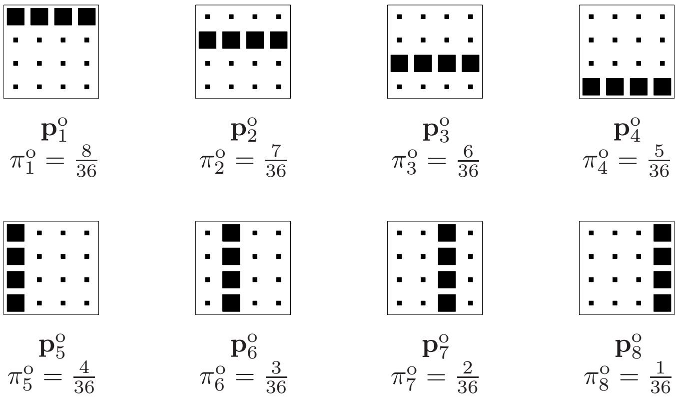


Figure 3.2: Parameters of the mixture of $M = 8$ 16-variate Bernoulli distributions used to generate the sample in section 3.3.2.1. For $m = 1, \dots, 8$, each 16-dimensional $\mathbf{p}_m$ vector is represented as a 4×4 image in which the area of each pixel is proportional to the value of its associated p_{md} parameter; for example, for the leftmost image, $\mathbf{p}_{1d}$ is 0.8 for $d \leq 4$ (row 1) and 0.2 for $d \geq 5$ (rows 2 to 4).

3.3 Theoretical and practical identifiability

3.3.1 Theoretical non-identifiability

The identifiability of a class of parametric models was defined in section 2.8 and refers to whether different values of the parameters can give rise to the same model, i.e., the same distribution—a fact that poses problems to the interpretability of the estimated model. Here we discuss the identifiability of the class of finite mixtures of multivariate Bernoulli distributions, which we use in chapter 5 to model the electropalatographic patterns, each of which is a binary vector of dimension 62. We will give empirical support to the fact that, in spite of the theoretical non-identifiability of this class of mixtures, estimation can still produce meaningful results in practice, thus lessening the importance of the identifiability problem.

Gyllenberg et al. (1994) prove that the class of finite mixtures of multivariate Bernoulli distributions is nontrivially non-identifiable[3] for all dimensions D. This means that there are many combinations of values of the parameter tuple $\boldsymbol{\Theta} = \{M,\ \{\pi_m, \mathbf{p}_m\}_{m=1}^M\}$ that produce an identical distribution $p(\mathbf{t})$: there exist at least two tuples $\boldsymbol{\Theta}$, $\boldsymbol{\Theta}'$ for which $p(\mathbf{t}|\boldsymbol{\Theta}) = p(\mathbf{t}|\boldsymbol{\Theta}')\ \forall \mathbf{t} \in \{0,1\}^D$. For example, it can be readily verified that the four mixtures given by the following parameter tuples represent the same distribution (here $D = 3$):

$$\begin{array}{ll}
\boldsymbol{\Theta}: & \big\{M = 1, \quad \big\{\pi_1 = 1,\ \mathbf{p}_1 = (\tfrac{1}{2}\ \tfrac{1}{2}\ \tfrac{1}{2})^T\big\}\big\} \\
\boldsymbol{\Theta}': & \big\{M = 2, \quad \big\{\pi_1 = \tfrac{1}{2},\ \mathbf{p}_1 = (\tfrac{1}{2}\ 0\ \tfrac{1}{2})^T\big\}, \quad \big\{\pi_2 = \tfrac{1}{2},\ \mathbf{p}_2 = (\tfrac{1}{2}\ 1\ \tfrac{1}{2})^T\big\}\big\} \\
\boldsymbol{\Theta}'': & \big\{M = 2, \quad \big\{\pi_1 = \tfrac{1}{4},\ \mathbf{p}_1 = (\tfrac{1}{2}\ 0\ \tfrac{1}{2})^T\big\}, \quad \big\{\pi_2 = \tfrac{3}{4},\ \mathbf{p}_2 = (\tfrac{1}{2}\ \tfrac{2}{3}\ \tfrac{1}{2})^T\big\}\big\} \\
\boldsymbol{\Theta}''': & \big\{M = 2, \quad \big\{\pi_1 = \tfrac{1}{4},\ \mathbf{p}_1 = (1\ \tfrac{1}{2}\ \tfrac{1}{2})^T\big\}, \quad \big\{\pi_2 = \tfrac{3}{4},\ \mathbf{p}_2 = (\tfrac{1}{3}\ \tfrac{1}{2}\ \tfrac{1}{2})^T\big\}\big\}\ .
\end{array}$$

However, it does not mean that for every parameter tuple $\boldsymbol{\Theta}$ there must exist at least one different $\boldsymbol{\Theta}'$ representing the same distribution. Identifiability is a property of the class of mixtures, rather than of a particular parameter tuple.

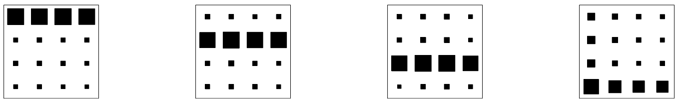
Hence, in principle there are many tuples in parameter space that are completely equivalent but that would give rise to different interpretations (recall that each prototype is supposed to represent a class of the population of binary vectors). This may also seem an insurmountable difficulty for parameter estimation. However, the fact that the EM algorithm of section 3.2.2 is guaranteed to converge to a proper maximum likelihood estimate and the practical studies shown in section 3.3.2 suggest that, given a sample from a mixture of multivariate Bernoulli distributions, EM maximum likelihood estimates of the parameters can be still interpretable.

3.3.2 Practical identifiability: experimental results

3.3.2.1 Synthetic data

We generated $N = 10\,000$ vectors in a binary space of $D = 16$ dimensions from a fixed mixture of $M = 8$ 16-variate Bernoulli distributions, whose parameters are shown in figure 3.2. We call these the *original* parameters

[3] As opposed to trivial non-identifiability, which is given by permutations of the mixture components or by coincident component distributions $p(\mathbf{t}|m)$ for several components.



$\mathbf{p}_1$ $\pi_1 = 0.2564$ | $\mathbf{p}_2$ $\pi_2 = 0.2401$ | $\mathbf{p}_3$ $\pi_3 = 0.2035$ | $\mathbf{p}_4$ $\pi_4 = 0.3001$

Figure 3.3: Parameters of a mixture of $M = 4$ 16-variate Bernoulli distributions estimated by maximum likelihood from the sample. $\mathbf{p}_1$, $\mathbf{p}_2$, $\mathbf{p}_3$ and $\mathbf{p}_4$ coincide very closely with $\mathbf{p}_1^o$, $\mathbf{p}_2^o$, $\mathbf{p}_3^o$ and $\frac{2}{3}\mathbf{p}_4^o + \frac{1}{3}\mathbf{p}_5^o$, respectively.

and denote them with a superscript "o", e.g. $\mathbf{p}_m^o$. From the sample alone, 10 maximum likelihood estimates were found for mixtures of $M = 4$, $M = 8$ and $M = 10$ components. We used the EM algorithm of section 3.2.2 with random starting values of the parameters p_{md} in the range $[\frac{1}{4}, \frac{3}{4}]$, stopping it when the relative change in log-likelihood was smaller than 10^{-6}. The starting values for the π_m parameters were fixed to $1/M$, thus giving each component the same weight at the beginning. The results were as follows[4]:

- Using the original number of components ($M = 8$), EM found the original parameters (both $\mathbf{p}_m$ and π_m) 9 out of 10 of the times; the normalised distance between the original and the estimated parameters was smaller than 0.0013 in those cases and the log-likelihood was $-94\,990$ (to 4 significant digits). However, the remaining estimate was a suboptimal maximum of the log-likelihood in which two of the components had the same $\mathbf{p}_m$ parameter ($\mathbf{p}_5 \approx \mathbf{p}_8 \approx \mathbf{p}_5^o$) and a log-likelihood of $-95\,201$. The difference between the log-likelihood values of the two estimates was of 0.2%.

- Using fewer components than necessary ($M = 4$) produced an estimate in which each prototype $\mathbf{p}_m$ was approximately either one of the original prototypes or a linear combination of several of the original prototypes (normalised distance smaller than 0.0023). Figure 3.3 shows the situation, for a particular estimate in which $\mathbf{p}_m \approx \mathbf{p}_m^o$ for $m = 1, 2, 3$ and $\mathbf{p}_4 \approx \frac{2}{3}\mathbf{p}_4^o + \frac{1}{3}\mathbf{p}_5^o$. As in the previous case, estimates having different prototypes also had very close log-likelihood values (differing in 0.5%).

- Using more components than necessary ($M = 10$) always produced the 8 original $\mathbf{p}_m^o$ vectors (normalised distance smaller than 0.0022) plus 2 extra ones, typically either repeated instances of some of the original ones or linear combinations of them. The log-likelihood value of each estimate did not differ from any of the others in more than 0.02%, indicating that once the 8 original prototypes are found, the remaining ones are largely irrelevant and reflect peculiarities of the sample used.

Experiments performed with other synthetic data sets produced similar results. We propose the following interpretation of the experimental facts. Given a large sample generated from a known mixture of multivariate Bernoulli distributions, let us construct another mixture in this way: first, pick up freely the number of components M; then, choose its $\mathbf{p}_1, \dots, \mathbf{p}_M$ prototypes either as some of the original ones or as linear combinations of them. Then, we claim that, for certain values of the mixing proportions, such a point in parameter space is very close to a maximum of the log-likelihood surface. However, we do not have theoretical support for this and we do not have a valid interpretation for the values of the mixing proportions.

The above results also suggest a procedure to follow when estimating an unknown mixture of multivariate Bernoulli distributions from a sample: choose freely a number of components M and, using EM from random starting points, find several (say 10) maximum likelihood estimates for it. Inspect the prototypes obtained. If they look the same for every estimate, then M is probably the right number of components and the estimate is very close to the true generating distribution. If a fixed group of prototypes appears in each estimate, and the rest of the prototypes are repetitions of those in the group, then M is probably too big; reduce it and start again. However, if there are different prototypes in different estimates, then M is probably too small; increase it and start again.

3.3.2.2 Electropalatographic (EPG) data

We used a subset[5] of electropalatography data from the EUR-ACCOR database containing 11 852 different 62-dimensional binary vectors (EPGs), obtained from 12 different utterances by a native English speaker. We

[4] We quantify the distance between two vectors $\mathbf{p}$, $\mathbf{q}$ in the D-dimensional rectangle $[0,1]^D$ with the normalised undirected distance $\frac{1}{D}\|\mathbf{p} - \mathbf{q}\|_2^2$, where $\|\cdot\|_2$ is the Euclidean norm. This distance is a real number in $[0,1]$ which averages to 1/6 for two uniformly random vectors. Observe that if $p_d = q_d + \epsilon$ for $d = 1, \dots, D$ then $\frac{1}{D}\|\mathbf{p} - \mathbf{q}\|_2^2 = |\epsilon|$ independently of D.

[5] This is the same kind of data as that used in chapter 5 and we refer the reader to it for a fuller description of the ACCOR data set and of electropalatography in general.

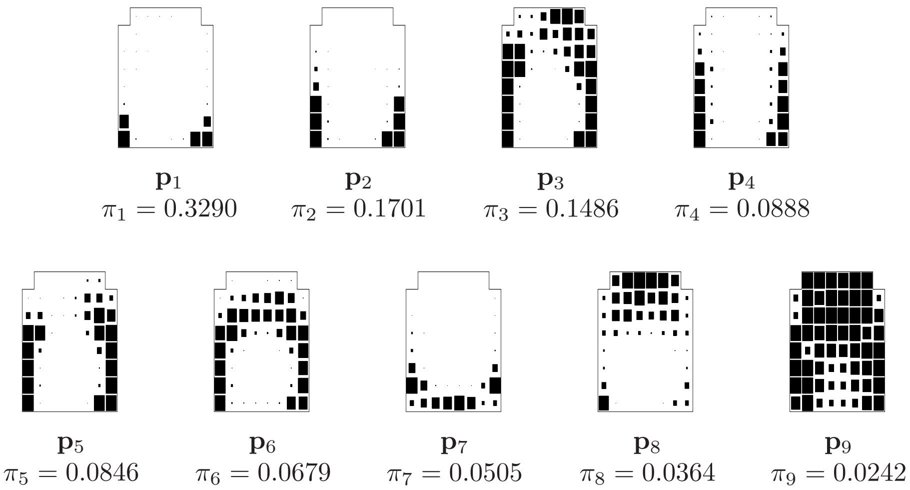


Figure 3.4: The 9 prototypes $\mathbf{p}_1, \dots, \mathbf{p}_9$ for a mixture of $M = 9$ multivariate Bernoulli distributions trained with the EPG data set. Each $\mathbf{p}_m$ vector, consisting of $D = 62$ values in the range $[0, 1]$, is customarily displayed as an image resembling the palate: the top row (alveolar part) contains parameters p_{m1} to p_{m6} from left to right, row 2 contains p_{m7} to p_{m14}, and so on till the bottom row (velar part), always from left to right. The area of each pixel is proportional to the value of its associated p_{md} parameter.

estimated the density of its distribution in 62-dimensional binary space using a finite mixture of multivariate Bernoulli distributions with $M = 6$ components. A number of estimates were found with the mentioned EM algorithm. As with the synthetic data set, the starting parameter values were $1/M$ for the π_m parameters and a random number in $\left[\frac{1}{4}, \frac{3}{4}\right]$ for the parameters p_{md}, and EM was stopped when the relative change in log-likelihood was smaller than 10^{-6}. We checked that, at that point, the norm of the gradient of the log-likelihood was 0 to acceptable precision and that the Hessian was negative definite, indicating that the point actually corresponded to a maximum. Examination of the parameter values at this point showed that:

- Several different kinds of estimates were found, each kind being characterised by a subset of the prototypes shown in fig. 3.4 and by specific values for the mixing proportions. Prototypes $\mathbf{p}_1$, $\mathbf{p}_2$, $\mathbf{p}_5$ and $\mathbf{p}_9$ appeared in almost every estimate (the normalised distance between corresponding prototypes did not exceed 0.02), while $\mathbf{p}_3$ and $\mathbf{p}_8$ were very common too.
- The log-likelihood of these mixtures was quite close, varying from $-138\,341$ for the combination $\{\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_5, \mathbf{p}_7, \mathbf{p}_8, \mathbf{p}_9\}$ to $-141\,697$ for the combination $\{\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3, \mathbf{p}_5, \mathbf{p}_6, \mathbf{p}_9\}$.
- Occasionally a prototype was found that could be expressed as a linear combination of some of the prototypes of fig. 3.4, e.g. $\frac{1}{2}\mathbf{p}_8 + \frac{1}{2}\mathbf{p}_9$.

We also found estimates for mixtures of M components, where M varied from 1 to 15. This made apparent the fact that for $M \gtrsim 9$, some prototypes appeared several times (with slight differences) in the same mixture, which therefore becomes trivial. This suggested selecting $M = 9$ as the optimum number of components for this data set[6], with typical values for the optimal parameters (both $\mathbf{p}_m$ and π_m) given in fig. 3.4. Considering a sequence of mixture estimates starting from $M = 1$ to $M = 15$, prototypes with high mixture proportions tended to appear early in the sequence, although at times somewhat distorted due to the interference with other prototypes. For example, prototype $\mathbf{p}_1$ in figure 3.4 was present in all mixtures, while $\mathbf{p}_6$ only starts to appear (very infrequently) for $M \geq 6$. However, prototype $\mathbf{p}_9$ appears very frequently for $M \geq 4$. Also, the log-likelihood for the data set considered increases with M and reaches a plateau for $M \approx 9$ (see fig. 3.5).

An interesting fact is that these prototypes are highly interpretable, corresponding to physically feasible EPGs and in fact assimilable to well-known quasi-static patterns in EPG studies such as those shown in fig. 5.3 (velar, alveolar, etc.), thus revealing important structure patterns in the data. These prototypes are similar to those produced by other methods, in particular latent variable models (chapter 5).

[6] An alternative way to select the critical M is to examine the log-likelihood curves for a validation set.

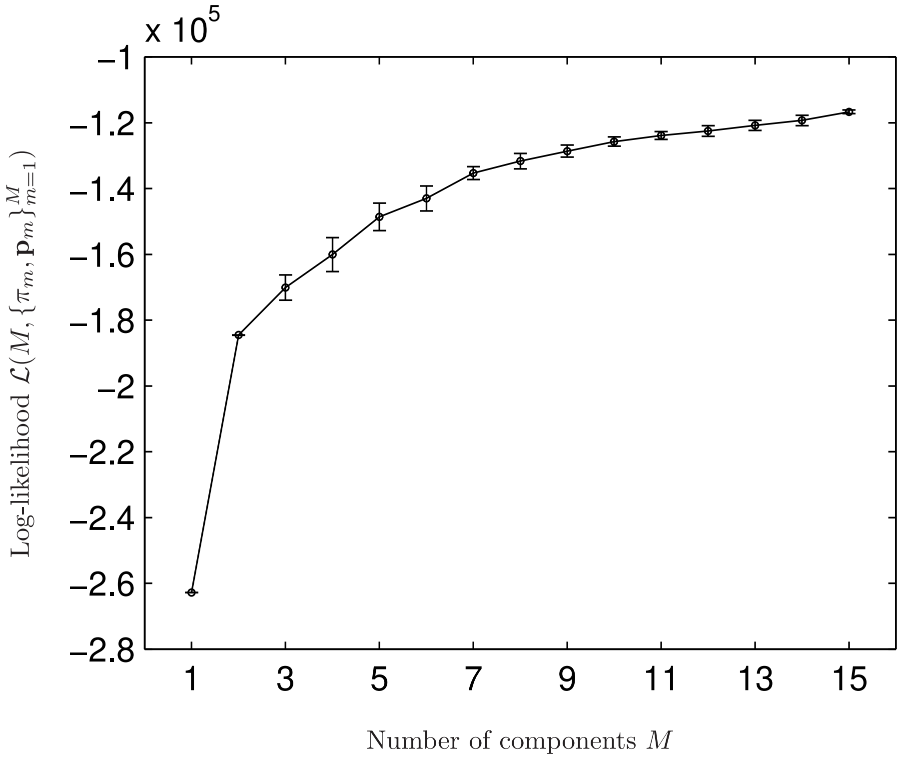


Figure 3.5: Log-likelihood of the mixture of multivariate Bernoulli distributions model for the EPG data as a function of the number of components M. Each value plotted corresponds to an average over 10 independent estimates and the error bars' length is equal to 2 standard deviations.

3.3.3 Conclusions

We have shown that, for the class of finite mixtures of multivariate Bernoulli distributions, the EM algorithm always converges to a proper stationary point of the log-likelihood, provided it is not started from a pathological point in parameter space (which is always possible). The reasons for this are the absence of singularities of value $+\infty$ in the log-likelihood surface and the fact that the EM algorithm always climbs that surface. Our Matlab implementation of the EM algorithm for finite mixtures of multivariate Bernoulli distributions is freely available in the Internet (see appendix C).

We have given empirical evidence that, for this particular class of finite mixture models, sensible and interpretable maximum likelihood estimates can be found even though that class of mixtures is not identifiable. This would suggest that the lack of identifiability might not be important from the practical point of view in some cases. Section 3.3.3.1 analyses the possible reasons for this with some detail, while section 3.3.3.2 shows that theoretical identifiability may not imply practical identifiability.

3.3.3.1 Abundance of equivalent parameter tuples

In the first place, the non-identifiability result is not surprising if one considers that a distribution defined over a discrete domain can be specified by a finite number of equations, one for each point of the domain. In our case, this means $2^D - 1$ equations (plus another one linearly dependent with them due to all the equations adding to one). Since the mixture has $MD + M - 1$ free parameters (one of the mixing proportions being linearly dependent on the others), making M larger than $\frac{2^D}{D+1}$ would yield an underdetermined system of equations with multiple solutions. However, even for small dimensions D this number is extremely large, and the number of components employed in practice will be much smaller. We restrict the rest of this discussion to a situation where the maximum number of components is much smaller than $\frac{2^D}{D+1}$, which yields an overdetermined system (where multiple solutions can still exist).

One reason for the apparent sparseness of the non-identifiability effect may be a low population of equivalent parameter tuples. Let us call *equivalent* to two different parameter tuples $\mathbf{\Theta}$, $\mathbf{\Theta}'$ which produce the same distribution. While we know that, for any dimension D, there exist equivalent parameter tuples, this does not mean that for each parameter tuple $\boldsymbol{\Theta}$ there will exist an equivalent, different one $\boldsymbol{\Theta}'$ (always for the case $M \ll \frac{2^D}{D+1}$). For example, it is easily seen that for the mixture in D dimensions with parameters given by:

$$\boldsymbol{\Theta}: \quad \left\{M = 2, \quad \left\{\pi_1 \in (0,1),\ \mathbf{p}_1 = (1\,1\ldots1)^T\right\}, \quad \left\{1-\pi_1,\ \mathbf{p}_2 = (0\,0\ldots0)^T\right\}\right\},$$

which produces a distribution

$$p(\mathbf{t}|\boldsymbol{\Theta}) = \begin{cases} \pi_1, & \mathbf{t} = (1\,1\ldots1) \\ 1-\pi_1, & \mathbf{t} = (0\,0\ldots0) \\ 0, & \text{otherwise} \end{cases},$$

there does not exist any equivalent parameter tuple $\boldsymbol{\Theta}'$ for $M < \frac{2^D}{D+1}$ (disregarding, as usual, permutations of mixture components and coincident component distributions). Thus, the actual practical problem of identifiability is: which parameter tuples $\boldsymbol{\Theta}$ have equivalent parameter tuples? Or, considering the partition of the space of parameter tuples into classes of equivalence (each class of equivalence consisting of all equivalent parameter tuples): what is the cardinality of each class? This is a difficult question to answer analytically with generality, but the experimental results suggest that nontrivial equivalence classes (consisting of more than one element) may be rare, perhaps pathological.

Another reason for the possibility of estimating a sensible parameter tuple seems to be that every estimate contains some of the original prototypes, and that the original number of components may be selected by inspection of a collection of maximum likelihood estimates obtained with the EM algorithm (as we did in section 3.3.2.1). Note that the likelihood function associated with a small sample (compared to the total number of possible different vectors, usually 2^D) need not be maximised by the original mixture.

3.3.3.2 Theoretical identifiability does not guarantee practical identifiability

Finally, let us remark that a reciprocal situation to the one described in this section may also be possible, as the following example in $D = 1$ dimension shows. Consider the class of mixtures of normal distributions, which is known to be identifiable for all dimensions (Everitt and Hand, 1981). That is, no two different mixtures of that class represent the same distribution: $p(\mathbf{x}; \{\pi_m, \boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m\}_{m=1}^M) = p(\mathbf{x}; \{\pi'_m, \boldsymbol{\mu}'_m, \boldsymbol{\Sigma}'_m\}_{m=1}^{M'})\ \forall \mathbf{x} \in \mathbb{R}^D$ implies $M = M'$ and $\pi_m = \pi'_m$, $\boldsymbol{\mu}_m = \boldsymbol{\mu}'_m$, $\boldsymbol{\Sigma}_m = \boldsymbol{\Sigma}'_m$ for all $m = 1, \ldots, M$, perhaps with a reordering of the indices. Now, let us consider the following two specific mixtures of univariate normal distributions ($D = 1$):

- The first mixture, f, has two equiprobable components ($M = 2$), with parameters:

  $$\left\{M = 2, \quad \left\{\pi_1 = \tfrac{1}{2},\ \mu_1 = -1,\ \sigma_1 = \zeta\right\}, \quad \left\{\pi_2 = \tfrac{1}{2},\ \mu_2 = 1,\ \sigma_2 = \zeta\right\}\right\}.$$

  That is, $f = \pi_1 f_1 + \pi_2 f_2$ where $f_1 \sim \mathcal{N}(\mu_1, \zeta)$ and $f_2 \sim \mathcal{N}(\mu_2, \zeta)$.

- The second mixture, g, is a single normal distribution ($M = 1$), with parameters:

  $$\{M = 1, \quad \{\pi = 1,\ \mu = 0,\ \sigma = \xi\}\}.$$

ζ and ξ are two positive numbers that control the dispersion of both mixtures. Due to the symmetry of the parameters, both mixture distributions are even functions of x.

The L_2-norm of a real function $f(x)$ is defined as $\|f\|_2 = \left(\int f^2(x)\,dx\right)^{\frac{1}{2}}$, where the integral is extended to the domain of f, $(-\infty, \infty)$ for the case of a univariate normal. The distance between two functions f and g can be defined in terms of the L_2-norm as $\|f - g\|_2$. Taking as f and g our two mixtures, respectively, let us compute the squared distance between them in the L_2-norm sense:

$$\begin{aligned}
\|f - g\|_2^2 &= \int_{-\infty}^{\infty} (f(x) - g(x))^2\,dx \\
&= \int_{-\infty}^{\infty} \left(\frac{1}{4}f_1^2(x) + \frac{1}{4}f_2^2(x) + \frac{1}{2}f_1(x)f_2(x) + g^2(x) - f_1(x)g(x) - f_2(x)g(x)\right) dx \\
&= \frac{1}{4}I(-1,\zeta) + \frac{1}{4}I(1,\zeta) + \frac{1}{2}I(-1,\zeta;1,\zeta) + I(0,\xi) - I(-1,\zeta;0,\xi) - I(1,\zeta;0,\xi) \\
&= \frac{1}{2\sqrt{\pi}}\left(\frac{1}{2\zeta}\left(1 + e^{-\frac{1}{\zeta^2}}\right) + \frac{1}{\xi} - \frac{2\sqrt{2}}{\sqrt{\zeta^2+\xi^2}}e^{-\frac{1}{2(\zeta^2+\xi^2)}}\right) = \mathcal{D}(\xi;\zeta)
\end{aligned}$$

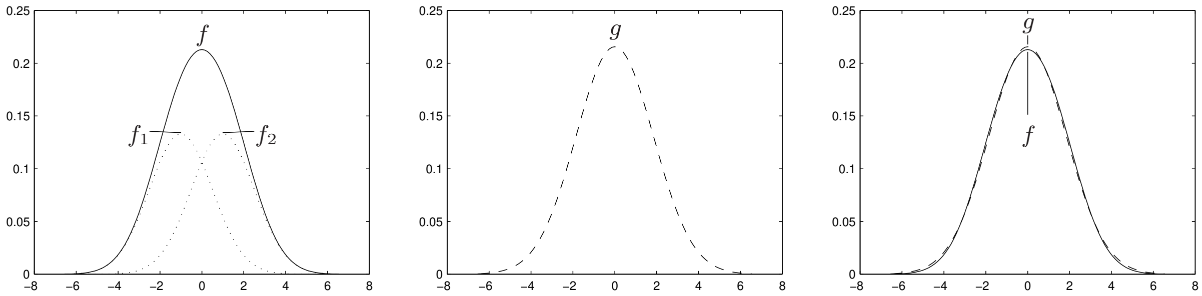


Figure 3.6: Density plots of two mixtures of univariate normal distributions. *Left*: two-component mixture $f(x) = \pi_1 f_1(x) + \pi_2 f_2(x)$, where $\pi_1 = \pi_2 = \frac{1}{2}$ and $f_1 \sim \mathcal{N}(\mu = -1, \sigma = 1.5)$, $f_2 \sim \mathcal{N}(\mu = 1, \sigma = 1.5)$ (component distributions f_1 and f_2: dotted line; mixture distribution f: solid line). *Centre*: single-component mixture $g(x) \sim \mathcal{N}(\mu = 0, \sigma = 1.85)$ (dashed line). *Right*: both mixture distributions f and g. $\|f\|_2 = 0.3928$, $\|g\|_2 = 0.3905$, $\|f - g\|_2 = 0.0081$.

where we have used the following formulae for the convolution of Gaussians:

$$I(\mu_1, \sigma_1; \mu_2, \sigma_2) = \int_{-\infty}^{\infty} \mathcal{N}(x; \mu_1, \sigma_1)\mathcal{N}(x; \mu_2, \sigma_2)\, dx = \frac{1}{\sqrt{2\pi(\sigma_1^2 + \sigma_2^2)}} e^{-\frac{1}{2}\left(\frac{\mu_1 - \mu_2}{\sqrt{\sigma_1^2 + \sigma_2^2}}\right)^2}$$

$$I(\mu, \sigma) = I(\mu, \sigma; \mu, \sigma) = \frac{1}{2\sqrt{\pi}\sigma}$$

where $\mathcal{N}(x; \mu, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$. Likewise, the L_2-norms of each mixture are:

$$\|f\|_2 = \sqrt{\frac{\left(1 + e^{-\frac{1}{\zeta^2}}\right)}{4\sqrt{\pi}\zeta}} \qquad \|g\|_2 = \frac{1}{\sqrt{2\sqrt{\pi}\xi}}.$$

Note that

$$\lim_{\zeta,\xi \to \infty} \mathcal{D}(\xi; \zeta) = 0 \qquad \lim_{\substack{\xi \to 0^+ \\ \zeta > 0}} \mathcal{D}(\xi; \zeta) = \infty \qquad \lim_{\substack{\xi \to \infty \\ \zeta > 0}} \mathcal{D}(\xi; \zeta) = \|f\|_2^2 > 0.$$

Thus, that distance can be made as small as desired by making both ζ and ξ large enough. But we do not need to go to pathological situations. It is easy to see that, given ζ, $\mathcal{D}$ has a minimum in $(0, \infty)$ as a function of ξ (i.e., there is a best L_2-approximation from g to f for every ζ). The case $\zeta = 1.5$ and $\xi = 1.85$ corresponds to one of these minima. It is depicted in fig. 3.6 and shows how close, in the L_2-norm sense, the two mixtures can get. In this case, the L_2-distance is just 0.0081, which is about 50 times smaller than the L_2-norm of any of the mixtures.

This implies that a sample of enormous size will be necessary to tell f from g in terms of likelihood of the parameters, which means that in practice they may become indistinguishable. That is, it would be difficult on the grounds of a given sample to determine whether the sample was generated by a single normal process g or by a combination of two different normal processes f_1 and f_2.

Thus, theoretical identifiability[7] does not guarantee practical identifiability and interpretation problems may still arise with samples of limited size.

[7] Not just of the class of normal distributions, but probably of other classes of mixtures which are identifiable in theory, specially if they have the property of universal approximation (Titterington et al., 1985; Scott, 1992).

Part II

Dimensionality reduction

Chapter 4

Dimensionality reduction

This chapter introduces and defines the problem of dimensionality reduction, discusses the topics of the curse of the dimensionality and the intrinsic dimensionality and then surveys non-probabilistic methods for dimensionality reduction, that is, methods that do not define a probabilistic model for the data. These include linear methods (PCA, projection pursuit), nonlinear autoassociators, kernel methods, local dimensionality reduction, principal curves, vector quantisation methods (elastic net, self-organising map) and multidimensional scaling methods. One of these methods (the elastic net) does define a probabilistic model but not a continuous dimensionality reduction mapping. If one is interested in stochastically modelling the dimensionality reduction mapping then the natural choice are latent variable models, discussed in chapter 2. We close the chapter with a summary and with some thoughts on dimensionality reduction with discrete variables.

4.1 Introduction

Consider an application in which a system processes data in the form of a collection of real-valued vectors: speech signals, images, etc. Suppose that the system is only effective if the dimension of each individual vector—the number of components of the vector—is not too high, where *high* depends on the particular application. The problem of dimensionality reduction appears when the data are in fact of a higher dimension than tolerated. For example, take the following typical cases:

- A face recognition/classification system based on $m \times n$ greyscale images which, by row concatenation, can be transformed into mn-dimensional real vectors. In practice, one could have images of $m = n = 256$, or 65536-dimensional vectors; if, say, a multilayer perceptron was to be used as the classification system, the number of weights would be exceedingly large and would require an enormous training set to avoid overfitting. Therefore we need to reduce the dimension. While a crude solution in this case would be to simply scale down the images to a manageable size, more elaborate approaches exist.
- A statistical analysis of a multivariate population. Typically there will be a few variables and the analyst is interested in finding clusters or other structure of the population and/or interpreting the variables. To that aim, it is quite convenient to visualise the data, which requires reducing its dimensionality to 2 or 3.

Therefore, in a number of occasions it can be useful or even necessary to first reduce the dimensionality of the data to a manageable size, keeping as much of the original information as possible, and then feed the reduced-dimension data into the system. Figure 4.1 summarises this situation, showing the dimensionality reduction as a preprocessing stage in the whole system.

More generally, whenever the intrinsic dimensionality of a data set is smaller than the actual one, dimensionality reduction can bring an improved understanding of the data apart from a computational advantage. Dimensionality reduction can also be seen as a feature extraction or coding procedure, or in general as a representation in a different coordinate system. This is the basis for the definition given in section 4.2.

4.1.1 Classes of dimensionality reduction problems

We attempt here a rough classification of the dimensionality reduction problems:

This chapter is an extended version of reference Carreira-Perpiñán (1996).

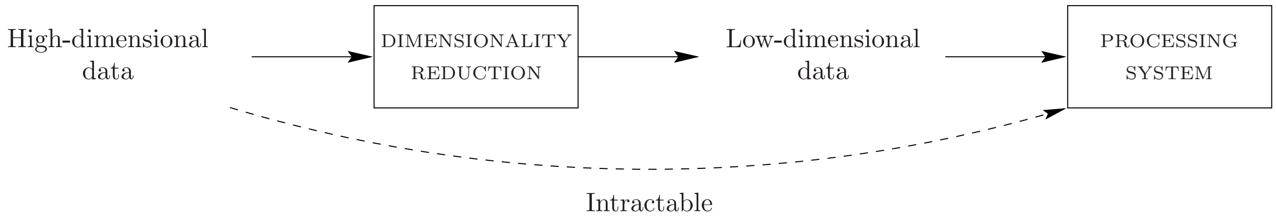


Figure 4.1: The dimensionality reduction problem. A given processing system is only effective with vector data of not more than a certain dimension, so data of higher dimension must be reduced before being fed into the system.

- *Hard* dimensionality reduction problems, in which the data have dimensionality ranging from hundreds to perhaps hundreds of thousands of components, and usually a drastic reduction (possibly of orders of magnitude) is sought. The components are often repeated measures of a certain magnitude in different points of space or in different instants of time. In this class we would find pattern recognition and classification problems involving images (e.g. face recognition, character recognition, etc.) or speech (e.g. auditory models).
- *Soft* dimensionality reduction problems, in which the data is not too high-dimensional (less than a few tens of components), and the reduction not very drastic. Typically, the components are observed or measured values of different variables, which have a straightforward interpretation. Most statistical analyses in fields like social sciences, psychology, etc. fall in this class.
- *Visualisation* problems, in which the data does not normally have a very high dimension in absolute terms, but we need to reduce it to 2 or 3 in order to plot it. Several representation techniques (reviewed by Scott, 1992) exist that allow to visualise up to about 5-dimensional data sets, using colours, rotation, stereography, glyphs or other devices, but they lack the appeal of a simple plot; a well-known one is the grand tour (Asimov, 1985). Chernoff faces (Chernoff, 1973) allow even a few more dimensions, but are difficult to interpret and do not produce a spatial view of the data.

If we allow the time variable in, we find two further categories: *static dimensionality reduction* and *time-dependent dimensionality reduction*. The latter could possibly be useful for vector time series, such as video sequences or continuous speech. We deal here only with static dimensionality reduction.

4.2 Definition of the problem of dimensionality reduction

Suppose we have a sample $\{\mathbf{t}_n\}_{n=1}^N$ of D-dimensional vectors lying in a data space $\mathcal{T}$ (usually $\mathbb{R}^D$ or a subset of it[1]). The fundamental assumption that justifies the dimensionality reduction is that the sample actually lies, at least approximately, on a manifold[2] (nonlinear in general) of smaller dimension than the data space. The goal of dimensionality reduction is to find a representation of that manifold (a coordinate system) that will allow to project the data vectors on it and obtain a low-dimensional, compact representation of the data.

Formally, dimensionality reduction consists of the following problem: given a sample $\{\mathbf{t}_n\}_{n=1}^N \subset \mathcal{T}$, find:

- A space $\mathcal{X}$ of dimension L (typically $\mathbb{R}^L$ or a subset of it).
- A **dimensionality reduction mapping F**:
$$\begin{array}{rccc} \mathbf{F}: & \mathcal{T} & \longrightarrow & \mathcal{X} \\ & \mathbf{t} & \mapsto & \mathbf{x} = \mathbf{F}(\mathbf{t}). \end{array}$$
  As in section 2.9.1, we call $\mathbf{x}$ the *reduced-dimension representative* of $\mathbf{t}$.
- A smooth, nonsingular[3] **reconstruction mapping f**:
$$\begin{array}{rccc} \mathbf{f}: & \mathcal{X} & \longrightarrow & \mathcal{M} \subset \mathcal{T} \\ & \mathbf{x} & \mapsto & \mathbf{t} = \mathbf{F}(\mathbf{x}). \end{array}$$

[1]As in chapter 2, we restrict ourselves to continuous variables.
[2]Section A.7 gives a formal definition of L-manifolds and coordinate systems.
[3]The reasons for requiring that the reconstruction mapping be smooth and nonsingular are the same as in chapter 2: to ensure that the domain and range have the same dimension (nonsingular) and to preserve the topographic structure of the domain (continuity). Piecewise smooth functions would be acceptable too.

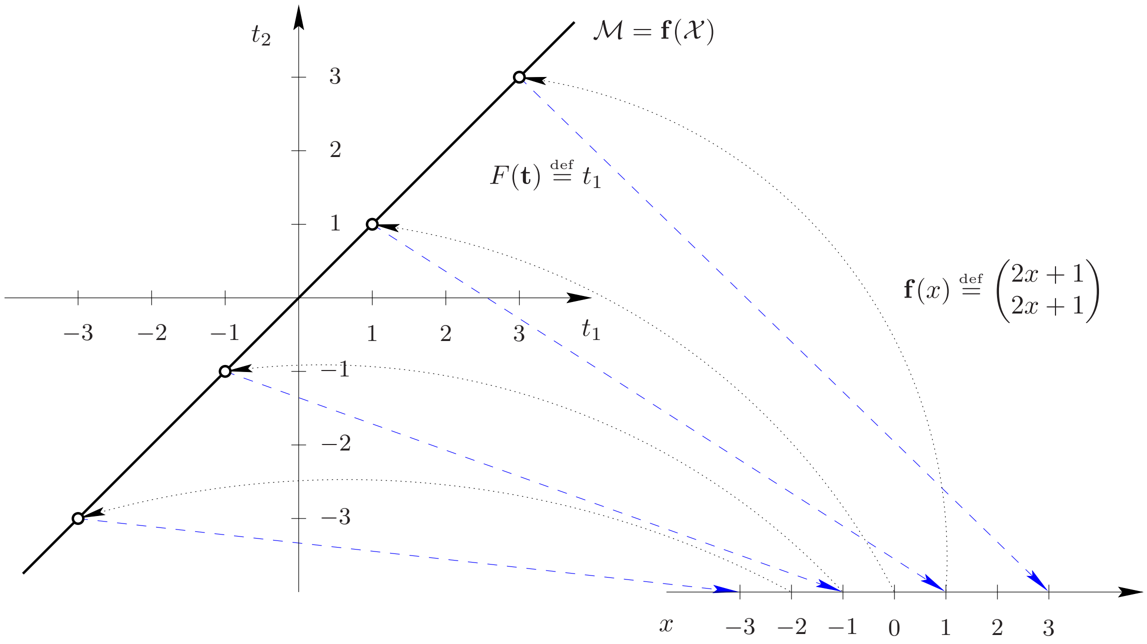


Figure 4.2: The dimensionality reduction mapping $\mathbf{F}$ (dashed lines) and the reconstruction mapping $\mathbf{f}$ (dotted lines) need not verify $\mathbf{F} \circ \mathbf{f} =$ identity.

Such that:

- $L < D$ is as small as possible.
- The following condition is satisfied:

$$\textit{The manifold } \mathcal{M} \stackrel{\text{def}}{=} \mathbf{f}(\mathcal{X}) \textit{ approximately contains all the sample points: } \{\mathbf{t}_n\}_{n=1}^N \underset{\sim}{\subsetneq} \mathcal{M}. \tag{DR}$$

  This condition can be restated in a different way:

$$\textit{The reconstruction error of the sample is small.} \tag{DR$'$}$$

  The reconstruction error of the sample is defined as $E_d(\{\mathbf{t}_n\}_{n=1}^N) \stackrel{\text{def}}{=} \sum_{n=1}^N d(\mathbf{t}_n, \mathbf{t}_n^*)$ where $\mathbf{t}_n^* \stackrel{\text{def}}{=} \mathbf{f}(\mathbf{F}(\mathbf{t}_n))$ is the reconstructed vector for point $\mathbf{t}_n$ and d is a suitable distance in the space $\mathcal{T}$ (e.g. the Euclidean distance in $\mathbb{R}^D$).

Conditions (DR) and (DR$'$) are not equivalent: (DR$'$) implies (DR) but not vice versa. This is because, for a given manifold $\mathcal{M} = \mathbf{f}(\mathcal{X}) \subset \mathcal{T}$, $\mathbf{F} \circ \mathbf{f}$ need not be the identity mapping, as fig. 4.2 shows. Thus, it is possible that $\{\mathbf{t}_n\}_{n=1}^N \subset \mathcal{M}$, i.e., for each $\mathbf{t}_n$ there exists a point $\mathbf{x}_n \in \mathcal{X}$ with $\mathbf{f}(\mathbf{x}_n) = \mathbf{t}_n$, but $\mathbf{F}(\mathbf{t}_n) \neq \mathbf{x}_n$. Therefore $\mathbf{t}_n^* \stackrel{\text{def}}{=} \mathbf{f}(\mathbf{F}(\mathbf{t}_n)) \neq \mathbf{t}_n$ and $E_d(\{\mathbf{t}_n\}_{n=1}^N)$ may be large. This is not an arbitrary argument: the dimensionality reduction mappings derived from latent variable models are of this kind, due to the existence of a probability distribution on the space $\mathcal{X}$ and of a noise model, as described in section 2.9.

Due to the dimensionality mismatch between $\mathcal{X}$ and $\mathcal{T}$, there will be a whole submanifold of dimension $D - L$ in $\mathcal{T}$ that will be mapped onto the same point in $\mathcal{X}$, i.e., $\mathbf{F}^{-1}(\mathbf{x}) \stackrel{\text{def}}{=} \{\mathbf{t} \in \mathcal{T} : \mathbf{F}(\mathbf{t}) = \mathbf{x}\}$ will be a submanifold of dimension $D - L$, as discussed in section 2.9.1. For the example of fig. 4.2, $\mathbf{F}^{-1}(\mathbf{x})$ is the vertical line passing through the point $\mathbf{t} = (x\ 0)^T$.

The election of the coordinate system for a given manifold is not unique. For example, in figure 4.3 a one-dimensional nonlinear manifold in $\mathbb{R}^3$ (a curve), namely a spiral of radius R and step s, is parameterised in terms of the dimensionless parameter x: $\mathcal{M} \stackrel{\text{def}}{=} \{\mathbf{t} \in \mathbb{R}^3 : \mathbf{t} = \mathbf{f}(x), x \in [x_A, x_B]\}$ for $\mathbf{f}(x) \stackrel{\text{def}}{=} (R \sin 2\pi x, R \cos 2\pi x, sx)^T$. We could reparameterise the manifold in terms of the arc length λ between points $\mathbf{t}_A \stackrel{\text{def}}{=} \mathbf{f}(x_A)$ and $\mathbf{t}_B \stackrel{\text{def}}{=} \mathbf{f}(x_B)$:

$$\lambda \stackrel{\text{def}}{=} \int_{\mathbf{t}_A}^{\mathbf{t}_B} \sqrt{\left(\frac{dt_1}{dx}\right)^2 + \left(\frac{dt_2}{dx}\right)^2 + \left(\frac{dt_3}{dx}\right)^2}\, dx = \sqrt{(2\pi R)^2 + s^2}(x_B - x_A).$$

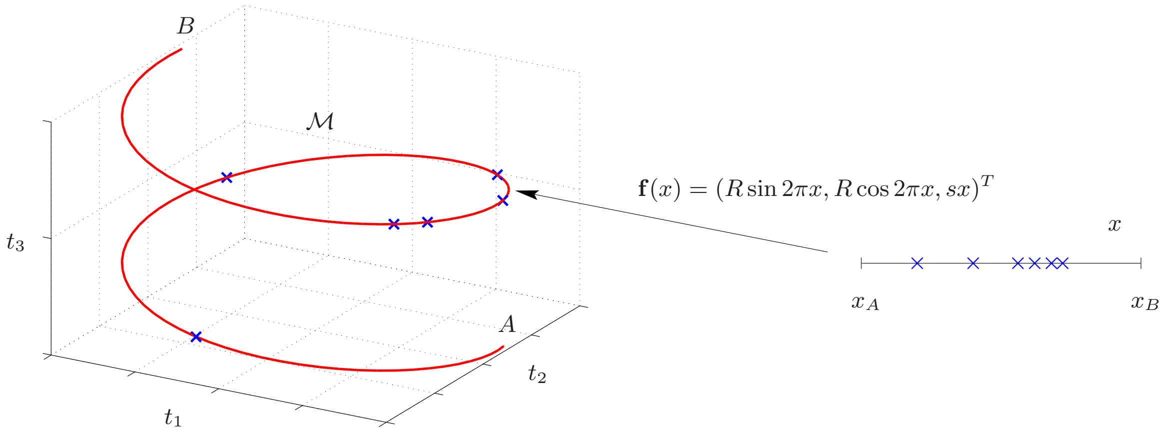


Figure 4.3: An example of coordinate representation of a one-dimensional manifold $\mathcal{M}$ in $\mathbb{R}^3$ (a curve): the segment of spiral $\mathcal{M} = \{\mathbf{t} \in \mathbb{R}^3 : \mathbf{t} = \mathbf{f}(x), x \in [x_A, x_B]\}$ for $\mathbf{f}(x) = (R\sin 2\pi x, R\cos 2\pi x, sx)^T$.

Yet another parameterisation would be in terms of the angle $\theta = 2\pi x$, etc. Another example are the Cartesian, spherical and cylindrical systems in $\mathbb{R}^3$. Any coordinate system is in principle acceptable, although some may be more appropriate than others for certain problems. Also, constraints on the problem may make the choice of coordinates unique. For example, in PCA the basis vectors are constrained to be orthonormal and ordered according to variance, which makes the choice of basis vectors unique (except when the data covariance matrix has multiple eigenvalues).

4.3 The curse of the dimensionality

The term *curse of the dimensionality*[4], coined by Bellman (1961), refers to the fact that, in the absence of simplifying assumptions, the sample size needed to estimate a function of several variables to a given degree of accuracy (i.e., to get a reasonably low-variance estimate) grows exponentially with the number of variables.

A related fact, responsible for the curse of the dimensionality, is the *empty space phenomenon* (Scott and Thompson, 1983): high-dimensional spaces are inherently sparse. For example, the probability that a point distributed uniformly in the unit 10-dimensional sphere falls at a distance of 0.9 or less from the centre is only 0.35. This is a difficult problem in multivariate density estimation, as regions of relatively very low density can contain a considerable part of the distribution, whereas regions of apparently high density can be completely devoid of observations in a sample of moderate size (Silverman, 1986). For example, for a one-dimensional standard normal $\mathcal{N}(0,1)$, 70% of the mass is at points contained in a sphere of radius one standard deviation (i.e., the $[-1,1]$ interval); for a 10-dimensional $\mathcal{N}(\mathbf{0},\mathbf{I})$, that same (hyper)sphere contains only 0.02% of the mass and one has to take a radius of more than 3 standard deviations to contain 70%. Therefore, and contrarily to our intuition, in high-dimensional distributions the tails are much more important than in one-dimensional ones.

The curse of the dimensionality has the following consequence for density estimation: since most density estimation methods are based on some local average of the neighbouring observations (Silverman, 1986), in order to find enough neighbours in high-dimensional spaces, the neighbourhood has to reach out farther and the locality is lost. Another problem caused by the curse of the dimensionality is that, if there are linear correlations in the data (a very likely situation in high dimensions), the optimal mean integrated squared error when estimating the data density will be very large even if the sample size is arbitrarily large (Scott, 1992).

[4]As a philosophical aside, it is interesting that the curse of the dimensionality appears implicitly in Francis Bacon's inductive method. Bacon introduced the idea of induction in 1620 in his *Novum Organum* as a response to deduction, which Aristotle had formalised in his *Organum*. In Bacon's inductive method, the task of the scientist would be to derive laws of nature by carefully constructing tables containing the experimental data. Each line in a table would correspond to a combination of values of the variables having an influence on the phenomenon observed and the presence or absence of that phenomenon (a kind of binary classification problem in our days!), or the degree to which the phenomenon manifested itself (multivariate regression!). Given the combinatorial explosion of the table size, acknowledged by Bacon, it is not surprising that the method was never implemented—although its merit resides in the introduction of the concept of induction rather than in the method.

However, it is important to remark that the curse of the dimensionality is governed not by the dimensionality D of the data space $\mathcal{T}$, but by the intrinsic dimensionality L of the data. Or, more precisely, by the dimension L of the space $\mathcal{X}$ which the dimensionality reduction method is imposing. This is because the sample size required really depends on the volume of hyperspace occupied by the manifold being modelled, of dimension L, not on the higher-dimension space where it is embedded: thus the sample size grows as $\mathcal{O}(e^L)$. An example of this are latent variable models based on Monte Carlo sampling of the latent space (section 2.4), such as GTM.

Due to the fundamental character of the curse of the dimensionality, all dimensionality reduction methods (particularly the more general ones) are affected by it to some extent through the number of parameters that need to be estimated.

4.3.1 The geometry of high-dimensional spaces

The geometry of high-dimensional spaces provides a few surprises related to the curse of the dimensionality. Although, in fact, one should say that the surprises are in the usual, intuitive low-dimensional cases of 1 to 3 dimensions, when compared to the general (asymptotic) case of higher dimensions. We illustrate now some of these intriguing results for the case of the Euclidean space $\mathbb{R}^D$. First note that:

- The volume of the D-hypersphere of radius R is $V(\mathbb{S}_R^D) = V(\mathbb{S}_1^D)R^D$ with dimension-dependent constant

$$V(\mathbb{S}_1^D) = \frac{\pi^{D/2}}{\Gamma(\frac{D}{2}+1)}$$

  where $\Gamma(x)$ is the gamma function.

- The volume of the D-hypercube of side $2R$ is $V(\mathbb{C}_R^D) = V(\mathbb{C}_1^D)R^D$ with dimension-dependent constant $V(\mathbb{C}_1^D) = 2^D$.

Both volumes depend exponentially on the linear size of the object, but the constants are very different. This has as an interesting consequence a distortion of the space. Consider the following situations in the limit of high dimensions:

Sphere inscribed in a hypercube (Scott, 1992): the ratio of the volume of the hypersphere to the volume of the hypercube is

$$\frac{V(\mathbb{S}_1^D)}{V(\mathbb{C}_1^D)} = \frac{\pi^{D/2}}{2^D\Gamma(\frac{D}{2}+1)} \xrightarrow[D\to\infty]{} 0.$$

That is, with increasing dimension the volume of the hypercube concentrates on its corners and the centre becomes less important. Table 4.1 and figure 4.4 show the volumes $V(\mathbb{S}_1^D)$, $V(\mathbb{C}_1^D)$ and the ratio between them for several dimensions.

Hypervolume of a thin shell (Wegman, 1990): consider the volume between two concentric spheric shells of respective radii R and $R(1-\epsilon)$, with ϵ small. Then the ratio

$$\frac{V(\mathbb{S}_R^D) - V(\mathbb{S}_{R(1-\epsilon)}^D)}{V(\mathbb{S}_R^D)} = 1 - (1-\epsilon)^D \xrightarrow[D\to\infty]{} 1.$$

Hence, virtually all the content of a hypersphere is concentrated close to its surface, which is only a $(D-1)$-dimensional manifold (see section A.7). Thus, for data distributed uniformly over both the hypersphere and the hypercube, most of the data fall near the boundary and edges of the volume. This example illustrates one important aspect of the curse of the dimensionality mentioned earlier. Figure 4.4 illustrates this point for $\epsilon = 0.1$.

Tail probability of the multivariate normal (Scott, 1992): the preceding examples make it clear that most (spherical) neighbourhoods of data distributed uniformly over a hypercube in high dimensions will be empty. In the case of the standard D-dimensional normal distribution, the equiprobable contours are hyperspheres. The probability that a point is within a contour of density ϵ times the value at the mode, or, equivalently, inside a hypersphere of radius $\sqrt{-2\ln\epsilon}$, is:

$$\Pr\left[\|\mathbf{x}\|^2 \le -2\ln\epsilon\right] = \Pr\left[\chi_D^2 \le -2\ln\epsilon\right] \tag{4.1}$$

D	1	2	3	4	$\cdots$	10
$V(\mathbb{S}_1^D)$	2	π	$\frac{4}{3}\pi$	$\frac{\pi^2}{2}$	$\cdots$	$\frac{\pi^5}{120} \approx 2.55$
$V(\mathbb{C}_1^D)$	2	4	8	16	$\cdots$	1024
$\frac{V(\mathbb{S}_1^D)}{V(\mathbb{C}_1^D)}$	1	$\frac{\pi}{4}$	$\frac{\pi}{6}$	$\frac{\pi^2}{32}$	$\cdots$	$\frac{\pi^5}{122880} \approx 0.0025$

Table 4.1: Volumes of unit D-hypersphere and D-hypercube.

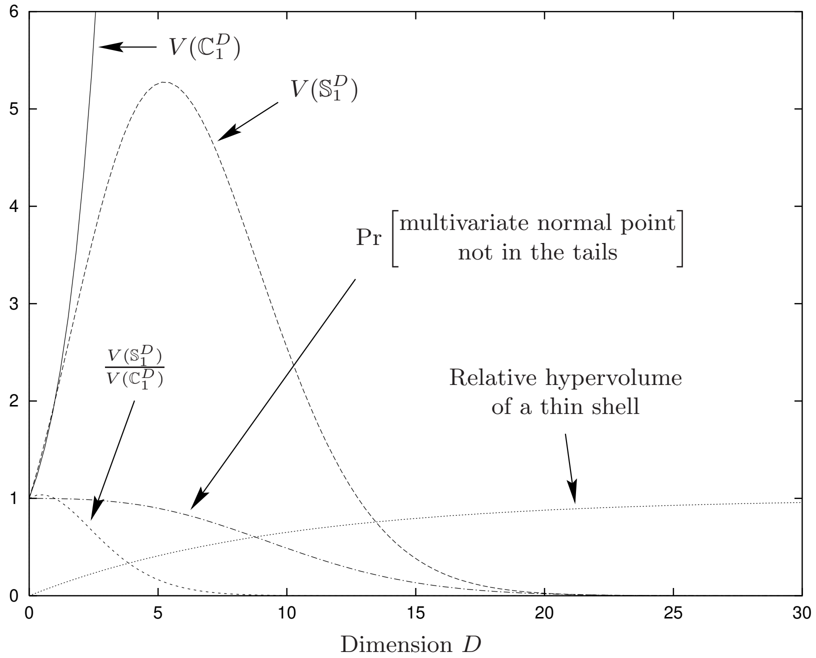


Figure 4.4: Dependence of several geometric quantities with the dimension (only natural numbers $D = 1, 2, \dots$ are meaningful). See the main text for an explanation.

because if $\mathbf{t} = (t_1, \dots, t_D)$ is distributed as a standard normal, then x_i, $i = 1, \dots, D$ are univariate standard normal and $\|\mathbf{t}\|^2 = \sum_{d=1}^{D} t_d^2$ is distributed as a χ^2 distribution with D degrees of freedom. Equation 4.1 gives the probability that a random point will not fall in the tails, i.e., that it will fall in the medium- to high-density region. Figure 4.4 shows this probability for $\epsilon = 0.01$ (a radius of 3 standard deviations) and several dimensions: notice how around $D = 5$ the probability mass of a multivariate normal begins a rapid migration into the extreme tails. In very high dimensions the entire sample will be in the tails!

Diagonals in hyperspace (Scott, 1992): consider the hypercube $[-1, 1]^D$ and let any of the diagonal vectors from the centre to a corner be denoted by $\mathbf{v}$. Then $\mathbf{v}$ is one of the 2^D vectors of the form $(\pm 1, \pm 1, \dots, \pm 1)^T$. The angle between a diagonal vector $\mathbf{v}$ and a coordinate axis $\mathbf{e}_d$ is given by

$$\cos\theta_D = \frac{\mathbf{v}\mathbf{e}_d}{\|\mathbf{v}\|\,\|\mathbf{e}_d\|} = \frac{\pm 1}{\sqrt{D}} \xrightarrow[D\to\infty]{} 0.$$

Thus, the diagonals are nearly orthogonal to all coordinate axes for large D.

Pairwise scatter diagrams essentially project the multivariate data onto all the 2-dimensional coordinate planes. Hence, any data cluster lying near a diagonal in hyperspace will be mapped into the origin in every paired scatterplot, while a cluster along a coordinate axis will be visible in some plot. Thus the choice of coordinate systems is critical in data analysis: 1- or 2-dimensional intuition is valuable but not infallible when continuing on to higher dimensions.

4.4 The intrinsic dimension of a sample

Consider a certain phenomenon governed by L independent variables. In practice, this phenomenon will actually appear as having (perhaps many) more degrees of freedom due to the influence of a variety of uncontrolled factors: noise, imperfection in the measurement system, addition of irrelevant variables, etc. However, provided this influence is not too strong as to completely mask the original structure, in principle we should be able to "filter" it out and recover the original variables or an equivalent set of them. We define the **intrinsic dimension**[5] of a phenomenon as the number of independent variables that explain satisfactorily that phenomenon. From a geometrical point of view, the intrinsic dimensionality would be the dimension L of the manifold that approximately embeds the sample data in a D-dimensional Euclidean space ($D > L$). The vagueness of the words *satisfactory* and *approximately* is intended, since the intrinsic dimensionality of a sample depends on the criteria that the user applies to the particular problem at hand, such as smoothness of the manifold, effect of noise, etc. A priori a discrete sample has dimension zero, but it is possible to make a manifold of any dimension pass through it. So the sample may have dimension one according to one criterion, dimension two according to a different one and so on. For example, in fig. 4.5 a one-dimensional manifold (the dotted curve) is forced to interpolate a set of points which naturally would seem to lie on a two-dimensional manifold (the shaded area). Thus, the problem of inferring the intrinsic dimensionality of a sample is ill-posed and requires of prior information to be solved.

The determination of the intrinsic dimensionality of a process given a sample of it is central to the problem of dimensionality reduction, because knowing it would eliminate the possibility of over- or underfitting. All the dimensionality reduction methods discussed in this thesis take the intrinsic dimensionality as a parameter to be given by the user; a trial-and-error process is necessary to obtain a satisfactory value for it (aided by model selection techniques such as cross-validation). In some practical applications, domain information may give insight into the intrinsic dimensionality. For probabilistic methods, such as latent variable models, a Bayesian approach can help to determine the intrinsic dimensionality. For example, one could consider the intrinsic dimensionality L as a trainable parameter over which a prior distribution is placed, perhaps uniform in $\{1, 2, \dots, D\}$ (to reflect lack of information), or perhaps decreasing monotonically with increasing dimension (to favour the reduction of dimensionality). The posterior mode of the intrinsic dimensionality given the data sample could be chosen as the optimal dimensionality if a single value is desired (although, strictly, inferences should use the posterior parameter distribution, including L, rather than a single value). This has the problem that the rest of the parameters depend on L. A possible strategy is the use of a hierarchical prior model, as Richardson and Green (1997) have done to learn the number of components (and the rest of the parameters) of a finite mixture. Another possibility is to use the automatic relevance determination (ARD) framework (MacKay, 1995b), as Bishop (1999) has proposed for the probabilistic PCA and PCA mixture models described in sections 2.6.2 and 2.7. Although the computations involved in the Bayesian approach are very complicated and no exact treatment is possible even for the simplest models, this is a promising research direction.

Finally, we show two unusual cases of manifolds. First consider the sample in figure 4.6. Without further information one could say that it corresponds to a two-dimensional manifold on the left side and to a one-dimensional manifold on the right side (whatever that may mean). It actually corresponds to a one-dimensional distribution with variable noise (much higher on the left side), obtained from a speech enhancement application analysed by Xie and van Compernolle (1996). They consider that the observed noisy speech Y is due to independent noise E corrupting the clean speech S additively in the spectral (log) domain, with S and E distributed normally in a frame basis. Thus $10^{\frac{Y}{10}} = 10^{\frac{S}{10}} + 10^{\frac{E}{10}}$ or $y = s + e$, where y, s and e are the log-magnitudes of the observed speech, clean speech and noise, respectively, and s and e follow a log-normal distribution. Fig. 4.6 shows a synthetically generated sample where $S \sim \mathcal{N}(\mu = 10, \sigma = 17)$ and $E \sim \mathcal{N}(\mu = 0, \sigma = 3)$. The solid line corresponds to the minimum mean squared error estimate, the posterior mean $\mathrm{E}\{s|y\}$.

The second case may be unlikely to arise in a practical problem, but nonetheless it has a theoretical interest. Figure 4.7 shows the first 5 approximations to a *space-filling curve*, the Hilbert curve. While each curve is one-dimensional, it can be proven that the limit to which this sequence of curves converges exists and is the square: a two-dimensional manifold. That is, a space-filling curve is a continuous map of an interval of the real line on a rectangle of $\mathbb{R}^2$ or some other higher-dimensional manifold. Other fractal curves, such as the Koch snowflake curve, have even a non-integral (Hausdorff) dimension between 1 and 2 (Barnsley, 1988; Peitgen et al., 1992).

[5] We will not attempt to define formally the concept of dimension, for which, in fact, many different mathematical definitions exist, each one trying to capture some desirable properties of the notion of dimension: topological dimension, covering dimension, Hausdorff dimension, fractal dimension... For our purposes, an intuitive idea of the concept of dimension will be enough. Falconer (1990) has more details about the definition of dimension.

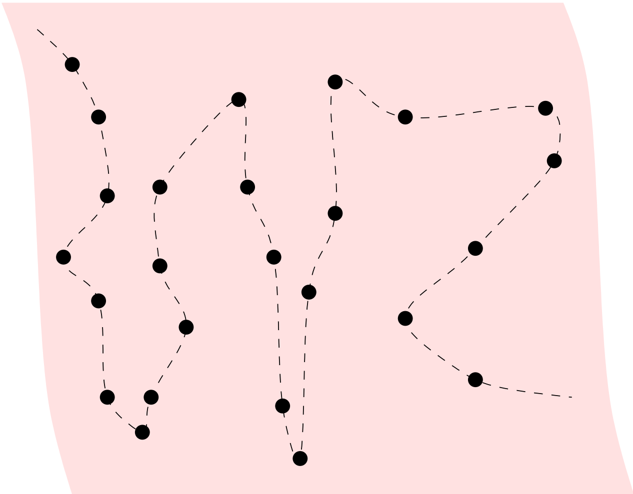


Figure 4.5: Curve or surface?

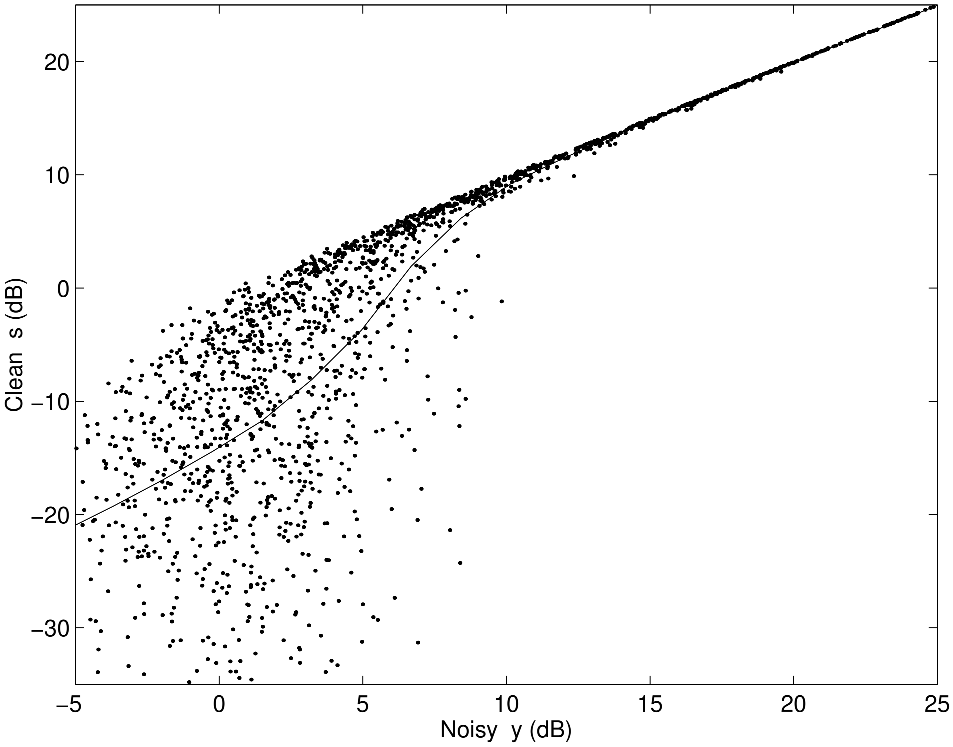


Figure 4.6: What is the intrinsic dimensionality of this sample? (adapted from Xie and van Compernolle, 1996 with permission from Elsevier Science).

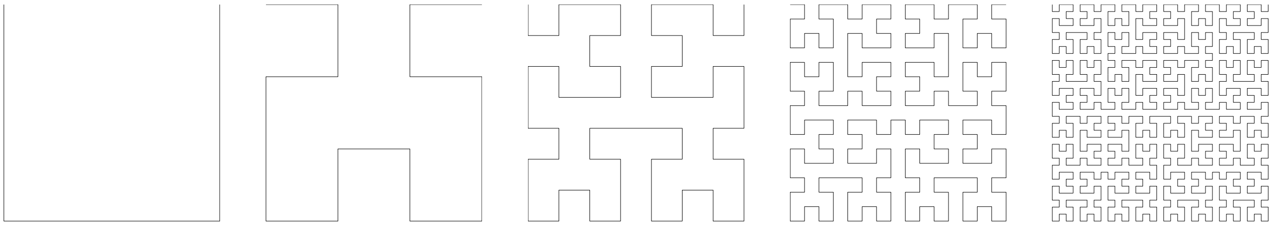


Figure 4.7: First 5 curves of the Hilbert space-filling curve sequence.

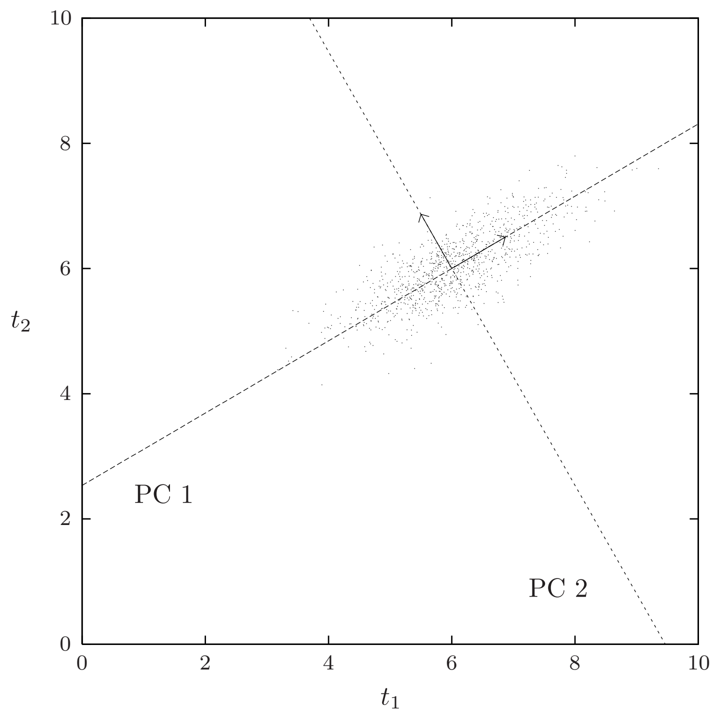


Figure 4.8: Bidimensional, normal point cloud with its principal components.

Generally, we know from set theory that we can map invertibly and continuously $\mathbb{R}^D$ into $\mathbb{R}$ or any interval of $\mathbb{R}$, i.e., $\operatorname{card}(\mathbb{R}^D) = \operatorname{card}(\mathbb{R}) = \operatorname{card}((0,1)) = \mathfrak{c}$ for any $D \in \mathbb{N}$ (Rotman and Kneebone, 1966); for example, using the diagonal Cantor construction: given $\mathbf{t} \in \mathbb{R}^D$, write each of its components $t_1, \dots, t_D$ as a binary expansion (which can be done in a unique way) and interleave the expansions to obtain the binary expansion of a number in $\mathbb{R}$. In principle, this would allow to find a nonlinear continuous mapping from $\mathbb{R}^D$ into $\mathbb{R}$ preserving all information: exact reduction of any dimensionality to $L = 1$! Of course, due to the finite precision of computers this is of no practical application, and even if it was possible, such a procedure would not help to understand the intrinsic dimensionality in the sense mentioned earlier. $\mathbb{R}^D$ and $\mathbb{R}$ may have the same cardinal, but they do not have the same dimension.

4.5 Principal component analysis

Principal component analysis (PCA) (Jackson, 1991; Jolliffe, 1986) is possibly the dimensionality reduction technique most widely used in practice, perhaps due to its conceptual simplicity, its analytical properties and the fact that relatively efficient algorithms (of polynomial complexity) exist for its computation. In signal processing it is known as the Karhunen-Loève transform.

Traditionally, PCA has been considered as a distribution-free linear dimensionality reduction technique, and that is the point of view that we follow in this section. However, it can be also seen as the maximum likelihood estimate of a specific latent variable model. We describe the probabilistic view of PCA in section 2.6.2.

Let us consider a sample $\{\mathbf{t}_n\}_{n=1}^N$ in $\mathbb{R}^D$ with mean $\overline{\mathbf{t}} \stackrel{\text{def}}{=} \frac{1}{N}\sum_{n=1}^N \mathbf{t}_n$ and covariance matrix $\boldsymbol{\Sigma} \stackrel{\text{def}}{=} \frac{1}{N}\sum_{n=1}^N (\mathbf{t}_n - \overline{\mathbf{t}})(\mathbf{t}_n - \overline{\mathbf{t}})^T$. The matrix $\boldsymbol{\Sigma}$ is symmetric semidefinite positive and admits a spectral decomposition

$$\boldsymbol{\Sigma} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^T$$

with $\mathbf{U} = (\mathbf{u}_1, \dots, \mathbf{u}_D)$ orthogonal, $\boldsymbol{\Lambda} = \operatorname{diag}(\lambda_1, \dots, \lambda_D)$ and $\lambda_d \geq 0$, $d = 1, \dots, D$. $\mathbf{u}_d$ is the normalised eigenvector of $\boldsymbol{\Sigma}$ associated with eigenvalue λ_d. The principal component transformation $\mathbf{x} = \mathbf{U}^T(\mathbf{t} - \overline{\mathbf{t}})$ yields a reference system in which the sample has mean $\mathbf{0}$ and diagonal covariance matrix $\boldsymbol{\Lambda}$ containing the eigenvalues of $\boldsymbol{\Sigma}$: the variables are now uncorrelated. Figure 4.8 shows an example. In this new reference system one can discard the variables with small variance, i.e., project on the subspace spanned by the first L principal components, and obtain a good approximation (the best linear one in the least squares sense) to the original sample: $\mathbf{x} = \mathbf{U}_L^T(\mathbf{t} - \overline{\mathbf{t}})$ with $\mathbf{U}_L = (\mathbf{u}_1, \dots, \mathbf{u}_L)$.

The key property of principal component analysis is that it attains the best dimensionality reduction linear map $\mathbf{t} \in \mathbb{R}^D \longrightarrow \mathbf{x} \in \mathbb{R}^L$ in the senses of:

- maximal variance in the projected space subject to orthonormality (Hotelling, 1933):

$$\max_{\mathbf{A}^T\mathbf{A}=\mathbf{I}} \left\{ \operatorname{tr}\left(\frac{1}{N} \sum_{n=1}^{N} \mathbf{x}\mathbf{x}^T \right) \right\} = \sum_{l=1}^{L} \lambda_l \text{ with } \mathbf{x} \stackrel{\text{def}}{=} \mathbf{A}^T(\mathbf{t} - \bar{\mathbf{t}}), \text{ attained at } \mathbf{A} = \mathbf{U}_L$$

- L_2-norm, or least squared sum of errors of the reconstructed data (Pearson, 1901):

$$\min_{\mathbf{A}} \left\{ \sum_{n=1}^{N} \left\| \mathbf{t} - \mathbf{t}^* \right\|^2 \right\} = N \sum_{l=1}^{L} \lambda_l \text{ with } \mathbf{t}^* \stackrel{\text{def}}{=} \mathbf{A}\mathbf{A}^T(\mathbf{t} - \bar{\mathbf{t}}), \text{ attained at } \mathbf{A} = \mathbf{U}_L$$

- maximal mutual information (assuming the data vectors $\mathbf{t}$ are distributed normally) between the original vectors $\mathbf{t}$ and their projections $\mathbf{x}$ (Kapur, 1989, pp. 502–504; Cover and Thomas, 1991):

$$\max_{\mathbf{A}} \left\{ I(\mathbf{t}; \mathbf{x}) \right\} = \frac{1}{2} \ln \left(\prod_{l=1}^{L} 2\pi e \lambda_l \right) \text{ with } \mathbf{x} \stackrel{\text{def}}{=} \mathbf{A}^T(\mathbf{t} - \bar{\mathbf{t}}), \text{ attained at } \mathbf{A} = \mathbf{U}_L$$

where $\lambda_1 > \cdots > \lambda_L$ are the first L eigenvalues of the covariance matrix.

Geometrically, the hyperplane spanned by the first L principal components is the regression hyperplane that minimises the orthogonal distances to the data. In this sense, PCA is a symmetric regression approach, as opposed to standard linear regression, which points one component as response variable and the rest as predictors. In fact, the principal component subspace is a principal manifold in the sense of section 4.8.

The first principal components are often used as starting points for other algorithms, such as projection pursuit regression, principal curves, Kohonen's maps or the generalised topographic mapping. PCA is also useful as a first, coarse dimensionality reduction stage where a lot of unnecessary directions of negligible variance are discarded, particularly in very high-dimensional data (for example, when each data vector represents a bitmapped image or a sampled time-varying curve).

A number of numerical techniques exist for finding all or the first few eigenvalues and eigenvectors of a square, symmetric, semidefinite positive matrix (the covariance matrix) in time $\mathcal{O}(D^3)$: singular value decomposition, Cholesky decomposition, etc. (Wilkinson, 1965; Golub and van Loan, 1996; Press et al., 1992). When the covariance matrix, of order $D \times D$, is too large to be explicitly computed one could use neural network techniques (section 4.5.1), some of which do not require more memory space other than the one needed for the data vectors and the principal components themselves. Unfortunately, these techniques (usually based on a gradient descent method) are much slower than traditional methods.

PCA can also be computed from the $N \times N$ scalar-product matrix $\frac{1}{N}\mathbf{T}\mathbf{T}^T$, where $\mathbf{T} \stackrel{\text{def}}{=} (\mathbf{t}_1, \dots, \mathbf{t}_N)^T$ (assuming centred vectors to simplify the notation). This is because $\frac{1}{N}\mathbf{T}\mathbf{T}^T$ has the same nonzero eigenvalues as $\mathbf{\Sigma} = \frac{1}{N}\mathbf{T}^T\mathbf{T}$, as can be seen from the singular value decomposition of $\mathbf{T}$:

$$\mathbf{T} = \mathbf{V}\mathbf{S}\mathbf{U}^T \Rightarrow \begin{cases} \mathbf{T}\mathbf{T}^T = \mathbf{V}\mathbf{S}\mathbf{S}^T\mathbf{V}^T \\ \mathbf{T}^T\mathbf{T} = \mathbf{U}\mathbf{S}^T\mathbf{S}\mathbf{U}^T \end{cases}$$

where $\mathbf{U}_{D\times D}$ and $\mathbf{V}_{N\times N}$ are orthogonal and $\mathbf{S}_{N\times D}$ has the singular values along its diagonal and zeroes elsewhere. The nonzero eigenvalues of $\mathbf{T}\mathbf{T}^T$ and $\mathbf{T}^T\mathbf{T}$ are the squared singular values. Using the scalar-product matrix instead of the covariance matrix is faster when the number of data points is smaller than the dimensionality, $N < D$, as is often the case in image processing applications, such as face recognition (Sirovich and Kirby, 1987). This is one of the bases of the kernel PCA method described in section 4.5.4.

The disadvantage of PCA is that it is only able to find a linear subspace and thus cannot deal properly with data lying on nonlinear manifolds. When the data is clustered, it can be more convenient to apply PCA locally (section 4.7).

The number of principal components to keep is a tricky question. Some rules of thumb are applied in practice, usually based on the *scree plot* (plot of the cumulative eigenvalue sum versus the number of components), such as to eliminate components whose eigenvalues are smaller than a fraction of the mean eigenvalue, or to keep as many as necessary to explain a certain fraction of the total variance, or to find where the curve

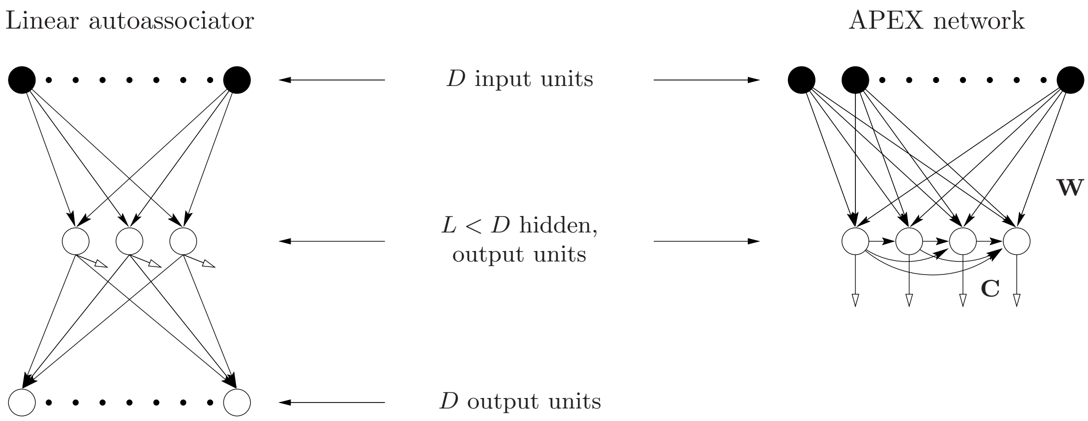


Figure 4.9: Two examples of neural networks able to perform a principal component analysis of its training set: left, a linear autoassociator, trained by backpropagation; right, the APEX network, with Hebbian weights **W** and anti-Hebbian, decorrelating weights **C**. In both cases, the number of hidden units determines how many principal components are kept.

changes its slope abruptly (Jolliffe, 1986). From a statistical goodness of fit point of view, one can also test[6] the hypothesis that the data covariance can be explained by the principal components (H_0) against the alternative that the covariance is unconstrained (H_1), although this procedure does not really tell when to stop adding components (Bartholomew, 1987, pp. 46–47 or Everitt, 1984, pp. 21–22).

The fact is that if the scree plot does not have a sharp decrease at some cutoff number of eigenvalues[7], it is very likely that the data cannot be modelled with a few linear variables, in which case it seems pointless to try hard to obtain the "right" number of eigenvalues and let alone to interpret them (see also section 2.8.1): the model is just wrong. For linear reconstruction purposes PCA is a legitimate technique and the number of eigenvalues L to use can be determined objectively from $\sum_{l=1}^{L} \lambda_l / \sum_{l=1}^{D} \lambda_l \leq 0.9$ (for 10% error, say). As such its use is recommended for (preliminary) feature extraction.

Estimating the intrinsic dimensionality of a data sample using PCA will result in a larger dimensionality than the real one if the data lies on a nonlinear manifold (e.g. a circle is a one-dimensional manifold but it cannot be embedded in a one-dimensional linear manifold). It is also clear that the number of nonzero eigenvalues of the data covariance matrix (with respect to some threshold) gives an upper bound for the intrinsic dimensionality.

4.5.1 Principal component analysis networks

There exist several neural network architectures capable to extract principal components, some of which are represented in fig. 4.9. They can be classified in:

- Autoassociators (also called autoencoders or bottleneck networks), which are linear two-layer perceptrons with D inputs, L hidden units and D outputs, trained to replicate the input in the output layer minimising the squared sum of errors, typically with backpropagation. Bourlard and Kamp (1988) and Baldi and Hornik (1989) showed that this network finds a basis of the subspace spanned by the first L principal components, not necessarily coincident with them.

- Networks based on Oja's rule (Oja, 1992) with some kind of decorrelating device, e.g. the network of Földiák (1989), the APEX network of Kung et al. (1994) or the generalised Hebbian algorithm of Sanger (1989).

Diamantaras and Kung (1996) review PCA networks in detail.

[6]This test was originally developed for factor analysis, but it can be readily extended to PCA via its interpretation as a latent variable model (section 2.6.2).

[7]Also, consider K eigenvalues of slowly decreasing value: $\lambda_i \geq \lambda_{i+1} \geq \cdots \geq \lambda_{i+K-1}$. They determine a slightly elongated K-dimensional ellipsoid with semiaxes of lengths $\sqrt{\lambda_i}, \dots, \sqrt{\lambda_{i+K-1}}$ in the K-dimensional subspace defined by their associated eigenvectors. Any direction in that subspace will have a variance in $[\lambda_{i+K-1}, \lambda_i]$ and thus all directions in that subspace (not just along the eigenvectors) are approximately equivalent (in the extreme case where $\lambda_i = \cdots = \lambda_{i+K-1}$ they define a hypersphere and all directions are strictly equivalent).

4.5.2 Nonlinear autoassociators

An obvious extension to linear autoassociators is the inclusion of nonlinear activation functions and several layers (see fig. 4.10). The representation obtained in the unit activations of one of the hidden layers (with $L < D$ units) can be taken as the reduced-dimension representative (the middle layer in the figure). This defines a dimensionality reduction mapping $\mathbf{F}$ and a reconstruction mapping $\mathbf{f}$ (sometimes called recognition and generative mappings, respectively). As in the linear case, the net is trained to replicate its input at the output layer in the least-squares sense by backpropagation or other method. Given their conceptual simplicity and the appeal of the idea of "squashing the input through a bottleneck," nonlinear autoassociators were used for dimensionality reduction quite early, e.g. by Saund (1989), Fleming and Cottrell (1990), Kramer (1991) or DeMers and Cottrell (1993), among others.

Bourlard and Kamp (1988) show that nonlinear autoassociators with only one hidden layer are no better than linear ones, i.e., than PCA. But clearly, nonlinear autoassociators with three hidden layers (as in fig. 4.10) must have, at least potentially, superior ability than linear ones for dimensionality reduction, since both $\mathbf{F}$ and $\mathbf{f}$ become universal approximators (Scarselli and Tsoi, 1998). Indeed, they have outperformed PCA in some applications. Malthouse (1998) cites several of them, particularly in chemometrics, where nonlinear autoassociators were popularised by Kramer (1991). Surprisingly, the approach has not been widely accepted as a dimensionality reduction method. Several reasons have been proposed for this from an empirical perspective:

- Nonlinear autoassociators are very slow to train. Rögnvaldsson (1994) has offered the following explanation for this: the risk that the Hessian of the error function of a multilayer perceptron is ill-conditioned grows with the number of layers. An ill-conditioned Hessian makes the error surface very flat and learning becomes very slow both with backpropagation and with second-order methods.

- Training with various local optimisers (e.g. gradient descent, conjugate gradient descent, stochastic gradient descent or a quasi-Newton method) very often results in local minima with a higher error than PCA (Kambhatla and Leen, 1997). Using neuronal spike trains data, Fotheringhame and Baddeley (1997) observed backpropagation to be slow and unreliable, while conjugate gradient descent worked better than PCA with synthetic data but not with real data.

- Several researchers, e.g. Fleming and Cottrell (1990), have observed that, at the end of the training (with backpropagation), the unit activations often concentrate on the linear region of the nonlinearity and therefore there is little difference with PCA networks.

These reasons seem to point to deficiencies in the optimisation algorithm rather than in the class of representations attainable. That is, a nonlinear autoassociator is potentially able to represent complex manifolds, but, for most initial values of the parameters, a local optimiser will end in a bad local minimum rather than in one of the good ones.

Some theoretical results are known for nonlinear autoassociators. Both the dimensionality reduction mapping $\mathbf{F}$ and the reconstruction mapping $\mathbf{f}$ are continuous if the unit nonlinearities are continuous (as is often the case, e.g. with the sigmoid). Malthouse (1998) uses this fact to show that nonlinear autoassociators can approximate neither self-intersecting manifolds nor discontinuous manifolds. He also notices that they cannot implement a dimensionality reduction mapping based on orthogonal projection on the closest point of the manifold (as does happen with principal curves, section 4.8): this would lead to a discontinuity in the dimensionality reduction mapping for those data points which are equidistant from different points of the hidden-layer manifold.

Nonlinear autoassociators have had some empirical success in other kinds of pattern recognition problems. Wiles et al. (1996) report finding good solutions to the travelling salesman problem[8], while Japkowicz et al. (2000) claim better performance than with linear autoassociators for classification in nonlinear multimodal domains.

4.5.3 Other linear transformations

Here we mention other usual linear transformations of the data, their relation to PCA and their effect on the covariance matrix. Consider again the sample $\{\mathbf{t}_n\}_{n=1}^N$ in $\mathbb{R}^D$ with mean $\bar{\mathbf{t}} \stackrel{\text{def}}{=} \frac{1}{N}\sum_{n=1}^N \mathbf{t}_n$ and covariance matrix $\boldsymbol{\Sigma} \stackrel{\text{def}}{=} \frac{1}{N}\sum_{n=1}^N (\mathbf{t}_n - \bar{\mathbf{t}})(\mathbf{t}_n - \bar{\mathbf{t}})^T$. Let $\mathbf{A}_{D\times L} = (\mathbf{a}_1, \dots, \mathbf{a}_L)$ with $\mathbf{a}_l \in \mathbb{R}^D$ a set of L projection directions.

[8] Dimensionality reduction here is conceptualised as going from the D-dimensional map of D cities (coded as 1-of-D) to a one-dimensional ordered list in which neighboring cities are listed close together.

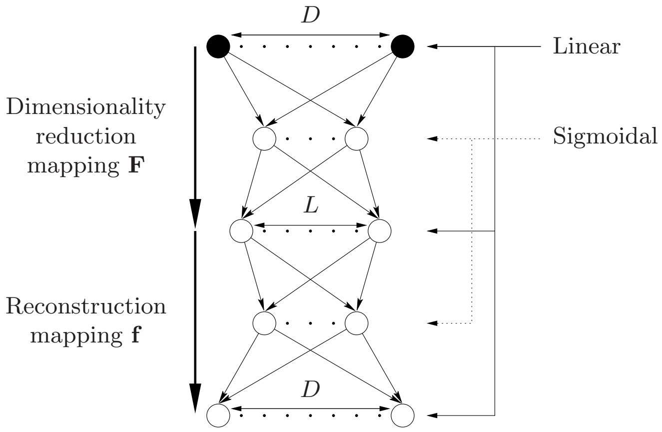


Figure 4.10: Nonlinear autoassociator, implemented as a four-layer nonlinear perceptron where $L < D$ and $\mathbf{t}^* \stackrel{\text{def}}{=} \mathbf{f}(\mathbf{F}(\mathbf{t}))$.

Transformation f	$\mathbf{t}' = f(\mathbf{t})$	$\overline{\mathbf{t}}'$	$\mathbf{\Sigma}'$	$\mathbf{U}'$	$\mathbf{u}'$	$\mathbf{\Lambda}'$	λ'
Translation	$\mathbf{t}+\mathbf{d}$	$\overline{\mathbf{t}}+\mathbf{d}$	$\mathbf{\Sigma}$	$\mathbf{U}$	same	$\mathbf{\Lambda}$	same
Rotation	$\mathbf{Rt}$	$\mathbf{R}\overline{\mathbf{t}}$	$\mathbf{R\Sigma R}^T$	$\mathbf{RU}$	rotated	$\mathbf{\Lambda}$	same
Axis scaling	$\mathbf{Dt}$	$\mathbf{D}\overline{\mathbf{t}}$	$\mathbf{D\Sigma D}$	$\neq$	$\neq$	$\neq$	$\neq$
Uniform axis scaling	$a\mathbf{t}$	$a\overline{\mathbf{t}}$	$a^2\mathbf{\Sigma}$	$\mathbf{U}$	same	$a^2\mathbf{\Lambda}$	scaled
Affine	$a\mathbf{t}+\mathbf{d}$	$a\overline{\mathbf{t}}+\mathbf{d}$	$a^2\mathbf{\Sigma}$	$\mathbf{U}$	same	$a^2\mathbf{\Lambda}$	scaled
Centring	$\mathbf{t}-\overline{\mathbf{t}}$	$\mathbf{0}$	$\mathbf{\Sigma}$	$\mathbf{U}$	same	$\mathbf{\Lambda}$	same
PCA	$\mathbf{U}^T(\mathbf{t}-\overline{\mathbf{t}})$	$\mathbf{0}$	$\mathbf{\Lambda}$	$\mathbf{I}$	$\{\mathbf{e}_d\}_{d=1}^D$	$\mathbf{\Lambda}$	same
Sphering	$\mathbf{\Sigma}^{-1/2}(\mathbf{t}-\overline{\mathbf{t}})$	$\mathbf{0}$	$\mathbf{I}$	$\mathbf{I}$	$\{\mathbf{e}_d\}_{d=1}^D$	$\mathbf{I}$	1

Table 4.2: Transformation of the covariance matrix under transformations on the sample. $\overline{\mathbf{t}}$ is the sample mean and $\mathbf{\Sigma}$ the sample covariance, with spectral decomposition $\mathbf{U\Lambda U}^T$. $a \in \mathbb{R}-\{0\}$, $\mathbf{d} \in \mathbb{R}^D$, $\mathbf{D} = \operatorname{diag}(d_1, \dots, d_D)$ is diagonal, $\mathbf{R}$ is orthogonal, $\mathbf{e}_d$ is the unit vector in the direction of t_d, the primes denote the new entity after the transformation f and the symbol $\neq$ is used to indicate that the subsequent transformation is complex (not obviously related to f).

Centring is a procedure by which we translate the sample so that its mean is at the origin: $\mathbf{t}' = \mathbf{t}-\overline{\mathbf{t}} \Rightarrow \overline{\mathbf{t}}' = \mathbf{0}$. Centring is inherited by any set of projections: $\mathrm{E}\{\mathbf{t}\} = \mathbf{0} \Rightarrow \mathrm{E}\{\mathbf{A}^T\mathbf{t}\} = \mathbf{A}^T\,\mathrm{E}\{\mathbf{t}\} = \mathbf{0}$.

Scaling achieves unit variance in each axis by dividing componentwise by its standard deviation $\sigma_d \stackrel{\text{def}}{=} \sqrt{(\mathbf{\Sigma})_{dd}}$: $\mathbf{t}' = \operatorname{diag}\left(\sigma_1^{-1}, \dots, \sigma_D^{-1}\right)\mathbf{t} \Rightarrow (\mathbf{\Sigma}')_{dd} = 1\ \forall d = 1, \dots, D$.

Sphering is an affine transformation that converts the covariance matrix (of the centred sample) into a unit variance matrix, thus destroying all the first- and second-order information of the sample: $\mathbf{t}' = \mathbf{\Sigma}^{-1/2}(\mathbf{t}-\overline{\mathbf{t}}) \Rightarrow \overline{\mathbf{t}}' = \mathbf{0}$ and $\mathbf{\Sigma}' = \mathbf{I}$, where[9] $\mathbf{\Sigma}^{-1/2} = \mathbf{\Lambda}^{-1/2}\mathbf{U}^T$. Sphering is inherited by any orthogonal set of projections: $\mathrm{E}\{\mathbf{t}\} = \mathbf{0}$ and $\operatorname{cov}\{\mathbf{t}\} = \mathbf{I} \Rightarrow \operatorname{cov}\{\mathbf{A}^T\mathbf{t}\} = \mathrm{E}\{(\mathbf{A}^T\mathbf{t})(\mathbf{A}^T\mathbf{t})^T\} = \mathbf{A}^T\,\mathrm{E}\{\mathbf{tt}^T\}\,\mathbf{A} = \mathbf{A}^T\mathbf{A} = \mathbf{I}$.

PCA is another affine transformation that converts the covariance matrix (of the centred sample) into a diagonal matrix, thus decorrelating the variables but preserving the variance information: $\mathbf{t}' = \mathbf{U}^T(\mathbf{t}-\overline{\mathbf{t}}) \Rightarrow \overline{\mathbf{t}}' = \mathbf{0}$, $\mathbf{\Sigma}' = \mathbf{\Lambda}$.

PCA and sphering are both translation and rotation invariant, i.e., applying a translation and a rotation to the data and then performing PCA or sphering produces the same results as performing them on the original data. Table 4.2 summarises the effect of linear transformations on the covariance matrix.

[9] $\mathbf{\Sigma}^{-1/2}$ is defined as a matrix $\mathbf{B}$ such that $\mathbf{B}^T\mathbf{B} = \mathbf{\Sigma}$ and so any matrix $\mathbf{B}' = \mathbf{RB}$ with $\mathbf{R}$ orthogonal is also valid. Canonically, we can take $\mathbf{\Sigma}^{-1/2} = \mathbf{\Lambda}^{-1/2}\mathbf{U}^T$.

4.5.4 Kernel PCA

Kernel PCA (Schölkopf et al., 1998) is an unsupervised feature extraction method closely related to PCA[10] that, given a data set $\{\mathbf{t}_n\}_{n=1}^N$ contained in an *input space* $\mathcal{T} \subset \mathbb{R}^D$, extracts up to $\max(N, D)$ features from a vector $\mathbf{t} \in \mathcal{T}$. It is based on the following ideas:

- Carrying out PCA on the dot-product matrix of the data points, as mentioned in section 4.5, which is an $N \times N$ symmetric matrix of rank smaller or equal than $\min(N, D)$.
- Nonlinearly mapping input vectors to a high-dimensional *feature space* $\mathcal{F}$, $\mathbf{\Phi} : \mathcal{T} \longrightarrow \mathcal{F}$, where standard PCA is performed—this requires dot products in $\mathcal{F}$. Each component of $\mathbf{\Phi}$ will be a particular real function of the variables $t_1, \dots, t_D$ and will give information on a particular relationship between those variables. To be able to account for many different relationships, the dimensionality of $\mathcal{F}$ will be extremely high, possibly a power of the dimensionality of $\mathcal{T}$. Standard PCA is recovered by taking $\mathbf{\Phi}$ as the identity function.
- Computing dot products in $\mathcal{F}$ via a kernel function in $\mathcal{T}$, $k : \mathcal{T} \times \mathcal{T} \longrightarrow \mathbb{R}$. This allows to perform PCA efficiently on the data set $\{\mathbf{\Phi}(\mathbf{t}_n)\}_{n=1}^N$ using the dot-product Gram matrix[11] $\mathbf{K} \stackrel{\text{def}}{=} \left(\mathbf{\Phi}(\mathbf{t}_m)^T \mathbf{\Phi}(\mathbf{t}_n)\right)_{mn}$ but never explicitly calculating $\mathbf{\Phi}$ for any point. The nonlinear nature of map $\mathbf{\Phi}$ means that the associated component analysis back in input space $\mathcal{T}$ is nonlinear.

Let us analyse more in detail the procedure:

PCA in feature space is easily seen to require the solution of the eigenvalue problem $N\lambda\mathbf{a} = \mathbf{K}\mathbf{a}$ for eigenvectors $\mathbf{a}$ and nonzero eigenvalues λ, which results in up to N nonzero eigenvalues and associated eigenvectors, normalised such that $\|\mathbf{a}_n\| = \lambda_n^{-1/2}$. This ensures that $\mathbf{v}_n \stackrel{\text{def}}{=} \sum_{m=1}^N a_{nm} \mathbf{\Phi}(\mathbf{t}_m)$ is a unit-norm eigenvector of the sample covariance matrix in feature space, $\mathbf{\Sigma}_{\mathcal{F}} \stackrel{\text{def}}{=} \frac{1}{N} \sum_{n=1}^N \mathbf{\Phi}(\mathbf{t}_n)\mathbf{\Phi}(\mathbf{t}_n)^T$. Orthogonal projection in $\mathcal{F}$ of a vector $\mathbf{\Phi}(\mathbf{t})$ on the principal component directions $\{\mathbf{v}_n\}_{n=1}^N$ is accomplished as usual via the scalar product and results in an associated set of nonlinear components in $\mathcal{T}$: $\mathbf{v}_n^T \mathbf{\Phi}(\mathbf{t})$ is the orthogonal projection of vector $\mathbf{\Phi}(\mathbf{t})$ on the nth principal component in $\mathcal{F}$, or the nonlinear projection of input vector $\mathbf{t}$ on the nth nonlinear component in $\mathcal{T}$.

Dot products in feature space Defining a kernel function $k(\mathbf{t}_1, \mathbf{t}_2) \stackrel{\text{def}}{=} \mathbf{\Phi}(\mathbf{t}_1)^T \mathbf{\Phi}(\mathbf{t}_2)$ creates a correspondence between kernel functions $k : \mathcal{T} \times \mathcal{T} \longrightarrow \mathbb{R}$ and nonlinear maps $\mathbf{\Phi} : \mathcal{T} \longrightarrow \mathcal{F}$. Rather than choosing $\mathbf{\Phi}$ and then defining k, which is straightforward but does not give an efficient form for k, one chooses k, which then implicitly defines $\mathbf{\Phi}$. But not every function k can be expressed as a dot product in some space $\mathcal{F}$. A way to select kernels is given by Mercer's theorem of functional analysis (Courant and Hilbert, 1953), which states a necessary condition for a kernel to be expressible as a dot product:

Theorem 4.5.1 (Mercer). *If k is the continuous kernel of an integral operator $K : \mathcal{L}_2 \longrightarrow \mathcal{L}_2$, $Kf(\mathbf{t}_2) \stackrel{\text{def}}{=} \int_{\mathcal{T}} k(\mathbf{t}_1, \mathbf{t}_2) f(\mathbf{t}_1)\, d\mathbf{t}_1$, which is positive, $\int_{\mathcal{T}\times\mathcal{T}} f(\mathbf{t}_1) k(\mathbf{t}_1, \mathbf{t}_2) f(\mathbf{t}_2)\, d\mathbf{t}_1\, d\mathbf{t}_2 \geq 0\ \forall f \in \mathcal{L}_2$, then $k(\mathbf{t}_1, \mathbf{t}_2) = \sum_{n=1}^\infty \lambda_n \psi_n(\mathbf{t}_1)\psi_n(\mathbf{t}_2)$ with $\lambda_n \geq 0\ \forall n$.*

Such kernels are called Mercer or reproducing kernels. We can then define $\mathbf{\Phi}(\mathbf{t}) \stackrel{\text{def}}{=} \left(\sqrt{\lambda_1}\psi_1(\mathbf{t}), \sqrt{\lambda_2}\psi_2(\mathbf{t}), \dots\right)^T$ and so $k(\mathbf{t}_1, \mathbf{t}_2) = \mathbf{\Phi}(\mathbf{t}_1)^T \mathbf{\Phi}(\mathbf{t}_2)$. Table 4.3 gives examples of such kernels, all of which have also been used in support vector machines (Schölkopf et al., 1999a; Cristianini and Shawe-Taylor, 2000). The $\mathbf{\Phi}(\mathbf{t})$ mapping associated with the polynomial kernel is the collection of all possible pth degree ordered products of components of $\mathbf{t}$; e.g. for $p = 2$ we have $\mathbf{t} = (t_1\ t_2)^T$ and $k(\mathbf{t}_1, \mathbf{t}_2) = (\mathbf{t}_1^T \mathbf{t}_2)^2 = \mathbf{\Phi}(\mathbf{t}_1)^T \mathbf{\Phi}(\mathbf{t}_2)$ with $\mathbf{\Phi}(\mathbf{t}) = (t_1^2\ t_2^2\ t_1 t_2\ t_2 t_1)^T$. As an application, each t_d could be a pixel in a bitmapped image.

Evidently, the usual properties of PCA (section 4.5) remain in feature space: orthogonal directions of maximal variance, minimal L_2-reconstruction error and maximal mutual information with respect to the inputs (under Gaussian assumptions). But not anymore in input space, where the procedure is nonlinear. Kernel PCA will be unitary invariant (i.e., independent of the orthogonal coordinates used) if the kernel depends only on dot products and/or distances (as those of table 4.3 do), since the only operations on the input vectors that the whole procedure involves are computing the kernel values.

[10] In this section, we will use interchangeably the terms PCA, standard PCA or usual PCA to mean the linear PCA of section 4.5.

[11] The formula for $\mathbf{K}$ assumes centred vectors in $\mathcal{F}$. A corrected expression for uncentred vectors that does not explicitly compute $\mathbf{\Phi}$ at any point is easily derived from the one given.

Kernel	$k(\mathbf{t}_1, \mathbf{t}_2)$
Polynomial	$(\mathbf{t}_1^T \mathbf{t}_2)^p$
Gaussian (radial basis function)	$e^{-\frac{\|\mathbf{t}_1 - \mathbf{t}_2\|^2}{2\sigma^2}}$
Sigmoid	$\tanh(a\mathbf{t}_1^T \mathbf{t}_2 + b)$

Table 4.3: Examples of Mercer kernel functions for kernel PCA and support vector machines. In the polynomial kernel, $p = 1$ gives standard PCA.

Kernel PCA requires N computations of the kernel function where standard PCA required a dot product, which is not a significant extra cost. The real problem appears with large data sets ($N \gg D$: the normal situation in most applications) since the dot product matrix $\mathbf{K}_{N\times N}$ becomes huge. In this case, standard acceleration tricks can be applied, such as extracting only the first few components or estimating $\mathbf{K}$ from a small subset of the data.

For dimensionality reduction purposes, the projections on the principal components can be taken as features. While standard PCA can extract up to $\min(N, D)$ or fewer nonnull features (the rank of the sample covariance matrix in $\mathcal{T}$-space, $\mathbf{\Sigma}$), kernel PCA can obtain up to N, since in $\mathcal{F}$-space the rank of the dot product matrix $\mathbf{K}$ can be up to N if $\mathbf{\Phi}$ is nonlinear. A natural definition of reconstruction would be based on the usual orthogonal projection on the first principal components in feature space, but in general not every point in the subspace spanned by those principal components will have a preimage in input space! That is, even though $\{\mathbf{\Phi}(\mathbf{t}_n)\}_{n=1}^N$ have by definition preimages $\{\mathbf{t}_n\}_{n=1}^N$, a point $\mathbf{v} \in \operatorname{span}\left\{\{\mathbf{\Phi}(\mathbf{t}_n)\}_{n=1}^N\right\}$ may not have one $\mathbf{t} \in \mathcal{T}$ with $\mathbf{\Phi}(\mathbf{t}) = \mathbf{v}$. This requires to define an approximate (in some sense) reconstruction mapping in input space. Schölkopf et al. (1999b) give an algorithm to find a close preimage in the L_2 sense. They approximate $\mathbf{v} = \sum_{n=1}^N a_n \mathbf{\Phi}(\mathbf{t}_n)$ by a multiple (for computational considerations) of a vector $\mathbf{\Phi}(\mathbf{t})$ on the image of the input space:

$$\min_{\mathbf{t}\in\mathcal{T}} \left\| \frac{\mathbf{v}^T \mathbf{\Phi}(\mathbf{t})}{\mathbf{\Phi}(\mathbf{t})^T \mathbf{\Phi}(\mathbf{t})} \mathbf{\Phi}(\mathbf{t}) - \mathbf{v} \right\| \Longleftrightarrow \max_{\mathbf{t}\in\mathcal{T}} \frac{(\mathbf{v}^T \mathbf{\Phi}(\mathbf{t}))^2}{\mathbf{\Phi}(\mathbf{t})^T \mathbf{\Phi}(\mathbf{t})}.$$

The maximisation can be carried out with standard methods or, for kernels that satisfy $k(\mathbf{t}_1, \mathbf{t}_2) = \kappa(\|\mathbf{t}_1 - \mathbf{t}_2\|^2)$ (e.g. the Gaussian kernel), with a fixed-point iteration method: taking the gradient of $(\mathbf{v}^T \mathbf{\Phi}(\mathbf{t}))^2$ with respect to $\mathbf{t}$ and equating to 0 results in

$$\mathbf{t}^{(\tau+1)} = \frac{\sum_{n=1}^N a_n \kappa'(\|\mathbf{t}_n^{(\tau)} - \mathbf{v}\|)\mathbf{t}_n^{(\tau)}}{\sum_{n=1}^N a_n \kappa'(\|\mathbf{t}_n^{(\tau)} - \mathbf{v}\|)}$$

which for the Gaussian kernel is formally identical to the fixed-point iteration scheme we give in eq. (8.4) for finding the modes of a Gaussian mixture.

Experimentally, Schölkopf et al. (1998) show that kernel PCA outperforms standard PCA as feature extraction preprocessor for some classification tasks over a wide range of polynomial kernels and that the nonlinear components can be interpreted as separating or splitting clusters in a toy problem. Using approximate preimages, Schölkopf et al. (1999b) show improvements over PCA when denoising patterns if a large number of nonlinear components are used. The intuitive explanation they propose is that, since kernel PCA can extract more components (up to N) than PCA (up to D), it can provide a larger number of components that carry information about the structure in the data before they start to carry noise information as well (which always happens for high-order components). This would seem counterproductive for dimensionality reduction purposes, though.

In summary, kernel PCA has the following advantages:

- We obtain nonlinear components without any nonlinear optimisation, just computing standard PCA.
- We can extract up to N components, which is usually much more than what PCA allows (D if $N > D$), although this may not be conducing to dimensionality reduction. As in PCA, we do not need to restart the procedure if we want more or less components.
- For feature extraction, it can be readily used wherever PCA is.

And the following disadvantages:

- The procedure is sensitive to the kernel used (with different kernels resulting in different performances) but we do not know a priori what kernel to use. Thus, while kernel PCA is free from falling in local minima (a ubiquitous problem with methods based on nonlinear optimisation), we have a whole space of kernel functions to explore.
- While in feature space the geometrical interpretation of the principal components as orthogonal directions of maximal variance remains, in general and a priori we do not really know what the first components geometrically are in input space.
- For large data sets ($N \gg D$) the algorithm must be approximated to limit its computational requirements.
- There is an uncomfortable lack of natural dimensionality reduction and reconstruction mappings due to the fact that the principal component subspace in feature space may not have preimages in input space.

4.6 Projection pursuit

Principal component analysis selects linear projections of the data according to maximal variance subject to orthogonality. In general, one can search for projections that satisfy other properties. This is the basis of projection pursuit (Friedman and Tukey, 1974; Huber, 1985; Jones and Sibson, 1987; Ripley, 1996). Projection pursuit is an unsupervised technique that picks *interesting* low-dimensional linear orthogonal projections of a high-dimensional point cloud by optimising a certain objective function called **projection index**. It is typically used in exploratory data analysis to take profit of the human ability to discover patterns in low-dimensional (1- to 3D) projections: clustering, skewness, kurtosis, concentration along nonlinear manifolds, etc. That is, it is mainly used for visualisation; but it is equally useful for dimensionality reduction and regression, as shown below.

The (scaled) variable loadings (components of the projection vectors) that define the corresponding solution indicate the relative strength that each variable contributes to the observed effect. As in factor analysis, applying a rotation or similar transformations to the projections will produce the same picture but with an easier interpretation of the variable loadings.

Projections are smoothing operations in that structure can be obscured but never enhanced: any structure seen in a projection is a shadow of an actual structure in the original space. It is of interest to pursue the sharpest projections, that will reveal most of the information contained in the high-dimensional data distribution. We consider that a projection is interesting if it contains structure. Structure is considered as departure from normality, since:

- For fixed variance, the normal distribution has the least information, in both the senses of Fisher information and negative entropy (Cover and Thomas, 1991).
- For most high-dimensional clouds, most low-dimensional projections are approximately normal (Diaconis and Freedman, 1984).

Therefore, the normal distribution is the least structured (or least interesting) density.

For example, figure 4.11 shows two 2D projections of a 3D data set consisting of two clusters. The projection on the plane spanned by $\mathbf{e}_2$ and $\mathbf{e}_3$ is not very informative, as both clusters confuse in one; this projection nearly coincides with the one in the direction of the first principal component, which proves that the projection index of PCA (maximal variance; see section 4.6.1) is not a good indicator of structure. However, the projection on the plane spanned by $\mathbf{e}_1$ and $\mathbf{e}_2$ clearly shows both clusters.

The projection pursuit procedure will tend to identify outliers because the presence of the latter in a sample gives it the appearance of nonnormality. This sometimes can obscure the clusters or other interesting structure being sought. Also, the sample covariance matrix is strongly influenced by extreme outliers, so that methods relying on it (e.g. through data sphering) will not be robust against outliers. The effect of outliers can be partially tackled by robust sphering, e.g. using a trimming method (where all observations that lie farther than a threshold from the mean are deleted).

4.6.1 The projection index

A **projection index** I is a real functional on the space of distributions on $\mathbb{R}^L$:

$$I: \; p \in L_2(\mathbb{R}^L) \longrightarrow I(p) \in \mathbb{R}.$$

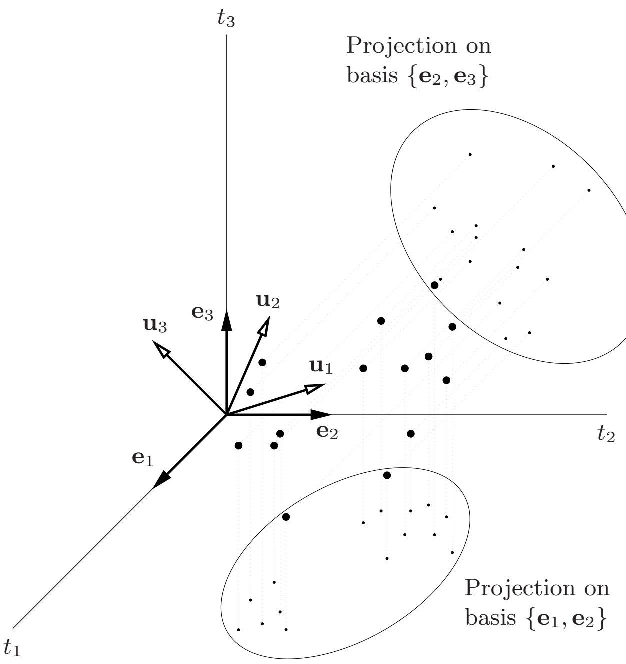


Figure 4.11: Two-dimensional projections of a three-dimensional data set. $\{\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3\}$ are the principal component directions.

p is the pdf of an L-dimensional random variable $\mathbf{x} \stackrel{\text{def}}{=} \mathbf{A}^T\mathbf{t}$, itself the projection (of matrix $\mathbf{A}_{D\times L}$) of a D-dimensional random variable $\mathbf{t}$. Abusing of notation, we will write $I(\mathbf{x})$ instead of $I(p)$. We will consider only one-dimensional projections for simplicity, but the treatment is easily extendable to multidimensional projections.

Projection pursuit attempts to find projection directions for a given distribution which produce local maxima of I. To make the maximisation problem independent of the length of the projection vectors and to obtain uncorrelated directions, the directions are constrained to be unit length and mutually orthogonal (i.e., the column vectors of $\mathbf{A}$ must be orthonormal). The optimisation problem is then

$$\max I(\mathbf{A}) \text{ subject to } \mathbf{A}^T\mathbf{A} = \mathbf{I}.$$

In general, there will be several interesting projections, each showing different insight, which correspond to local optima of the projection index. In fact, a difficulty with many projection pursuit indices that has often been observed (Friedman, 1987; Ripley, 1996) is the existence of many local maxima, in particular with small data sets, which makes finding good, high maxima difficult with local optimisers.

Huber (1985) lists the following properties that a good projection index should satisfy:

- Have continuous first (at least) derivative, to allow the use of gradient methods.
- Be rapidly computable—as well as its derivative(s)—since the optimisation procedure requires evaluating it many times.
- Be invariant to all nonsingular affine transformations in the data space, to discover structure not captured by the correlation.
- Satisfy: $I(\mathbf{x}_1 + \mathbf{x}_2) \leq \max(I(\mathbf{x}_1), I(\mathbf{x}_2))$ because, by the central limit theorem, $\mathbf{x}_1 + \mathbf{x}_2$ must be more normal (less interesting) than the less normal of $\mathbf{x}_1$ and $\mathbf{x}_2$. It follows that $I(\mathbf{x}_1 + \cdots + \mathbf{x}_N) \leq I(\mathbf{x})$ if $\mathbf{x}_1, \dots, \mathbf{x}_N$ are copies of $\mathbf{x}$, and therefore $I(\mathcal{N}) \leq I(\mathbf{x})$ if $\mathcal{N}$ is normal.

4.6.1.1 Examples of projection indices

Consider a random variable $\mathbf{t}$ of expectation $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$.

- Average:

$$I(\mathbf{x}) \stackrel{\text{def}}{=} \mathrm{E}\left\{\mathbf{x}\right\}.$$

In this case $\max_{\|\mathbf{a}\|=1} I(\mathbf{a}^T\mathbf{t}) = \|\boldsymbol{\mu}\|$ for $\mathbf{a} = \boldsymbol{\mu}/\|\boldsymbol{\mu}\|$.

- Variance:

$$I(\mathbf{x}) \stackrel{\text{def}}{=} \operatorname{var}\{\mathbf{x}\}.$$

In this case $\max_{\|\mathbf{a}\|=1} I(\mathbf{a}^T\mathbf{t}) = \lambda_1$ for $\mathbf{a} = \mathbf{u}_1$, the largest eigenvalue of $\boldsymbol{\Sigma}$ and its associated normalised eigenvector, respectively. In other words, this index finds the first principal component. PCA is then a particular case of projection pursuit.

- Standardised absolute cumulants $k_m(\mathbf{x})$ (defined in section A.2):

$$I(\mathbf{x}) \stackrel{\text{def}}{=} \frac{|k_m(\mathbf{x})|}{k_2(\mathbf{x})^{m/2}} \qquad m > 2.$$

- Fisher information:

$$I(\mathbf{x}) \stackrel{\text{def}}{=} J(\boldsymbol{\theta}) \stackrel{\text{def}}{=} \mathrm{E}\left\{\left(\frac{\partial p(\mathbf{x};\boldsymbol{\theta})}{\partial\boldsymbol{\theta}}\right)\left(\frac{\partial p(\mathbf{x};\boldsymbol{\theta})}{\partial\boldsymbol{\theta}}\right)^T\right\}$$

where $p(\mathbf{x};\boldsymbol{\theta})$ depends on some parameters $\boldsymbol{\theta}$. $J(\boldsymbol{\theta} = \operatorname{cov}\{\mathbf{x}\})$ is minimised by the normal density (Huber, 1985).

- Negative Shannon entropy:

$$I(\mathbf{x}) \stackrel{\text{def}}{=} -h(\mathbf{x}) = \mathrm{E}\{\ln p(\mathbf{x})\} = \int p(\mathbf{x}) \ln p(\mathbf{x})\, d\mathbf{x}.$$

For fixed variance, this index is minimised by the normal density (Cover and Thomas, 1991). Its drawback is that it is difficult to compute, so that several approximations have been proposed. Jones and Sibson (1987) propose two ways to evaluate it:

  - By numerical integration or implementing it as a sample entropy: $\frac{1}{N}\sum_{n=1}^{N} \ln p(\mathbf{a}^T\mathbf{t}_n)$, where p is a nonparametric density estimate of the projected points $\mathbf{a}^T\mathbf{t}_n$. Both are very slow to compute.
  - Approximating p by an expansion in terms of the cumulants (skew and kurtosis); for 1D projections:

$$\int p \ln p \approx \frac{1}{12}\left(k_3^2 + \frac{1}{4}k_4^2\right).$$

Cumulant-based approximations to the differential entropy arise from a Gram-Charlier or Edgeworth polynomial expansion of the density (Kendall and Stuart, 1977) and have been also used in the context of independent component analysis (Comon, 1994). However, they give a poor approximation to the entropy, since the cumulants are much more sensitive to structure in the tails of the distribution than in its centre (a fact that is accentuated if the cumulants, being very sensitive to outliers, are estimated from finite samples). Also, the skew and kurtosis are not meaningful for multimodal distributions. For 1D projections, Hyvärinen (1998) gives a further approximation of the differential entropy based on a first-order approximation of the density of maximum entropy given some simple constraints. In section 8.7.6 we give upper and lower bounds for the entropy of a Gaussian mixture of known parameters.

Jones (1983) found little difference in practice between $\int p \ln p$ and $\int p^2$. However, for fixed variance the integral $\int p \ln p$ is minimised by the normal distribution, while $\int p^2$ is minimised by the Epanechnikov kernel (Silverman, 1986).

- The original univariate index of Friedman and Tukey (1974), of difficult optimisation and dependent on a parameter h, is large whenever many points are clustered in a neighbourhood of size h. Huber (1985) noted that this index is essentially based on $\int p^2 = \|p\|_2^2 = \mathrm{E}_p\{p(x)\}$.

- The univariate exploratory projection pursuit index of Friedman (1987):

$$I(x) \stackrel{\text{def}}{=} \int_{-1}^{1}\left(p(u) - \frac{1}{2}\right)^2 du = \int_{-1}^{1} p^2(u)\, du - \frac{1}{2}$$

where $u \stackrel{\text{def}}{=} 2\Phi(x) - 1 \in [-1, 1]$ and Φ is the standard normal cdf (the data are assumed to have been sphered). u is uniform in $[-1, 1]$ if x is standard normal in $(-\infty, \infty)$ and hence nonuniformity of u

will mean nonnormality of x. The index is minimised by the normal distribution and in practice it is implemented approximately by a Legendre polynomial expansion.

This index can be reexpressed by transforming back from the u variable to x as

$$I(x) = \int_{-\infty}^{\infty} \frac{(p(x) - \phi(x))^2}{2\phi(x)}\, dx$$

which suggests some variations, such as $\int (p - \phi)^2$ (Hall, 1989) or $\int (p - \phi)^2\phi$ (Cook et al., 1993).

- Intrator and Cooper (1992) present an objective function formulation of the BCM neuron (a synaptic modification rule proposed to explain visual cortical plasticity by Bienenstock et al., 1982) which is equivalent to a projection index that seeks multimodality.

Many other indices exist, often designed for one- or two-dimensional projections, that measure various kinds of departure from normality (e.g. Eslava and Marriott, 1994; Posse, 1990, 1995). Extension to high dimensions is often difficult.

Indices based on polynomial moments operate by approximating the density by a truncated series of orthogonal polynomials (Legendre, Hermite, etc.), such as the cumulant-based indices mentioned. They do not have to be recomputed at each step of the numerical procedure, as they can be derived for each projection direction from sufficient statistics of the original data set. However, they perform poorly unless high-order polynomials are used, since deviation from normality cannot be captured in second-order correlations, as mentioned in the discussion of independent component analysis (section 2.6.3).

4.6.2 Projection pursuit regression and density approximation

Low-dimensional projections of a distribution can be used as building blocks for function approximation. If the number of projections used L is smaller than the number of predictor variables D, then this combines dimensionality reduction (of the predictor variables) and regression. This idea is the basis of the following procedures:

Projection pursuit regression (Friedman and Stuetzle, 1981) is a nonparametric regression approach that works by additive composition, constructing an approximation to the desired response function by means of a sum of low-dimensional smooth functions, called *ridge functions*, that depend on low-dimensional projections through the data (we consider a one-dimensional function for simplicity):

$$f(\mathbf{t}) = \sum_{l=1}^{L} g_l(\mathbf{a}_l^T\mathbf{t}).$$

Algorithms exist to estimate the projection directions $\{\mathbf{a}_l\}_{l=1}^L$ and ridge functions $\{g_l\}_{l=1}^L$ nonparametrically (Friedman and Stuetzle, 1981) or parametrically, using a neural network (Hwang et al., 1994; Zhao and Atkeson, 1996).

Generalised additive models (Hastie and Tibshirani, 1990) are a particular case of projection pursuit regression where the projection directions are fixed to the coordinate axes:

$$f(\mathbf{t}) = \alpha + \sum_{d=1}^{D} g_d(t_d).$$

It is more easily interpretable and the individual components $g_d(t_d)$ can be plotted, but it is more restricted. Hastie and Tibshirani (1990) give a backfitting[12] algorithm to estimate $\{g_d\}_{d=1}^D$ nonparametrically.

Multivariate adaptive regression splines (MARS) (Friedman, 1991) are an extension to generalised additive models to allow interactions between variables: each basis function g_d is a product of one-dimensional spline functions, each one depending on one data variable.

Several methods developed in the chemometrics literature, such as *principal component regression* and *partial least squares* (reviewed by Frank and Friedman, 1993) are also closely related to projection pursuit

[12]The backfitting algorithm is an iterative method to fit additive models that fits each term to the residuals given the rest. It is a version of the Gauss-Seidel method of numerical linear algebra.

regression[13]. Such methods include mechanisms to determine the number of projections that gives the smallest regression error on a test set. They usually perform much better than ordinary least squares regression when the sample size N is smaller than the dimensionality D and the predictor variables have a high degree of collinearity (as is the case in many chemical applications).

Projection pursuit density approximation (Friedman et al., 1984) uses a multiplicative composition so that the estimate can be nonnegative and integrate to 1:

$$f(\mathbf{t}) = \prod_{l=1}^{L} h_l(\mathbf{a}_l^T \mathbf{t}).$$

Friedman et al. (1984) and Huber (1985) give algorithms to fit the model. This approach is related to the recently proposed product-of-experts architecture (Hinton, 1999).

4.7 Local dimensionality reduction

In local dimensionality reduction methods, a global model for the data manifold is built as a combination of several simple local models (usually linear). This is justified by several reasons:

- Taylor's theorem: any differentiable function becomes approximately linear in a sufficiently small region around a point.
- The data manifold may actually consist of separate manifolds which may or may not be connected together in one piece; i.e., it may be clustered.
- The intrinsic dimensionality of the data may vary along the manifold. Consider, for example, a manifold with the aspect of fig. 4.6; or the Lorenz attractor (Peitgen et al., 1992), which globally spans three dimensions but locally can be described with only two in most of the space (fig. 4.12).
- The intrinsic dimensionality may not vary, but the orientation may vary as one moves along the manifold. For example, the Lorenz attractor could be embedded in two non-parallel planes.

The key point is that, while the global data manifold may be highly nonlinear and require the whole data space to embed it, individual parts of it may require simple, linear models. In fact, using a complex global model able to represent a large number of manifolds (via a large number of parameters), such as nonlinear autoassociators (section 4.5.2), has several disadvantages:

- The power of the model is wasted in those areas of the space where the manifold is approximately linear.
- A large data set is required to fit a large number of parameters.
- Training becomes difficult because, due to the high flexibility of the model, the error function is likely to have a lot of local minima.

In contrast, in local dimensionality reduction we split the global manifold into simple parts that can be learned easily: fast, with little data and no (or few) local minima. This, then, does not result anymore in a single, global dimensionality reduction mapping (from the data space to a single low-dimensional space) because each local mapping has its own low-dimensional space with its own dimension; and likewise there is no single, global reconstruction mapping. This also happened with mixtures of latent variable models (section 2.9.3). Local dimensionality reduction thus requires:

- Simple dimensionality reduction models as building blocks (typically PCA), usually distributed around the space and each one having a limited reach (hence the locality).
- A way to determine the dimensionality of each component.
- A responsibility assignation rule that, given a point in data space, assigns a weight or responsibility for it to each component. This can be seen as clustering.
- A way to learn both the local models (manifold fitting) and the responsibility assignment (clustering).

[13]Surprisingly, Frank and Friedman (1993) make no mention of projection pursuit regression in their review of chemometrics regression tools.

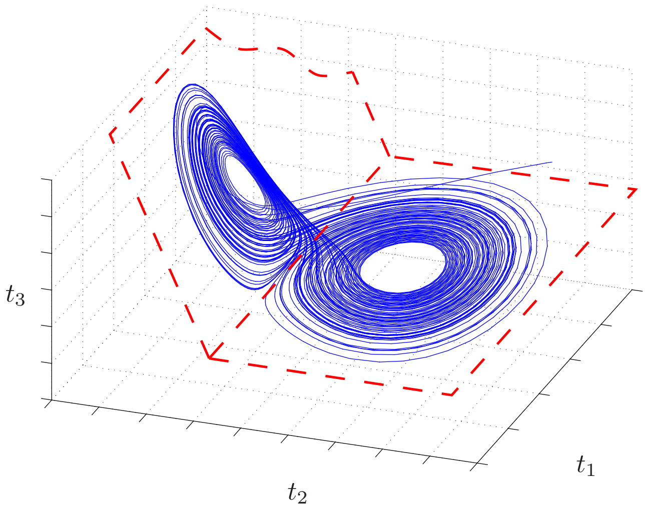


Figure 4.12: The Lorenz attractor can locally be described with two dimensions in most of the three-dimensional space.

The responsibility assignation can be *hard* or *soft*:

- Hard: a single component receives all the responsibility and the rest receive no responsibility at all. It is a winner-take-all approach, usually a form of vector quantisation.
- Soft: the responsibility is distributed among all components as a partition of unity, so that when training, a given data point will result in an update of all components; and when reducing dimensionality, the reduced-dimension representative will be the average of the local reduced-dimension representatives weighted by the respective responsibilities.

It has been observed empirically that soft assignation of responsibilities outperforms hard assignation, which in a probabilistic framework is a consequence of the optimality (in the L_2 sense) of the mean of a distribution to represent the whole distribution—the mode (a form of hard assignation) not being optimal. But the average of two points in different local models need not (and probably will not) belong to the global manifold. We found this situation in section 2.9.1 when having to pick a point of the posterior distribution in latent space as reduced-dimension representative, and we will find it in more detail in chapter 7 when reconstructing arbitrarily missing values. The moral is that data points for which more than one local model are significatively responsible are problematic.

Also, a soft assignment will yield a continuous dimensionality reduction mapping (if the local models have continuous dimensionality reduction mappings as well), which violates the self-consistency condition (4.2) of principal curves, as discussed in section 4.8.

From a probabilistic point of view, the concept of local models and responsibility assignation is naturally expressed as a mixture (of latent variable models) and was covered in section 2.9.3. The training criterion is then log-likelihood rather than reconstruction error, since the probability model attempts to model the noise as well as (and separately from) the underlying manifold. Formulating the local dimensionality reduction problem as a mixture of distributions results in a unified view of the whole model and its probabilistic nature brings a number of well-known advantages, in particular the fact that typically we can derive an EM algorithm that will train all parameters (those of the local models and those of the responsibility assignment) at the same time, with guaranteed convergence and often in a simple way: the E step assigns the responsibilities while the M step fits each local model. Other advantages and also disadvantages are mentioned in the conclusions of the thesis (chapter 11).

In the rest of this section we briefly mention some local dimensionality reduction approaches that do not define a density model in the data space. As one would expect, all of them use PCA-based components, given the attractive properties of PCA mentioned in section 4.5. Therefore, the global manifold is piecewise linear; we can visualise it as a collection of patches glued together. In the region of data space where a given individual model is much more responsible than all the others the dimensionality reduction mapping is then the orthogonal projection on the local principal component subspace. Therefore, the self-consistency

initialise
  $\{\boldsymbol{\mu}_m\}_{m=1}^{M} \leftarrow$ random subset of the data points — Centroids
  $\boldsymbol{\Sigma}_m \leftarrow \mathbf{I}$ for each $m = 1, \dots, M$ — Covariances
repeat
  Assign each data point to the cluster that best reconstructs it — Hard assignation
  for $m = 1, \dots, M$ — For each cluster
    $\boldsymbol{\mu}_m \leftarrow$ mean of data points assigned to cluster m — Update centroids
    $\boldsymbol{\Sigma}_m \leftarrow$ covariance matrix of data points assigned to cluster m — Update covariances
    $\mathbf{U}_m \leftarrow$ first principal components of $\boldsymbol{\Sigma}_m$ — Update local PCA directions
  end
until the fractional change in average reconstruction error is small
return $\{\boldsymbol{\mu}_m, \mathbf{U}_m\}_{m=1}^{M}$

Figure 4.13: Pseudocode of the VQPCA algorithm of Kambhatla and Leen (1997).

condition (4.2) of principal curves is locally satisfied. Globally this need not be true—and it certainly is not in the case of soft assignment.

Bregler and Omohundro (1994, 1995) perform the clustering with a Gaussian mixture starting from the k-means solution (Duda and Hart, 1973) and associate to every centroid thus obtained a local PCA. Although the algorithmic details are not clear in these papers, it seems that a fixed number of nearest neighbours of each centroid is used to compute a local covariance matrix, from which one infers how many principal components to keep as well as the local principal component subspace. Therefore, some data points may participate in the covariance matrices of different, neighbouring centroids. The dimensionality reduction is soft, where the weight of each local model is its posterior probability (with respect to the Gaussian mixture density) given the data point. They use this model for learning a space of lip shapes in a lip-tracking application.

The *optimally adaptive transform coding* of Dony and Haykin (1995) consists of a collection of principal component subspaces with the peculiarity that they are centred at the same point. Therefore, no centroids need to be estimated, although the approach is clearly more limited. Responsibility is computed in a hard fashion both for dimensionality reduction and for training (i.e., for a given data point only one model has its orientation updated). The winner model is the most parallel to the data vector, i.e., the one over which the projection of the data vector is longest. The update of the model is done via an online PCA algorithm, such as those mentioned in section 4.5.1. They use the model for compression and feature extraction of images.

Hinton et al. (1997) applied a mixture of PCAs to model the manifold of handwritten characters. Each PCA module was implemented with a linear autoassociator and both hard and soft versions of training were used, based on k-means and EM, respectively. For the soft version, the training criterion was a log-likelihood function derived from an image model (not from a probabilistic view of PCA). They found a similar performance between the mixture of PCAs and a mixture of factor analysers (which generates a density model for the data).

The *vector quantisation PCA algorithm (VQPCA)* of Kambhatla and Leen (1997) uses a vector quantisation algorithm to produce a hard partition of the data space (a Voronoi tessellation, defined in section 4.10.2). In each of the Voronoi cells a separate PCA is fitted, whose and mean covariance matrix are calculated from the data points in that cell. Rather than assigning data points to centroids using the Euclidean distance, they use a distortion function that takes into account not only the distance to the centroid but also the projection on its local subspace. The iterative algorithm is shown in fig. 4.13. They apply their model to dimensionality reduction of speech and image data and show it to outperform nonlinear autoassociators in both speed and reconstruction error.

It is interesting that the partitions formed when clustering according to reconstruction error alone can be nonlocal, as simulations by Kambhatla and Leen (1997) and Tipping and Bishop (1999a) showed.

The literature contains many similar local dimensionality reduction models, so the ones we have presented should be considered only as representative. References to such other models can be found in several of the papers mentioned (Hinton et al., 1997; Kambhatla and Leen, 1997; Tipping and Bishop, 1999a). The same kind of ideas has been used not only for dimensionality reduction but also for regression and classification, as in the hierarchical mixtures of experts of Jordan and Jacobs (1994).

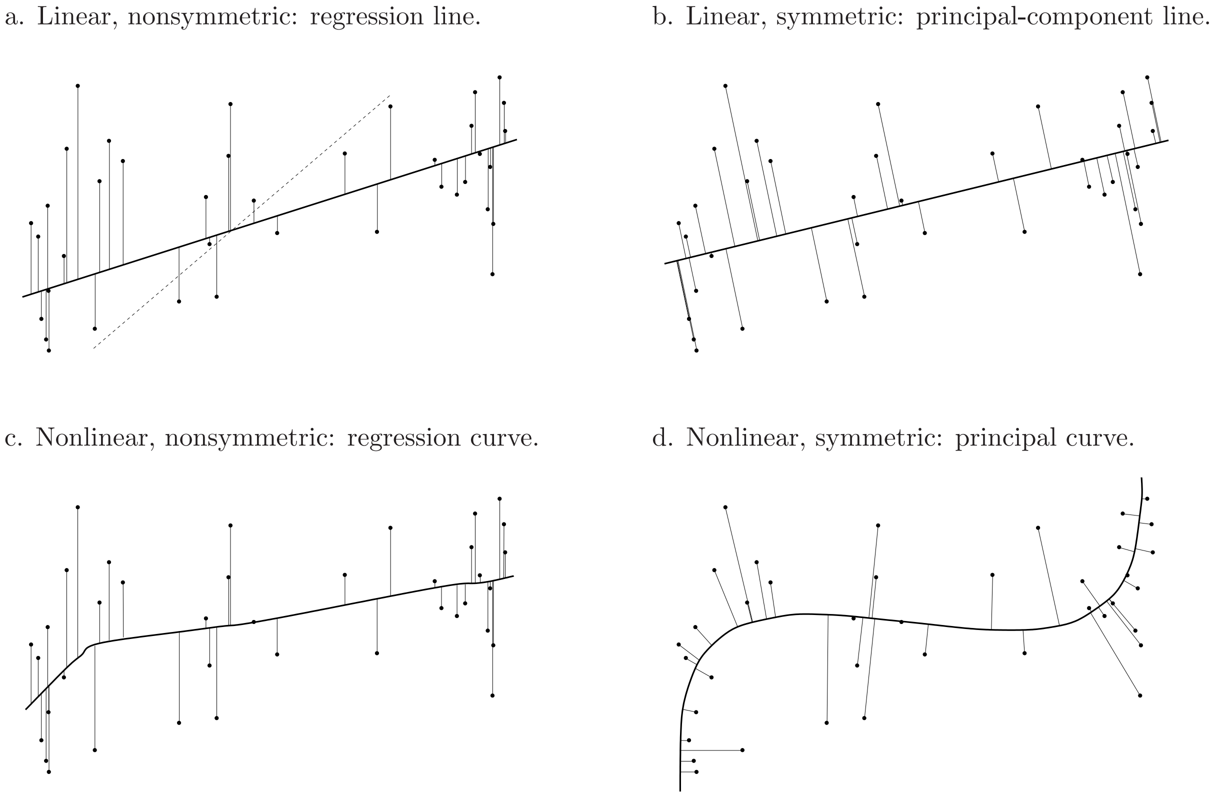


Figure 4.14: Principal curves as generalised (nonlinear, symmetric) regression. (a) The linear regression line minimises the sum of squared deviations in the response variable (or in the independent one, for the dashed line). (b) The principal-component line minimises the sum of squared deviations in all the variables. (c) The smooth regression curve minimises the sum of squared deviations in the response variable, subject to smoothness constraints. (d) The principal curve minimises the sum of squared deviations in all the variables, subject to smoothness constraints. From Hastie and Stuetzle (1989).

4.8 Principal curves

Principal curves are intuitively defined as smooth curves that pass through the middle of a D-dimensional data set, providing a nonlinear summary of it (Hastie and Stuetzle, 1989). They are estimated in a nonparametric way, i.e., their shape is suggested by the data. Their motivation is a generalisation of regression (see fig 4.14):

- Linear regression minimises the sum of squared deviations in the response variable $y = ax + b$ (i.e., in the vertical direction in an X-Y graph). Thus changing the roles produces a different regression line (dotted line).
- The first principal component is a regression line symmetrical with respect to all the variables, minimising orthogonal deviation to that line.
- Nonlinear regression produces a curve that minimises the sum of squared deviations in the response variable (the vertical deviations) subject to some form of smoothness constraint. Again, changing the roles produces a different regression curve.
- Principal curves are a natural generalisation for nonlinear, symmetric regression: they attempt to minimise the sum of squared deviations in all the variables (i.e., the orthogonal or shortest distance from the curve to the points) subject to smoothness constraints.

Therefore, if we consider that $\mathbf{f} : x \in \mathcal{X} \subset \mathbb{R} \longrightarrow \mathbf{t} = \mathbf{f}(x) \in \mathbb{R}^D$ is a smooth curve in $\mathbb{R}^D$ parameterised by its arc length x, principal curves define, in the sense of section 4.2:

- A dimensionality reduction mapping F: given $\mathbf{t} \in \mathbb{R}^D$, $F(\mathbf{t})$ is the arc length of the nearest point[14] in the curve to $\mathbf{t}$ in the Euclidean distance, i.e., the arc length of its orthogonal projection on the curve.
- A reconstruction mapping $\mathbf{f}$ given by the principal curve parametric equation, $\mathbf{f}(x)$.

The advantage of using the arc length parametrisation is that the distance *along the curve* between points $\mathbf{f}(x_1)$ and $\mathbf{f}(x_2)$ is simply $|x_1 - x_2|$ (i.e., the geodetic distance, which in general is larger than the Euclidean, or straight-line, distance between $\mathbf{f}(x_1)$ and $\mathbf{f}(x_2)$).

We say that a curve is **self-consistent** with respect to a distribution if the average of all data points that (orthogonally) project onto a given point on the curve coincides with the point on the curve:

$$\forall x \in \mathcal{X}: \ \mathrm{E}\left\{\mathbf{t}|F(\mathbf{t}) = x\right\} = \mathbf{f}(x). \tag{4.2}$$

We can then say that principal curves pass through the middle of the data in a smooth way and are self-consistent for that distribution. Using this property, Hastie and Stuetzle (1989) prove that, for a given distribution, principal curves are the stationary points of the average of the Euclidean distance of a data point to its projection on the curve for perturbations of bounded norm and bounded derivative. This result, verified by principal components if only straight lines are considered, confirms the aforementioned role of principal curves as nonlinear, symmetric regression.

However, this definition of principal curves poses several questions: for what kinds of distributions do principal curves exist, how many different principal curves exist for a given distribution and what are their properties? These questions have only been answered for some particular cases:

- For ellipsoidal distributions (e.g. the normal distribution) the principal components are principal curves. In higher dimensions, the subspaces spanned by any subset of principal components are principal manifolds.
- For spherically symmetric distributions any straight line through the mean is a principal curve.
- For 2D spherically symmetric distributions a circle with centre at the mean and radius $\mathrm{E}\left\{\|\mathbf{t}\|\right\}$ is a principal curve too and has smaller expected squared distance from the distribution than the straight lines.
- If a straight line is self-consistent, then it is a principal component. In other words, linear principal curves are principal components.

For a model of the form $\mathbf{t} \stackrel{\text{def}}{=} \mathbf{f}(x) + \mathbf{e}$, with $\mathbf{f}$ smooth and $\mathrm{E}\left\{\mathbf{e}\right\} = 0$ (such as all the latent variable models described in chapter 2), $\mathbf{f}$ is not a principal curve in general, as fig. 4.15 shows. This means that the principal curve is biased for the functional model, although the bias seems to be small and to decrease to 0 as the variance of the errors gets small relative to the radius of curvature of $\mathbf{f}$. This bias is a consequence of the self-consistency condition (4.2); in fact, relaxing it to the condition (2.5) and considering $\mathbf{t}$ and x as random variables results in a continuous latent variable model, which is unbiased by definition (eq. (2.5) in section 2.3.1). Banfield and Raftery (1992) give a robust estimation method for closed principal curves that reduces both bias and variance.

The definition of principal curves can be naturally extended to several dimensions, in which case we could call them *principal manifolds* (although the arc length, or unit-speed, parameterisation is not naturally defined for more than one dimension). However, once again the curse of the dimensionality makes smoothing in several dimensions hard unless data are abundant. Note also that principal curves depend critically on the scaling of the features, as all projection techniques do.

The definition of the projection on the principal manifold $\mathcal{M}$ as orthogonal projection leads necessarily to discontinuities in the dimensionality reduction mapping if the projection sets of two points in $\mathcal{M}$ intersect:

$$\exists \mathbf{t} \in \mathbf{F}^{-1}(\mathbf{x}_1) \cap \mathbf{F}^{-1}(\mathbf{x}_2) \Rightarrow \text{ is } \mathbf{F}(\mathbf{t}) = \mathbf{x}_1 \text{ or } \mathbf{F}(\mathbf{t}) = \mathbf{x}_2?$$

An example is the centre of the circle in figure 4.15, since for any two points in the circle, $\mathbf{F}^{-1}(x_1) \cap \mathbf{F}^{-1}(x_2) = \{C\}$. Choosing one of the possibilities (as Hastie and Stuetzle (1989) do: the one with smallest arc length) leads to a discontinuity at such point $\mathbf{t}$. This will happen at some data points for any manifold except when no two projection sets intersect, i.e., they are all parallel, which implies that the principal manifold is a hyperplane—spanned by principal components.

Hastie and Stuetzle (1989) give a construction algorithm for principal curves, shown in fig. 4.16. Although by definition principal curves are fixed points of this algorithm, it has not been proven to converge in general. Observe that:

[14] For definiteness, in the exceptional case where there are several nearest points, we take the one with largest arc length.

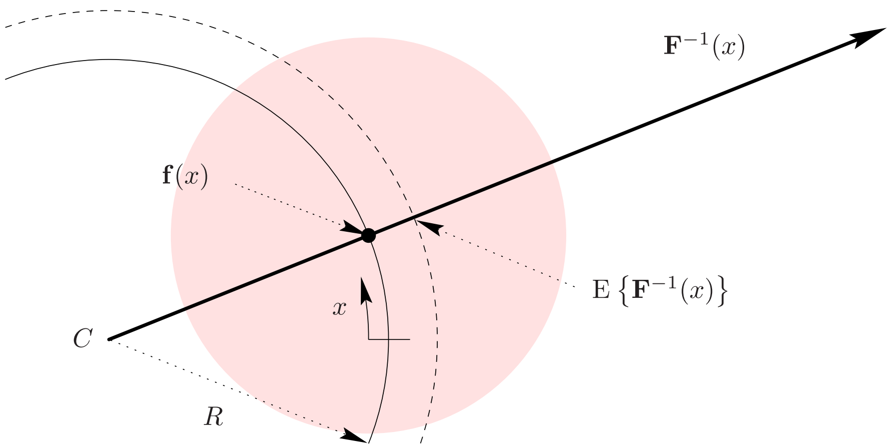


Figure 4.15: Bias in principal curves. In this example, the data distribution is $\mathbf{t} \stackrel{\text{def}}{=} \mathbf{f}(x) + \mathbf{e}$ where $\mathbf{f}$ is the circle of radius R (solid line), parameterised by arc length x, and $\mathbf{e} \sim \mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I})$. The radius of the shaded circle is equal to σ. For any point $\mathbf{f}(x)$ on the curve, $\mathbf{F}^{-1}(x)$ (the set of points that are closest to $\mathbf{f}(x)$, i.e., whose reduced-dimension representative is x) is, by symmetry, the half-line starting in the circle centre, passing through $\mathbf{f}(x)$ and going towards infinity (marked thick). However, $\mathrm{E}\left\{\mathbf{F}^{-1}(x)\right\} \neq \mathbf{f}(x)$ and therefore the generating curve $\mathbf{f}$ is not self-consistent in the sense of condition (4.2). The radius of the self-consistent circle (dotted line) is larger than R and can be found to be equal to $R + \left(1 + \operatorname{erf}\left(\frac{R}{\sigma\sqrt{2}}\right)\right)^{-1} \sqrt{\frac{\pi}{2}}\sigma e^{-\frac{1}{2}\left(\frac{R}{\sigma}\right)^2}$.

- The algorithm converges to the first principal component if the conditional expectation operation is replaced by fitting a least-squares straight line. Principal curves are then local minima of the distance function (sum of squared distances).
- For probability distributions, both operations—projection and average or conditional expectation—reduce the expected distance from the points to the curve; for discrete data sets this is unknown.
- The algorithm does not necessarily produce smooth curves, in particular at the curve endpoints.
- The smoothness of the resulting principal curve is sensitive to the size of the neighbourhood over which averaging takes place.

By relaxing the self-consistency condition (4.2), Tibshirani (1992) redefines principal curves based on a continuous mixture model and estimates it via an EM algorithm. His view is equivalent to nonparametric estimation of a continuous latent variable model, as discussed in sections 2.5.2 and 2.3.1.1.

initialise
  $\tau \leftarrow 0$ — Iteration counter
  $\mathbf{f}^{(0)} \leftarrow$ first principal components — Principal curve
repeat
  Project the distribution onto $\mathbf{f}^{(\tau)}$, i.e., compute ${F^{(\tau)}}^{-1}(x)$ for each $x \in \mathcal{X}$. — Projection step
  $\mathbf{f}^{(\tau+1)}(x) \leftarrow \mathrm{E}\left\{\mathbf{t} | F^{(\tau)}(\mathbf{t}) = x\right\} = \mathrm{E}\left\{{F^{(\tau)}}^{-1}(x)\right\}$. — Averaging step
  Reparameterise $\mathbf{f}^{(\tau+1)}$ in terms of arc length.
until $\mathbf{f}^{(\tau+1)} \equiv \mathbf{f}^{(\tau)}$ — Self-consistence condition
return $\mathbf{f}^{(\tau)}$

Figure 4.16: Pseudocode of the construction algorithm for principal curves from Hastie and Stuetzle (1989). Since principal curves are estimated nonparametrically, the averaging step requires a local average of the data set.

Hastie and Stuetzle (1989) give an alternative definition of principal curves, related to spline smoothing, as curves that minimise the distance (conveniently defined) of the data points to the curve subject to a smoothness constraint. Thus, it becomes a regression problem of the data $\{\mathbf{t}_n\}_{n=1}^N$ as a function of the unknown variables $x_1, \dots, x_L$ (in L dimensions). This is then very similar to latent variable models but without a joint probability model for $\{\mathbf{t}, \mathbf{x}\}$. This second definition of principal curves has proven more amenable to developments. LeBlanc and Tibshirani (1994) implement principal curves parametrically in this sense using a MARS model (mentioned in section 4.6.2) extensible to higher dimensions. Kégl et al. (2000) prove that, by restricting the definition to curves of a given length at most, such principal curves always exist if the distribution has finite second moments and give a learning algorithm for them based on polygonal lines (restricted to the one-dimensional case).

The natural interpretation of principal curves as nonlinear, symmetric regression seems very attractive in terms of least-squares dimensionality reduction. However, before they become a practical framework for dimensionality reduction, a number of theoretical and practical questions must be answered and estimation algorithms that work in arbitrary dimensions must be developed.

4.9 Methods based on vector quantisation

Vector quantisation (Gray, 1984; Gray and Neuhoff, 1998) consists of summarising a data distribution in data space $\mathbb{R}^D$ via a discrete collection of **reference** or **codebook vectors** $\{\boldsymbol{\mu}_m\}_{m=1}^M \subset \mathbb{R}^D$. Once the codebook has been constructed (the reference vectors trained), for which various algorithms are available, a given data point $\mathbf{t} \in \mathbb{R}^D$ becomes *quantised* to the closest reference vector to it according to some convenient distance, usually the Euclidean. As such, vector quantisation is useful for classification but not for dimensionality reduction. A limited form of dimensionality reduction becomes possible if one imposes a topographic structure on the reference vectors. This can be done via a learning rule or objective function that, at the same time that it tries to fit the reference vectors to the data, discourages configurations that violate the topographic arrangement. Several such methods exist, of which we mention two: Kohonen's self-organising maps and the elastic net.

None of these methods properly defines a manifold that embeds the reference vectors[15]: the L-dimensional manifold in data space is defined indirectly by the location of the reference vectors. Therefore, no continuous dimensionality reduction and reconstruction mappings in the sense of section 4.2 are derived: given a data point, all one can do is to assign it to the closest reference vector and use its associated knot in the topographic arrangement as reduced-dimension representative, or interpolate in some way. This latter option consists of defining dimensionality reduction and reconstruction mappings by assuming the low-dimensional space to be embedded in an Euclidean space (e.g. by placing the lattice nodes of a self-organising map in the $\mathcal{X} = [0,1]^D$ hypercube in an equispaced way) and then fitting a regularised universal mapping approximator to the functions $\mathcal{X} \xrightarrow{\mathbf{f}} \mathcal{T}$ (reconstruction) and/or $\mathcal{T} \xrightarrow{\mathbf{F}} \mathcal{X}$ (dimensionality reduction), learned in a supervised way from the reference vectors and lattice points.

4.9.1 Kohonen's self-organising maps

Let $\{\mathbf{t}_n\}_{n=1}^N$ be a sample in the data space $\mathcal{T} = \mathbb{R}^D$. Kohonen's self-organising maps (SOMs) (Kohonen, 1995) learn, in an unsupervised way, a mapping between a 2D lattice[16] $\mathcal{X}$ and the data space that preserves the two-dimensional topology of the lattice and adapts to the manifold spanned by the sample. One can visualise the learning process as a flat sheet that twists around itself in D dimensions to resemble as much as possible the distribution of the data vectors.

Each of the reference vectors $\{\boldsymbol{\mu}_m\}_{m=1}^M$ in data space is associated with a node v_m in the 2D lattice $\mathcal{X}$. Assume we have defined two distances (typically Euclidean) $d_\mathcal{T}$ and $d_\mathcal{X}$ in the data space and in the lattice, respectively. The topology of the lattice is determined by the **neighbourhood function** $h(v_m, v_{m'})$. This is a symmetric function with values in $[0,1]$ that behaves as an inverse distance in the lattice: $h(v_m, v_m) = 1$ for any node v_m and, given a node v_m, for any other node $v_{m'}$ $h(v_m, v_{m'})$ is smaller the farther apart node $v_{m'}$ is from node v_m in the lattice. The neighbourhood of node v_m is composed of those nodes for which $h(v_m, v_{m'})$ is not negligibly small. In practice, usually $h(v_m, v_{m'}) = \exp(-d_\mathcal{X}(v_m, v_{m'})/2\sigma^2)$, where σ quantifies the range of the neighbourhood.

[15] One can always fit an interpolating manifold of some kind to the reference vectors once these have been trained, but this is then the original problem applied to the reference vectors rather than to the original sample.

[16] In the typical case, but the idea is valid for L-dimensional topological arrangements.

The reference vectors are initially distributed at random (or perhaps are a random set of the data vectors, or are scattered along the first principal components). A competitive learning rule, **Kohonen learning**, is applied iteratively over all data vectors until convergence is achieved. Given a data vector $\mathbf{t}_n$, let $\boldsymbol{\mu}_{m^*}$ be the reference vector closest to $\mathbf{t}_n$ in data space:

$$m^* = \arg \min_{m=1,\dots,M} d_{\mathcal{T}}(\boldsymbol{\mu}_m, \mathbf{t}_n).$$

Learning occurs as follows (where τ is the iteration index and $\alpha^{(\tau)} \in [0,1]$ is the learning rate):

$$\boldsymbol{\mu}_m^{(\tau+1)} = \boldsymbol{\mu}_m^{(\tau)} + \alpha^{(\tau)} h^{(\tau)}(v_{m^*}, v_m)(\mathbf{t}_n - \boldsymbol{\mu}_m^{(\tau)}) = (1-\rho)\boldsymbol{\mu}_m^{(\tau)} + \rho \mathbf{t}_n$$

i.e., reference vector $\boldsymbol{\mu}_m$ is drawn a distance $\rho = \alpha^{(\tau)} h^{(\tau)}(v_{m^*}, v_m)$ toward data vector $\mathbf{t}_n$. The update affects only vectors whose associated nodes lie in the neighbourhood of the winner v_{m^*} and its intensity decreases with the iteration index τ because both $\alpha^{(\tau)}$ and the range of $h^{(\tau)}$ must decrease with τ for convergence reasons.

Intuitively it seems that the reference vectors $\boldsymbol{\mu}_m$ will become abundant in regions of $\mathbb{R}^D$ where the $\mathbf{t}_n$ are common and sparse where the $\mathbf{t}_n$ are uncommon, thus following the distribution of the data vectors. However, they do not approximate the data density even with infinite data; in fact, they underestimate the density where it is high and overestimate it where it is low (Kohonen, 1995, p. 110–111).

A batch training algorithm for SOMs exists that can be written as (see Mulier and Cherkassky, 1995 and references therein):

$$\boldsymbol{\mu}_m^{(\tau+1)} = \frac{\sum_{m'=1}^M h^{(\tau)}(v_{m'} - v_m) N_{m'}^{(\tau)} \boldsymbol{\mu}_{m'}^{(\tau)}}{\sum_{m'=1}^M h^{(\tau)}(v_{m'} - v_m) N_{m'}^{(\tau)}}$$

where $N_{m'}^{(\tau)}$ is the number of data points $\{\mathbf{t}_n\}_{n=1}^N$ that lie in the Voronoi cell of reference vector $\boldsymbol{\mu}_{m'}^{(\tau)}$ (i.e., the number of data points whose closest reference vector is $\boldsymbol{\mu}_{m'}^{(\tau)}$) and each iteration contains the updates due to all the data points. This equation has the form of a kernel regression (Nadaraya-Watson estimator; Silverman, 1986) where the response variables are the data variables $\mathbf{t}$, the predictor variables are the nodes v (which can be assumed to lie on an Euclidean space), the (unnormalised) kernels are $h^{(\tau)}(v_{m'} - v) N_{m'}^{(\tau)}$ and the "training set" is $\{(v_m, \boldsymbol{\mu}_m^{(\tau)})\}_{m=1}^M$, with both kernels and training set depending on the iteration index τ. From this point of view, at each iteration the SOM defines a continuous mapping $\mathbf{t} = \mathbf{f}^{(\tau)}(v)$ from latent space onto data space:

$$\mathbf{f}^{(\tau)}(v) = \frac{\sum_{m'=1}^M h^{(\tau)}(v_{m'} - v) N_{m'}^{(\tau)} \boldsymbol{\mu}_{m'}^{(\tau)}}{\sum_{m'=1}^M h^{(\tau)}(v_{m'} - v) N_{m'}^{(\tau)}}$$

as Mulier and Cherkassky (1995) argue. However, for $\tau \to \infty$, $h^{(\tau)}(v_{m'} - v) \to \delta(v_{m'} - v)$ and so the mapping $\mathbf{f}^{(\infty)}$ is only defined at the nodes $\{v_m\}_{m=1}^M$. Thus, a trained SOM does not define a continuous mapping from latent to data space.

In summary, Kohonen learning creates an L-dimensional arrangement such that:

- The number density of reference vectors in data space is approximately proportional to a power (smaller than one) of the data probability density.
- The mapping from the L-dimensional arrangement into data space (ideally) preserves the topographic ordering: nearby points in the data space are mapped onto nearby points in the lattice.

Kohonen's SOMs have proven successful in many practical applications, particularly in visualisation. However, their heuristic nature results in several shortcomings:

- Many parameters must be tuned by trial and error with little guarantee of success: the shape of the lattice (rectangular, etc.), the number of codebook vectors or the schedules for the evolution of the neighbourhood function and learning rate. In general, no schedules that guarantee convergence and no proofs of convergence exist. Thus, training is unreliable. However, the shape of the neighbourhood function is largely irrelevant (as happens in kernel estimation).
- No cost function to optimise can be defined, although SOMs have been shown to be approximately related to probabilistic cost functions (Luttrell, 1994; Utsugi, 1997a,b).
- Neither a probability distribution function nor a manifold function is obtained for the data.

Table 4.4 compares Kohonen's SOMs with GTM (section 2.6.5), which—being defined as a latent variable model—enjoys much more attractive theoretical properties. The table applies to most of the variations of SOMs that have been proposed (see Kohonen, 1995 for a review).

	SOM	GTM
Internal representation of manifold	Nodes $\{v_m\}_{m=1}^M$ in L-dimensional array, held together by neighbourhood function h	Point grid $\{\mathbf{x}_k\}_{k=1}^K$ in L-dimensional latent space that keeps its topology through smooth mapping $\mathbf{f}$
Definition of manifold in data space	Indirectly by locations of reference vectors	Continuously by mapping $\mathbf{f}$
Objective function	No	Yes: log-likelihood
Self-organisation	Difficult to quantify	Smooth mapping $\mathbf{f}$ preserves topology
Convergence	Not guaranteed	Yes, by the EM algorithm
Smoothness of manifold	Depends on $\alpha^{(\tau)}$ and $h^{(\tau)}$	Depends on basis functions parameters and prior distribution $p(\mathbf{x})$
Generative model	No; hence no density function	Yes
Additional parameters to select	$\alpha^{(\tau)}$, $h^{(\tau)}$ arbitrarily	None
Speed of training	Comparable according to Bishop et al. (1998b)	
Magnification factors	Approximated by the difference between reference vectors	Exactly computable anywhere

Table 4.4: Comparison between GTM and Kohonen's SOM.

4.9.2 The elastic net

The travelling salesman problem (Lawler et al., 1985) is a combinatorial optimisation task that requires to find the shortest circular tour through a set of N cities that passes through each city exactly once. It is an NP-complete problem: solving it is $\mathcal{O}(e^N)$. The elastic net algorithm (Durbin and Willshaw, 1987) generates good solutions in much less time by stretching a circular tour formed by M knots elastically linked in succession to fit the cities (the topographic arrangement of the elastic net can be extended to any dimension); that is, it is an algorithm for continuous, rather than discrete, optimisation. Let $\{\mathbf{t}_n\}_{n=1}^N$ represent the positions of the N cities in $\mathbb{R}^D$ and $\{\boldsymbol{\mu}_m\}_{m=1}^M$ the elastic net knots. The algorithm consists of the updating rule

$$\Delta\boldsymbol{\mu}_m \stackrel{\text{def}}{=} \alpha \sum_{n=1}^N w_{nm}(\mathbf{t}_n - \boldsymbol{\mu}_m) + \frac{1}{2}\beta K(\boldsymbol{\mu}_{m+1} - 2\boldsymbol{\mu}_m + \boldsymbol{\mu}_{m-1}) \qquad w_{nm} \stackrel{\text{def}}{=} \frac{e^{-\frac{1}{2}\left(\frac{\|\mathbf{t}_n-\boldsymbol{\mu}_m\|}{K}\right)^2}}{\sum_{m'=1}^M e^{-\frac{1}{2}\left(\frac{\|\mathbf{t}_n-\boldsymbol{\mu}_{m'}\|}{K}\right)^2}}$$

where α and β are constants and K is the scale parameter (which is typically annealed, i.e., slowly decreased to zero). The α term pulls the path toward the cities while the β term pulls neighbouring path points toward each other. The rule can be seen as an energy minimisation (or wiring length minimisation in the context of cortical maps; Durbin and Mitchison, 1990):

$$E(\{\boldsymbol{\mu}_m\}_{m=1}^M, K) \stackrel{\text{def}}{=} -\alpha K \sum_{n=1}^N \ln\left(\sum_{m=1}^M e^{-\frac{1}{2}\left(\frac{\|\mathbf{t}_n-\boldsymbol{\mu}_m\|}{K}\right)^2}\right) + \frac{\beta}{2}\sum_{m=1}^M \left\|\boldsymbol{\mu}_m - \boldsymbol{\mu}_{m+1}\right\|^2 \qquad \Delta\boldsymbol{\mu}_m = -K\frac{\partial E}{\partial \boldsymbol{\mu}_m} \quad (4.3)$$

or (by exponentiation, analogously to the use of the Boltzmann-Gibbs distribution in statistical mechanics) as a maximum a posteriori estimation of the following probability model (Durbin et al., 1989):

- Prior probability of a tour that favours short tours: $p(\{\boldsymbol{\mu}_m\}_{m=1}^M) \stackrel{\text{def}}{\propto} \prod_{m=1}^M e^{-\frac{\beta}{2\alpha K}\left\|\boldsymbol{\mu}_m-\boldsymbol{\mu}_{m+1}\right\|^2}$. That is, a correlated Gaussian. Topologies other than the nearest-neighbour one can be used (Dayan, 1993; Utsugi, 1997a).
- Probability of the city collection given a tour: $p(\{\mathbf{t}_n\}_{n=1}^N|\{\boldsymbol{\mu}_m\}_{m=1}^M) \stackrel{\text{def}}{=} \prod_{n=1}^N p(\mathbf{t}_n|\{\boldsymbol{\mu}_m\}_{m=1}^M)$ with $p(\mathbf{t}_n|\{\boldsymbol{\mu}_m\}_{m=1}^M) \stackrel{\text{def}}{=} \frac{1}{M}\sum_{m=1}^M (2\pi K^2)^{-D/2} e^{-\frac{1}{2}\left(\frac{\|\mathbf{t}_n-\boldsymbol{\mu}_m\|}{K}\right)^2}$. That is, a product of Gaussian mixtures with isoprobable components centred at the tour knots.

The algorithm minimises E at large K, where there is a single minimum, and then tracks the minimum down to small K, where every city location $\mathbf{t}_n$ is matched by some tour knot $\boldsymbol{\mu}_m$, as Durbin et al. (1989) prove.

4.10 Distance-preserving methods

In this section we discuss an approach to dimensionality reduction based on representing the data manifold by a low-dimensional Euclidean space where the distances between observed points are preserved. This is a purely geometric point of view, as opposed to the generative one of latent variable models or the minimum squared reconstruction error (orthogonal projection) of several of the methods of this chapter. That is, given a noisy sample $\{\mathbf{t}_n\}_{n=1}^N$ from a data manifold $\mathcal{M} \subset \mathcal{T} \subset \mathbb{R}^D$ of dimensionality $L \leq D$, the aim of distance-preserving methods is to find a collection of points $\{\mathbf{x}_n\}_{n=1}^N$ in a metric space $\mathcal{X} \subset \mathbb{R}^L$ associated with the high-dimensional sample such that the distance (to be defined) between two points $\mathbf{t}_n$ and $\mathbf{t}_m$ is approximately the same as the distance (usually Euclidean) between their associated low-dimensional points $\mathbf{x}_n$ and $\mathbf{x}_m$.

The low-dimensional Euclidean space $\mathcal{X}$ is not unique, since any rigid motion (translation, orthogonal rotation or reflection) of the $\{\mathbf{x}_n\}_{n=1}^N$ points preserves the distances between them; and the topology of $\mathcal{X}$ should match that of the data manifold $\mathcal{M}$. For example, if the latter is a closed surface, then using a rectangular low-dimensional space will lead to discontinuities.

Regarding the construction of a dimensionality reduction mapping $\mathcal{T} \xrightarrow{\mathbf{F}} \mathcal{X}$ and a reconstruction mapping $\mathcal{X} \xrightarrow{\mathbf{f}} \mathcal{T}$, there are two possibilities:

- Implicit definition: analogously to the case of methods based on vector quantisation, the data manifold is implicitly defined through the collection of points $\{\mathbf{t}_n\}_{n=1}^N$ in data space, and together with their associated points $\{\mathbf{x}_n\}_{n=1}^N$ in the low-dimensional map, a dimensionality reduction mapping and a reconstruction mapping are also implicitly defined. Such implicit definitions can be made explicit by fitting to them a regularised parametric model with supervised learning, e.g. an MLP with weight decay. Regularising the mapping approximator is very important since the whole process implicitly assumes that there is no noise—clearly a dangerous assumption since the interpoint distances (either through a straight line or through a geodesic) are heavily influenced by noise. Traditional multidimensional scaling, the Sammon mapping and methods based on geodetic distances generally belong to the implicit-definition type.

- Explicit definition: rather than directly learning the map points $\{\mathbf{x}_n\}_{n=1}^N$, define a parametric dimensionality reduction mapping $\mathbf{x} \stackrel{\text{def}}{=} \mathbf{F}(\mathbf{t}; \boldsymbol{\Theta})$ and learn instead the parameters $\boldsymbol{\Theta}$ that minimise the stress function (4.4) below. This allows to map new data vectors not in $\{\mathbf{t}_n\}_{n=1}^N$ and has the additional advantage of having fewer parameters compared to directly learning $\{\mathbf{x}_n\}_{n=1}^N$ (which requires LN parameters). This approach has been proposed by Webb (1995) (see also Webb, 1999), who used a radial basis function network for $\mathbf{F}$ with multidimensional scaling and iterative majorisation[17] as stress minimisation algorithm; and by Mao and Jain (1995), who used a multilayer perceptron for $\mathbf{F}$ with Sammon's mapping and online gradient descent (resulting in a variation of backpropagation) with an optional momentum term as stress minimisation algorithm.

4.10.1 Multidimensional scaling (MDS)

Multidimensional scaling (MDS) (Cox and Cox, 1994; Kruskal and Wish, 1978; Mardia et al., 1979; Webb, 1999) is the traditional statistical method for uncovering structure in a data set by plotting a low-dimensional map that preserves the proximities in the (high-dimensional) data set. That is, MDS plots similar objects close together. It has been applied to psychology, sociology, anthropology, economy, educational research, etc. MDS is actually a set of mathematical techniques differing in various theoretical and algorithmic aspects.

Suppose we have a set of N objects and that a measure of the similarity of these objects with each other is known. This measure, called *proximity*, is a number that indicates how similar two objects are or are perceived to be. It can be obtained in different ways, e.g. by asking people to judge the psychological closeness of the stimulus objects. What MDS does is to draw a spatial representation or map in which each object is represented by a point and the distances between points resemble as faithfully as possible the original

[17] Iterative majorisation is an minimisation algorithm where at each iteration one defines a majorisation function (i.e., an upper bound of the objective function) that has a single, easily computable minimum—typically a quadratic function. Under certain conditions, the sequence of minima converges to a minimum of the objective function. It does not require computing gradients of the latter but is very slow.

	A	B	C	···	0
A	82	04	08		03
B	06	84	37		04
C	04	38	87		12
···				···	
0	09	03	11		94

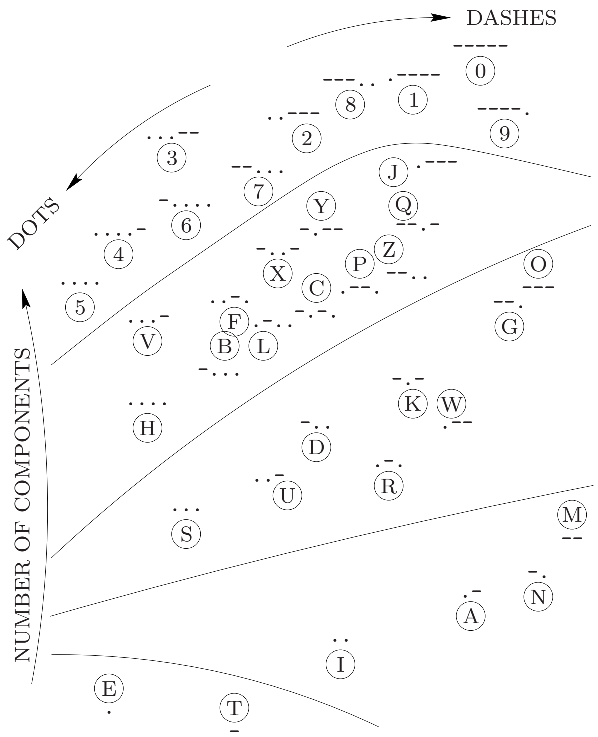


Figure 4.17: *Left*: data from Rothkopf (1957) on similarities among Morse code symbols. *Right*: 2D map obtained for the Morse code similarities by Shepard (1963) with multidimensional scaling.

similarity information; i.e., the larger the dissimilarity between two objects, the farther apart they should be in the spatial representation. This geometrical configuration of points reflects the hidden structure of the data and often makes it much easier to understand.

Let us consider a classical example. Confusions among 36 auditory Morse code signals were collected by Rothkopf (1957). Our "high-dimensional" objects are here the signals, each of which consists of a sequence of dots and dashes, such as `-.-` for `K` and `..---` for `2`. Subjects who did not know Morse code listened to a pair of signals (produced at a fixed rapid rate by machine and separated by a quiet period of 1.4 seconds), and were required to state whether the two signals they heard were the same or different. Each number in the table of fig. 4.17 is the percentage of roughly 150 observers who responded "same" to the row signal followed by the column signal. This matrix is roughly symmetric, with large diagonal entries and small off-diagonal entries, as expected, and contains the proximities data. Figure 4.17 (right) shows the result of applying MDS to those proximities (from Shepard, 1963) using a two-dimensional map. The 36 circles represent the points found and are labelled with the corresponding Morse code. In this case, MDS clearly shows that the data are governed by a sort of length parameter, the number of components of the signal, as well as by the individual numbers of dots and dashes.

All is needed for MDS is the proximities matrix, not the actual locations of the objects in a hypothetic high-dimensional space, which may be nonsensical, as in the Morse code example. The proximities need not be distances in the mathematical sense; in particular, they need not satisfy the symmetry property or the triangle inequality. However, for the dimensionality reduction problem these proximities will be determined from a set of points $\{\mathbf{t}_n\}_{n=1}^N$ in a high-dimensional space $\mathcal{T} \subset \mathbb{R}^D$. Typical definitions of proximity for this case are the Euclidean distance (L_2-norm), the Mahalanobis distance with respect to some (semi)positive definite matrix (e.g. the inverse data covariance matrix, in which case it is equivalent to applying MDS to the presphered data), the Manhattan distance (L_1-norm), etc.

Formally, assume the input data are the pairwise proximities[18] $\{\delta_{nm}\}_{n,m=1}^N$. If they come from a data set $\{\mathbf{t}_n\}_{n=1}^N$, then $\delta_{nm} \stackrel{\text{def}}{=} d_{\mathcal{T}}(\mathbf{t}_n, \mathbf{t}_m)$. For an L-dimensional map, the output data will be a set of points $\{\mathbf{x}_n\}_{n=1}^N \subset \mathbb{R}^L$, referred to some unimportant coordinate system, such that the distances $d_{nm} \stackrel{\text{def}}{=} d_{\mathcal{X}}(\mathbf{x}_n, \mathbf{x}_m)$ $\forall n, m = 1, \dots, N$ (typically Euclidean) are as close as possible to a function f of the corresponding proximities,

[18] This is called two-way MDS. In three-way MDS we have several sets of proximities (e.g. in different times or by different subjects).

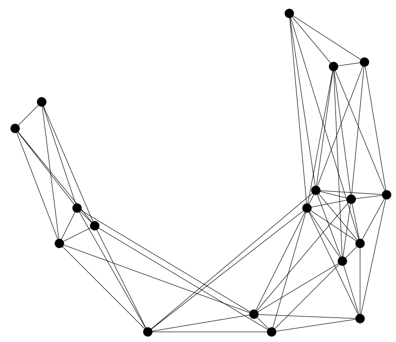


Figure 4.18: The horseshoe phenomenon. The graph shows a two-dimensional Euclidean map obtained by MDS where points whose associated objects have a similarity above a certain threshold are linked by a line.

$f(\delta_{nm})$. The computational procedure is as follows. Define an objective function, traditionally called *stress*, such as:

$$\text{stress}(\{\mathbf{x}_n\}_{n=1}^N, f) \stackrel{\text{def}}{=} \sqrt{\frac{\sum_{n,m=1}^N (f(\delta_{nm}) - d_{nm})^2}{\text{scale factor}}} \tag{4.4}$$

where the scale factor will typically be $\sum_{n,m=1}^N d_{nm}^2$. If the map is defined via a parametric function $\mathbf{F}$, $\mathbf{x} \stackrel{\text{def}}{=} \mathbf{F}(\mathbf{t};\mathbf{\Theta})$, then $d_{nm} \stackrel{\text{def}}{=} d_{\mathcal{X}}(\mathbf{F}(\mathbf{t}_n;\mathbf{\Theta}), \mathbf{F}(\mathbf{t}_m;\mathbf{\Theta}))$ and so the stress is a function of $\mathbf{\Theta}$ and f. Then, find the function f and the map $\{\mathbf{x}_n\}_{n=1}^N$ or $\mathbf{\Theta}$ that produce the minimal stress (for which there are several algorithms available).

If f is constrained to be monotonic, the ordering of the distances will be preserved (even if they are nonuniformly stretched or shrunk). Such MDS is called *metric.* The particular case where f is the identity (or linear) and the stress function is $\sum_{n,m=1}^N (\delta_{nm} - d_{nm})^2$ is called *classical scaling* and has an analytic solution which, if the distances come from a data set in an Euclidean space, is basically equivalent to PCA. If only the ordering of the proximities is used (but not their values), the MDS is called *nonmetric* or *ordinal* and is particularly suitable when the distances are qualitative and only determine a rank

Once f and $\{\mathbf{x}_n\}_{n=1}^N$ or $\mathbf{\Theta}$ have been determined, the solution map can be freely translated and rotated (perhaps to appear in a more aesthetical way) without changing the value of the stress. f can be plotted together with the pairs (δ_{nm}, d_{nm}) in a scatter (or Shepard) diagram, that plots the distance in L-dimensional space versus the proximities. If the proximities are dissimilarities (i.e., dissimilar objects have a large proximity value), it will be a rising pattern; otherwise, it will be a falling one. If each of the objects has an associated value y_n, regression can be performed on the generated map to further help to interpret the data: $\{(\mathbf{x}_n, y_n)\}_{n=1}^N$.

Degeneracy can happen if the objects have a natural clustering and the dissimilarities between objects in different clusters are (almost) all larger than the dissimilarities within each cluster. In this case, (almost) all points of a single cluster will converge to a single location, the stress will converge to 0 and the scatter diagram will be a staircase.

4.10.1.1 Selection of the map dimension

Obviously, the larger the dimension of the map is, the smaller the stress will be, but one should keep L as small as possible, ideally to match the intrinsic dimensionality of the data (in a visualisation application one will even force L to be less than 3, but for generic dimensionality reduction this is not necessary). Too small a dimension of the map can give a misleading view of the data. For example, points apparently clustered in a 2D map can actually lie far apart in a 3D one. For two-dimensional maps, a simple way to embed information from the original data in the map is to draw a line between every pair of objects whose proximity exceeds some threshold value: the presence of long, haphazardly crossing lines will indicate a discrepancy between closeness in the data and closeness in the space. Clusters will only be valid if they are consonant with the lines, i.e., points within a cluster should be well connected with each other and poorly connected with those outside the cluster. Sometimes the lines can connect many points in some nonlinear shape, like in figure 4.18, which suggests that only one curvilinear dimension would be enough to give a reasonable description of the data. This is called the *horseshoe phenomenon.*

To assure an adequate degree of statistical stability, the map dimension cannot be arbitrarily large for a given sample size. A rule of thumb often used in statistics (Kruskal and Wish, 1978) is that the number of (significant) pairs of objects should be at least twice the number of parameters to be estimated:

$$\frac{N(N-1)}{2} \geq 2NL \Rightarrow N \geq 4L+1$$

but this just gives a trivial upper bound on L.

Another rule of thumb, analogous to the scree plot for PCA (plot of the cumulative eigenvalue sum versus the number of components) is to plot the stress obtained for map dimensions $L = 1, 2, 3 \dots$ and try to determine a cutoff point; for example, where the slope of the stress curve changes abruptly, or where the stress falls below a certain threshold. As in PCA, there is no guarantee that such heuristic rules can find the intrinsic dimensionality. Besides, while running PCA for different numbers of components is immediate, for MDS it can be quite time consuming.

4.10.1.2 Problems of MDS

- There is no foolproof method to select the appropriate dimension of the map; one must try several.
- MDS does a much better job in representing large distances (the global structure) than small ones (the local structure).
- Contrarily to principal component analysis, in MDS one cannot obtain an $(L-1)$-dimensional map out of an L-dimensional one by dropping one coordinate (or, in general, by linearly projecting along some direction). That is, it does not verify the property of additivity mentioned in section 2.6.2.

4.10.1.3 The Sammon mapping

With the objective of preserving the interpoint Euclidean distances of a collection of real vectors $\{\mathbf{t}_n\}_{n=1}^N$, Sammon (1969) proposed a particular type of MDS where the low-dimensional Euclidean space points $\{\mathbf{x}_n\}_{n=1}^N$ are chosen to minimise the criterion

$$E(\{\mathbf{x}_n\}_{n=1}^N) \stackrel{\text{def}}{=} \frac{1}{\sum_{n<m}^N \delta_{nm}} \sum_{n<m}^N \frac{(\delta_{nm} - d_{nm})^2}{\delta_{nm}}$$

where δ_{nm} is the distance in $\mathbb{R}^D$ between $\mathbf{t}_n$ and $\mathbf{t}_m$ and d_{nm} is the distance in $\mathbb{R}^L$ between $\mathbf{x}_n$ and $\mathbf{x}_m$. Unlike in usual MDS, this criterion gives weight to small distances, which helps to detect clusters. The Sammon mapping is the mapping implicitly defined by the pairs $\{(\mathbf{t}_n, \mathbf{x}_n)\}_{n=1}^N$. Sammon (1969) proposed a diagonal Newton method to minimise the criterion function:

$$\mathbf{x}_{nl}^{(\tau+1)} = \mathbf{x}_{nl}^{(\tau)} - \eta \frac{(\partial E/\partial x_{nl})^{(\tau)}}{(\partial^2 E/\partial x_{nl}^2)^{(\tau)}}$$

with a "magical factor" $\eta \approx 0.3$ or 0.4 and, as starting point for the $\{\mathbf{x}_n\}_{n=1}^N$, random values or the projections on the first L principal components of the data. This algorithm is a poor minimiser according to Ripley (1996), who notes that it often achieves very bad local minima from random starting points and that it is difficult to set η to achieve convergence.

4.10.2 Methods for preserving the geodetic distances

In the problem we are concerned with, dimensionality reduction of continuous data, the distance between points is the Euclidean distance in $\mathbb{R}^D$ (the length of the straight line segment joining both points), or perhaps some other definition of distance, which depends only on the point coordinates. However, this distance is not useful because it ignores the data manifold: two points that are close in the Euclidean distance in $\mathbb{R}^D$ may be far from each other inside the (low-dimensional) manifold defined by the data, as fig. 4.19 illustrates. Inputting such distances to an MDS or Sammon method would produce a wrong representation. The relevant distances are really the distances along the manifold, or more precisely, the geodetic distances. The **geodetic distance** between two points of a manifold is defined as a minimum of the length of a path joining both points that is contained in the manifold, and such minimal-length paths are called **geodesics**[19] (do Carmo, 1976).

[19] Actually, a geodesic is usually defined as a curve on a manifold that has zero acceleration on the manifold, i.e., that the acceleration vector (second derivative of the curve with respect to some parametrisation) is perpendicular to the manifold and therefore its orthogonal projection on it is zero. From this definition follow some interesting minimising properties such as the one mentioned.

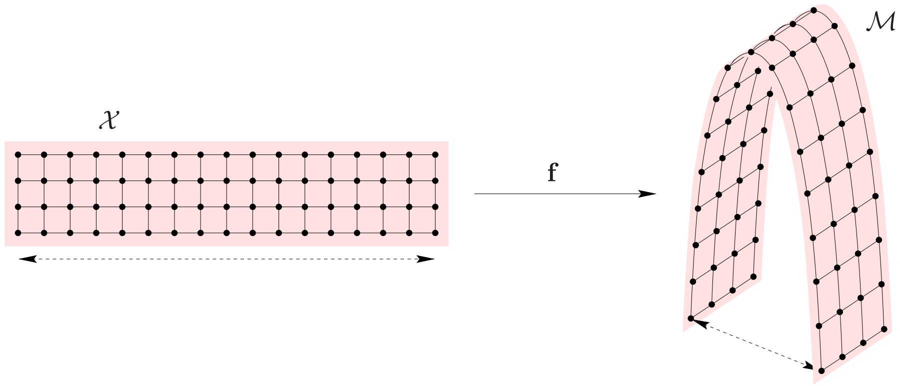


Figure 4.19: Distances along the straight line between two points (Euclidean distance) versus distances along a geodesic in the manifold (geodetic distance). Points close in the former sense may be far in the latter.

Depending on the manifold, there may be more than one geodesic between two given points (e.g. in a sphere, the two sections of a great circle passing through the two points are geodesics), but we will ignore that fact in this discussion.

Computing geodetic distances along arbitrary nonlinear manifolds is complicated, requiring the solution of the Euler-Lagrange equation for the arclength functional, which is a second order nonlinear differential equation. A computationally feasible approach consists of discretising the manifold as a Delaunay triangulated graph (Preparata and Shamos, 1985; Aurenhammer, 1991). Let us first define the Voronoi diagram. Given a collection of M vectors $\{\boldsymbol{\mu}_m\}_{m=1}^M$ in $\mathbb{R}^D$, the **Voronoi diagram** of $\mathbb{R}^D$ induced by those vectors is the collection of Voronoi cells $\{\mathcal{V}_m\}_{m=1}^M$ defined as

$$\mathcal{V}_m \stackrel{\text{def}}{=} \{\mathbf{t} \in \mathbb{R}^D : \ d(\mathbf{t}, \boldsymbol{\mu}_m) \leq d(\mathbf{t}, \boldsymbol{\mu}_n) \ \forall n \neq m\} \quad m = 1, \dots, M$$

i.e., cell $\mathcal{V}_m$ is the set of points closest to $\boldsymbol{\mu}_m$ than to any other vector according to a certain distance d defined in $\mathbb{R}^D$. Each Voronoi cell is a convex D-dimensional polyhedron (a D-polytope) and the union of all cells is the space $\mathbb{R}^D$. The dual of the Voronoi diagram is the **Delaunay triangulation** of the collection $\{\boldsymbol{\mu}_m\}_{m=1}^M$, defined as the graph with vertices $\{v_m\}_{m=1}^M$ (vertex v_m being associated with vector $\boldsymbol{\mu}_m$) and adjacency matrix $\mathbf{A} = (a_{mn})$ verifying

$$a_{mn} = \begin{cases} 1, & \mathcal{V}_m \cap \mathcal{V}_n \neq \varnothing \\ 0, & \mathcal{V}_m \cap \mathcal{V}_n = \varnothing \end{cases}$$

i.e., two nodes are connected if and only if their associated Voronoi cells are adjacent. Analogous definitions are derived for a manifold $\mathcal{M}$ of $\mathbb{R}^D$ by taking the intersection of each Voronoi cell with $\mathcal{M}$; the collection of links of the resulting *restricted Delaunay triangulation* is a subset of that of the unrestricted Delaunay triangulation (for $\mathcal{M} \equiv \mathbb{R}^D$). The key point to note is that the Delaunay triangulation restricted to a manifold carries the topological structure of the manifold and that it enforces a discretisation of paths along the manifold that allows to compute (approximate) geodetic distances, as well as to plan paths from one point to another. Planning of paths through geodesics also offers interesting possibilities to applications where interpolation in data space is required, such as in morphing or animating an object from one three-dimensional configuration to another one (Bregler and Omohundro, 1995; Tenenbaum, 1998). Figure 4.20 illustrates these ideas.

In the rest of this section, we describe an algorithm to obtain the restricted Delaunay triangulation (topology-preserving networks) and a method that uses it to perform MDS with geodetic distances (ISOMAP).

4.10.2.1 Topology-preserving networks

Martinetz and Schulten (1994) give an algorithm to obtain the Delaunay triangulation induced on a manifold. They call the resultant graph a *topology-preserving network*. Their approach is opposite to that of self-organising maps: in SOMs, one fixes the links of the graph (which determine its topology: linear array, planar grid, etc.) and tries to fit it to the data manifold. This does not work if the dimensionality or the topology of the graph do not match those of the data manifold. The algorithm of Martinetz and Schulten (1994) works the other way, specifying the nodes and then constructing only the appropriate links, which leads to the graph. The algorithm, showed in fig. 4.21, requires as input a set of reference vectors $\{\boldsymbol{\mu}_m\}_{m=1}^M$ in data space which

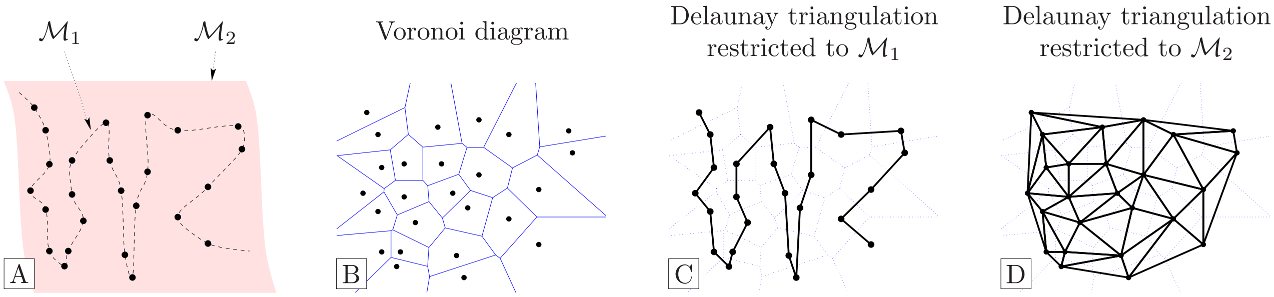


Figure 4.20: Voronoi diagram (graph B) and restricted Delaunay triangulation (graphs C and D). For a given collection of points (graph A), the restricted Delaunay triangulation depends on the manifold that embeds them: the dotted line $\mathcal{M}_1$ in graph C and the shaded area $\mathcal{M}_2$ in graph D. The points and the manifolds on graph A are the same as those of fig. 4.5.

inputs	
$\{\boldsymbol{\mu}_m\}_{m=1}^{M} \subset \mathcal{M} \subseteq \mathbb{R}^D$ uniformly distributed on $\mathcal{M}$	Reference vectors
$\mathcal{U}(\mathcal{M})$	Uniform distribution on $\mathcal{M}$
initialise	
Graph with nodes $\{v_m\}_{m=1}^{M}$ and no links	Topology-preserving network
repeat forever	
Draw $\mathbf{t}$ from $\mathcal{U}(\mathcal{M})$	
$m_1 \leftarrow \arg\min_{m=1,\dots,M} \lVert \mathbf{t} - \boldsymbol{\mu}_m \rVert$	Closest node
$m_2 \leftarrow \arg\min_{\substack{m=1,\dots,M \\ m \neq m_1}} \lVert \mathbf{t} - \boldsymbol{\mu}_m \rVert$	Second closest node
Connect nodes v_{m_1} and v_{m_2}	
end	
return graph	

Figure 4.21: Pseudocode of the construction algorithm for topology-preserving networks of Martinetz and Schulten (1994).

is *uniformly* distributed over the data manifold. The reason for this requirement is that the reference vectors must define the shape of the manifold. To each reference vector $\boldsymbol{\mu}_m$ we associate a node v_m in a graph in which initially there are no links. To derive the appropriate links, the algorithm uses an online rule that they term competitive Hebbian: given a point drawn uniformly from the data manifold $\mathcal{M}$, connect the nodes associated with the two reference vectors closest to it. The rule is competitive because only the link between the two winners (the two closest reference vectors) is updated; and it is Hebbian because, if we assume that each node has an activity proportional to the proximity of its associated reference vector to the data point presented, then the connection between two nodes is strengthened if both nodes have a high activity. It is intuitively clear that this must converge to the Delaunay triangulation restricted to $\mathcal{M}$ and indeed Martinetz and Schulten (1994) prove so under the assumption that the reference vectors are (using their nomenclature) *dense* in the data manifold $\mathcal{M}$: for any point $\mathbf{t}$ in $\mathcal{M}$, the triangle formed by $\mathbf{t}$ and the two closest reference vectors must be contained in $\mathcal{M}$. However, this definition is only appropriate for manifolds of the same dimensionality as the embedding space $\mathbb{R}^D$, since it trivially holds for convex manifolds of any dimensionality and never holds for nonconvex manifolds of lower dimensionality.

Clearly, if the data manifold coincides with the data space then an unrestricted Delaunay triangulation is obtained. While we are concerned here with the Delaunay triangulation restricted to a low-dimensional data manifold, the unrestricted Delaunay triangulation is interesting in its own right for many other situations, such as the finite-element method and other geometric and graph-theoretical problems mentioned by Martinetz and Schulten (1994).

The number of links from a node in a fully-connected graph of M nodes is $M-1$, which leads to $\mathcal{O}(M^2)$ for the total number of links in the graph. But if only local connections are allowed, as in the Delaunay triangulation, the average number of links from a node becomes proportional to L (the dimensionality of the graph), and so the total number of links in the graph becomes $\mathcal{O}(LM)$. In reality, this is $\mathcal{O}(Le^L)$ since the number of reference vectors necessary to represent an L-dimensional manifold is exponential in L.

The algorithm of Martinetz and Schulten (1994) has several problems:

- There is no way to assess convergence. The fact that points are blindly drawn from a uniform distribution on $\mathcal{M}$ means that a lot of them will be redundant, since they will establish links already done and we may need to wait very long until a point is drawn that establishes a necessary link. We presume that the algorithm will build a proportion of the Delaunay links early during training, with the remaining ones appearing very slowly afterwards—the slower the higher the manifold dimensionality is. Thus, the graph obtained when the algorithm is stopped after a certain number of iterations will probably contain many missing links if the number of reference vectors is large. This problem would disappear if there was a way of setting to zero the probability of the manifold region affecting a link just created (so that no further samples are drawn there).

  In practice, a finite data sample $\{\mathbf{t}_n\}_{n=1}^N$ is used, which is all the information we have about the data manifold. This ensures that the algorithm stops after N point presentations, but the resulting graph is affected by the factors mentioned below.

- It is not easy to distribute uniformly the reference vectors over an arbitrary manifold, in particular when the manifold is defined by a data sample (our case). Directly using a data sample (that, ignoring the noise for the moment, belongs to the data manifold by definition) is unlikely to work in general, since the data distribution over the manifold need not be uniform (e.g. see the sample in fig. 2.13). This means that the triangulation found will be too coarse in low probability areas of the manifold and far too dense in high-probability areas.

- It is difficult to decide how many reference vectors to use and the final triangulation is very sensitive to this. The number of reference vectors is intuitively related to the Nyquist (spatial) frequency, since they are effectively a sample of $\mathcal{M}$ from which $\mathcal{M}$ could be reconstructed in principle. It also must depend exponentially on L, as the volume of $\mathcal{M}$ does. This is again a result of the curse of the dimensionality and is analogous to the situation found with the Monte Carlo sampling of the latent space in latent variable models (section 2.4).

  For storage efficiency and speed of training, the number of reference vectors should be kept as small as possible. This means selecting a small subset of the data sample—which may severely reduce the number of reference vectors in low-probability areas of the manifold. Thus, a critical point of the algorithm is to select a subset of M points from the data sample such that it spans the data manifold uniformly and M is neither too large (which would lead to many missing links in the graph) nor too small (which would yield a poor approximation of the manifold and of the geodetic distances).

- Sensitivity to noise. The reference vectors are assumed to lie in the data manifold, i.e., the noise is considered zero—a clearly untenable assumption in many practical problems. If the noise level is high enough many points will fall out of the data manifold (if the noise was such that the points were perturbed inside the manifold there would be no problem, of course, but this is never going to happen). Such points, in particular outliers, will result in the establishment of links between reference vectors which are far away in the manifold, distorting the topological structure of the graph. Thus we expect the graph representation to degrade ungracefully with the noise level.

4.10.2.2 The ISOMAP algorithm

Tenenbaum (1998) has proposed a straightforward combination of the topology-preserving network algorithm with multidimensional scaling for manifold modelling that he calls ISOMAP: (1) to use ordinal multidimensional scaling with the geodetic interpoint distances computed with the method of Martinetz and Schulten (1994) to derive a low-dimensional Euclidean map and then (2) fit a radial basis function network to the mapping from the low-dimensional points to the observed ones or vice versa, thus defining reconstruction and dimensionality reduction mappings, respectively. In particular, ISOMAP works as follows. Given a sample $\{\mathbf{t}_n\}_{n=1}^N \subset \mathbb{R}^D$, it randomly selects M reference vectors from it and constructs an approximation to the restricted Delaunay triangulation graph whose nodes are associated with those M reference vectors, using the method of Martinetz and Schulten (1994). The infinite sequence of points uniformly distributed over the data manifold that this algorithm requires is approximated by the data sample $\{\mathbf{t}_n\}_{n=1}^N$. Once such graph has been obtained, it is used to compute the geodetic distances between every two reference vectors using Floyd's algorithm, whose complexity is $\mathcal{O}(M^3)$. Tenenbaum claims to be able to determine the intrinsic dimensionality of the data manifold by trying several dimensions and plotting the stress obtained, but, as mentioned in section 4.10.1.1, this heuristic guarantees very little. ISOMAP then uses an RBF network to implement the mapping between reference vectors and the low-dimensional points found by the ordinal MDS.

ISOMAP is a promising method, but it inherits the problems of all the methods it is based on, namely the lack of guarantee that the reference vectors uniformly span the data manifold[20] and the potential problems of local minima when approximating implicitly defined mappings via a universal mapping approximator. It also has a high computational complexity: $\mathcal{O}(M^3)$ to compute the $\mathcal{O}(M^2)$ geodetic distances, where the number of reference vectors M is $\mathcal{O}(e^L)$, plus several trial-and-error MDS runs on those distances (till a convenient dimension L is determined), plus the final RBF fit. Tenenbaum (1998) claims that the geodetic distances computed from the graph are very good approximations to the true ones for some toy examples, although the distances are slightly overestimated due to the discretisation of the manifold. Since the version of MDS used, coincident with eq. (4.4), represents large distances much better than small ones, as mentioned in section 4.10.1.2, the net effect may be a misrepresentation of distances at all scales.

Tenenbaum et al. (2000) present a variation of ISOMAP where (1) the topology-preserving network is constructed, instead of with the algorithm of Martinetz and Schulten (1994), simply by linking every data point with its K nearest neighbouring data points (or with all other data points within a distance ϵ of itself), with each link weighted by the corresponding distance; and (2) an easy-to-minimise stress function with a single minimum is used, akin to classical scaling (solvable via the principal components of a matrix derived from the geodetic distances). Tenenbaum et al. (2000) state that, in the limit of infinite training data, this new scheme recovers the true dimensionality and geometric structure of the data manifold if this belongs to a certain class of Euclidean manifolds (which excludes manifolds such as hemispheres and tori). They give a proof based on the fact that as training data increases, the graph becomes denser and better approximates the true geodetic distances; the actual convergence rate depends on unknowns such as the curvature of the data manifold, the separation between branches and the data density. Unfortunately, in practice this proof is of little use because, generally, training data in high dimensions will be scarce and not uniformly distributed on the manifold; and the computational complexity (at least quadratic in the number of data points) would preclude using many data points anyway. Also importantly, step (1) now crucially depends on the neighbourhood size (K or ϵ): if too large, it will include data points from other branches of the manifold, shortcutting them and resulting in wrong geodetic distances; if too small, it may not contain enough neighbours. Determining a good neighbourhood size may be difficult in practice. Finally, while (2) is faster than an arbitrary MDS (metric or ordinal), it is also more restrictive: its unique minimum may not be as good as some of the local minima of an arbitrary MDS. Other problems mentioned earlier remain: sensitivity to noise, bad local minima when explicitly learning a dimensionality reduction mapping and no foolproof way to determine the intrinsic dimensionality.

ISOMAP also needs a more thorough evaluation than the fact that it seemed to work in the examples given by Tenenbaum (1998) and Tenenbaum et al. (2000). Particularly dubious are the results regarding the problem of finding the manifold spanned by a set of bitmapped face images in various azimuth and elevation view angles, where ISOMAP magically picks the (to a human perceptually salient) azimuth and elevation degrees of freedom but not the also prominent changes in illumination and translation and the noise. The data set used (10 000 images) is also small given the dimensionality of the images (32×32). Similar criticism applies to the manifold spanned by the images of a hand undergoing non-rigid articulated motion, which cannot be modelled with only two degrees of freedom (Amir Assadi, personal communication).

4.10.2.3 Summary

Representing a manifold by a skeleton graph from a noisy sample from the manifold includes two steps:

- finding an approximate set of vectors that are uniformly spread on the manifold
- finding the restricted Delaunay triangulation of those points.

The algorithm of Martinetz and Schulten (1994) is a first step toward this objective. This skeleton graph can then be used to compute pairwise geodetic distances, feed them into an ordinal MDS method and solve the nonlinear regression between the low-dimensional map and the high-dimensional sample with a universal

[20] The toy examples offered by Tenenbaum (1998), as well as those by Martinetz and Schulten (1994) and Tenenbaum et al. (2000), have the data sample uniformly distributed on the manifold by definition. In this ideal case, it is easy to obtain a collection of reference vectors uniformly spread over the manifold by applying a vector quantisation algorithm (Martinetz and Schulten (1994) use the neural gas) or even by simply choosing a random subset of a data sample.

mapping approximator. This is the basis of the ISOMAP procedure of Tenenbaum (1998). Defining manifolds via the geodetic distance is an exciting idea and a promising avenue for further research.

Preservation of distances can also be imposed on other models, for example GTM. Tenenbaum (1998) and Marrs and Webb (1999) have criticised the GTM model because it often finds estimates that offer a distorted view of the high-dimensional manifold. That is, the distances along the data manifold in data space are stretched or shrunk in a complex way in different amounts in different regions of the latent space—even though the mapping $\mathbf{f}$ may be approximating well the data manifold. In section 2.8.3 we mentioned the disadvantages that this has for visualisation of structure in the data, even though such a distorted coordinate system is as valid as any other to which it can be invertibly mapped. Anyway, it would be good to be able to enforce the preservation of geodetic distances in GTM. Marrs and Webb (1999) try to achieve this via a regularisation term in the log-likelihood that constrains GTM's mapping from latent onto data space to be unit-speed on the average, so that unit steps in latent space correspond to unit steps in data space on the average. We discussed this approach in section 2.8.3.

4.10.3 Locally linear embedding

Roweis and Saul (2000) have recently proposed a dimensionality reduction method that they call *locally linear embedding* (LLE). Given a collection $\{\mathbf{t}_n\}_{n=1}^N$ of training points, it uses a linear mapping to capture local neighbourhood relations—representative of the local geometry of the data manifold—that are then preserved as much as possible in an associated low-dimensional collection of points $\{\mathbf{x}_n\}_{n=1}^N$, which is the end product, just as in MDS. Specifically, each data point is reconstructed by least squares as a linear combination of its K nearest neighbouring data points (or of all other data points within a distance ϵ of itself):

$$\hat{\mathbf{t}}_n = \sum_{m\in\mathcal{N}(\mathbf{t}_n)} w_{nm}\mathbf{t}_n \qquad \min E_1\left(\{w_{nm}\}\right) \stackrel{\text{def}}{=} \sum_{n=1}^N \left\|\mathbf{t}_n - \hat{\mathbf{t}}_n\right\|^2 \text{ subject to } \sum_{m\in\mathcal{N}(\mathbf{t}_n)} w_{nm} = 1$$

where $\mathcal{N}(\mathbf{t}_n)$ is the set of neighbours of data point $\mathbf{t}_n$. The least-squares problem is solved for the optimal weights w^*_{nm} as N separate linear systems of $K\times K$ equations, regularised if $K<D$ (in which case the systems are underdetermined). The error function E_1 is invariant to translations (since the weights sum one), rotations and uniform scalings of the neighbourhoods. Ideally, minimising E_1 results in the weights characterising local, intrinsic geometric properties of the data manifold. Such properties can then be preserved in a low-dimensional representation of the data by imposing on it the weights, again as an objective function to be minimised by least-squares, but this time over the low-dimensional points $\{\mathbf{x}_n\}_{n=1}^N$ for constant weights:

$$\min E_2\left(\{\mathbf{x}_n\}_{n=1}^N\right) \stackrel{\text{def}}{=} \sum_{n=1}^N \left\|\mathbf{x}_n - \hat{\mathbf{x}}_n\right\|^2 \qquad \hat{\mathbf{x}}_n = \sum_{m\in\mathcal{N}(\mathbf{t}_n)} w^*_{nm}\mathbf{x}_n.$$

The points $\{\mathbf{x}_n\}_{n=1}^N$ are constrained to be zero-mean and unit-covariance to eliminate arbitrary translations and rotations. The minimisation is then a constrained quadratic programming problem with a unique minimum that can be found by solving an eigenvalue problem. Note that it cannot be decomposed into an independent subprogram for each n as before, i.e., the local neighbourhood relations interact. However, it is additive in the same sense PCA is: the map with $L+1$ dimensions is obtained from the one with L by computing the next eigenvalue. The method can also be run using as input the pairwise distances between data points instead of the data points themselves, just as in MDS. As usual, a mapping approximator can be fit to the pairs $\{(\mathbf{t}_n,\mathbf{x}_n)\}_{n=1}^N$ to obtain a dimensionality reduction mapping. This will define a single global mapping, unlike the local dimensionality reduction methods of section 4.7.

LLE is attractively simple and has similarities with MDS methods, in particular with ISOMAP. In fact, it can be considered as a topography-preserving MDS, where the local topography is preserved by linear neighbourhood relations rather than by the pairwise distances. It also shares some of the disadvantages of MDS methods: (1) likely sensitivity to noise and to the training set used; (2) sensitivity to data density that is not uniform on the data manifold (which makes difficult to obtain local neighbourhoods in low-density regions); (3) no dimensionality reduction mapping is defined, the only way being to fit a mapping approximator to the pairs $\{(\mathbf{t}_n,\mathbf{x}_n)\}_{n=1}^N$, which is prone to bad local minima; (4) quadratic complexity on the training set size, both when computing the neighbourhood weights and when solving the eigenvalue problem for the low-dimensional points; (5) no foolproof way to determine the intrinsic dimensionality; and finally and crucially, (6) no foolproof way to determine the neighbourhood size K, which determines how local the geometric properties captured by the weights are. Locality implies a small neighbourhood (more so since the implementation is linear) but not so small that no neighbours are available.

Roweis and Saul (2000) apply LLE to a toy problem, to a face images dataset like that of ISOMAP and to a word categorisation example. More thorough evaluations are necessary that elucidate the impact of the caveats mentioned.

4.11 Conclusions

We have defined the problem of dimensionality reduction as the search for an economic coordinate representation of a submanifold of a high-dimensional Euclidean space, a problem so far not yet solved in a satisfactory and general way. We have then given an overview of several techniques for dimensionality reduction.

In general, the central place occupied by PCA has not been taken by any nonlinear technique. PCA remains the favourite feature extractor, particularly for very high-dimensional data (such as images), where it is typically used in a preprocessing stage. Projection pursuit is another linear dimensionality reduction technique, guided by an arbitrary criterion rather than by PCA's (maximal variance or, equivalently, minimal L_2-error). Kernel PCA is an attractively simple nonlinear extension of PCA that has performed well in some applications as feature extractor. However, more complete performance comparisons are necessary, and more importantly some theoretical insight in the interpretation of the nonlinear components, the effect of the kernel and the definition of dimensionality reduction and reconstruction mappings.

Local dimensionality reduction, particularly PCA-based, is a good and reasonably fast approach that combines some of the benefits of PCA with a nonlinear model: economy of parameters and flexibility. However, it is not very accurate unless many local models are used. Global nonlinear dimensionality reduction (e.g. autoassociators) requires many parameters and training is difficult, requiring much time and data and being very prone to bad local minima.

The definition of principal curves makes them intuitively appealing. However, while they have been used for small dimensions (Hastie and Stuetzle, 1989; Banfield and Raftery, 1992), they are not mature enough algorithmically for high-dimensional dimensionality reduction.

Two classes of methods, those based on topological vector quantisation (section 4.9) and those based on MDS with either straight-line or geodetic distances (section 4.10), have as primary product a collection of reference vectors in observed space and an associated collection of low-dimensional vectors preserving topology and metric structure, respectively. The disadvantage of these methods is that the data manifold and the dimensionality reduction and reconstruction mappings are defined implicitly through the pairs $\{(\boldsymbol{\mu}_m, \mathbf{x}_m)\}_{m=1}^M$ (for vector quantisation) or $\{(\mathbf{t}_n, \mathbf{x}_n)\}_{n=1}^N$ (for MDS). Defining such mappings explicitly requires fitting in a supervised way a mapping approximator to that set of pairs. In an ideal situation, the dimensionality of the $\mathcal{X}$ space is appropriate, and the associations $\mathbf{x}_m \leftrightarrow \mathbf{t}_m$ are correct and are instances of a one-to-one mapping $\mathcal{X} \underset{\mathbf{F}}{\overset{\mathbf{f}}{\rightleftarrows}} \mathcal{M} \subset \mathcal{T}$ (a bijection). But even in this best case, fitting a global flexible model to either of the mappings, $\mathbf{f}$ or $\mathbf{F}$, is likely to end up in bad local minima, depending sensitively on the starting point of the optimiser, the optimiser itself and the architecture of the approximator. Some knowledge about the manifold must be injected, probably via a regularisation term in the objective function, but it is not clear how well the final mappings will generalise to unseen data. For MDS it is possible to define the low-dimensional vectors via an explicit dimensionality reduction mapping, although the problem of bad local minima is still serious. Two further problems of distance-based methods are (1) their high computational complexity, at least quadratic on the training set size; and (2) the fact that the training data should be uniformly distributed over the data manifold, since such methods try to model the geometry of that manifold—for which the training set acts as a scaffold—but not the density of the data.

None of the methods discussed in this chapter define a density model for the data (except PCA and the elastic net in their probabilistic interpretation). While this is not necessary to obtain dimensionality reduction and reconstruction mappings, it is to know the distribution over the manifold and the latent space. Self-organising maps give an indication of the density over the manifold via the distribution of the reference vectors on it, but as mentioned in section 4.9.1 they do not properly define a density model. The advantages of probabilistic methods over non-probabilistic ones are discussed in chapter 11. One somewhat surprising property of dimensionality reduction with probabilistic models is that, due to the noise model, the dimensionality reduction and reconstruction mappings are not the inverse of each other (specifically, $\mathbf{F} \circ \mathbf{f}$ is not the identity, as happens for probabilistic PCA; see section 2.9.1). However, basing the dimensionality reduction on an Euclidean distance criterion (to the low-dimensional, nonlinear manifold) leads to discontinuous mappings.

Two central ideas underlie several of the methods:

- Points close together in data space should be mapped close together in low-dimensional space (vector

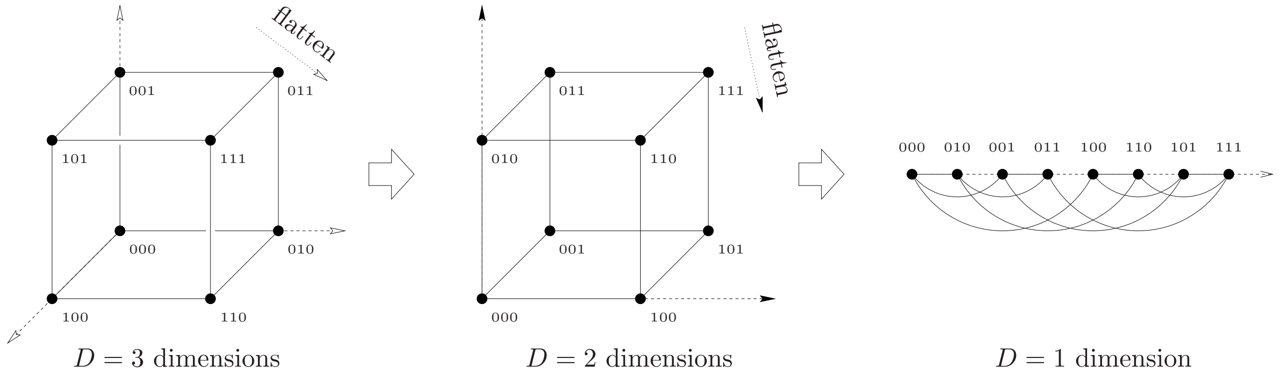


Figure 4.22: The topological structure of a quantised high-dimensional Euclidean space can be preserved in lower dimensions (even one dimension) by *folding* or *flattening* it and preserving the connections. In the low-dimensional representation, the connections still reflect the topology of the high-dimensional space, not of the low-dimensional one. The representation can be further generalised by labelling the connections with *distance* or *similarity* values.

quantisation methods, such as self-organising maps and the elastic net, and MDS methods).

- The manifold should pass through the middle of the data (principal curves, nonprobabilistic PCA).

Two major issues that remain open are:

- To overcome the curse of the dimensionality, which results in many parameters that demand huge sample sizes to obtain reasonable results. Most of the techniques reviewed still suffer of this to an extent.
- To determine the intrinsic dimensionality of a distribution given a sample of it. Knowing it would reduce the possibility of over- or underfitting. Model selection techniques, such as those based on the value of an error criterion as a function of the number of dimensions used (as in PCA or MDS), are unreliable.

4.12 Can dimensionality reduction be achieved with discrete variables?

We have discussed dimensionality reduction of continuous observed variables in terms of continuous latent variables, as in PCA, autoassociators or methods based on multidimensional scaling. This is due to the fact that topographic mappings are naturally defined in continuous Euclidean spaces thanks to their intrinsic definition of distance, whatever the dimensionality of the space. Discrete spaces can work like continuous spaces by assuming that they are a sample or discretisation of an underlying continuous space and ensuring that the learning algorithm respects the topographic structure of that space, as happens with GTM or self-organising maps. In such cases, the higher the resolution of the sample, the better the collection of discrete values will approximate the underlying continuous space. Therefore, this use of discrete variables is essentially not different from using continuous latent variables—it is just that we cannot use high-dimensional continuous variables directly in an analytically exact, or otherwise desirable, way. In particular, the dimension of the manifold induced in observed space is the same as that of the underlying low-dimensional continuous space (e.g. 2 for a two-dimensional grid in a self-organising map).

We mention here a different possibility, without going into any detail of how it could actually be implemented. If a Euclidean space of dimension D is discretised into regions, we can represent the neighbourhood relations between these regions (and thus their topological structure) with a graph, as in fig. 4.22(left). This graph, which is D-dimensional, can be flattened to any lower dimension, up to 1 (fig. 4.22(right)) while still keeping the topological information of the D-dimensional space via the graph edges. In the figure, the arrangement on the right is apparently one-dimensional but can be unfolded to reveal a three-dimensional structure; this is revealed by the presence of long-range connections in the one-dimensional arrangement (extending farther than the one-dimensional nearest neighbours). The graph edges can be labelled with numerical values representing distance or similarity.

One cannot help comparing this to the complex pattern of short- and long-range connections existing in different areas of the brain. The primary visual cortex of mammals, usually considered as a two-dimensional sheet of neurons, is known to have orderly representations of a number of external stimuli, such as position in the visual field, eye of origin, orientation, direction of movement, spatial frequency and disparity. In fact, dimensionality reduction models such as self-organising maps and the elastic net have been very successful at replicating such organisation (Swindale, 1996). In the hippocampus, the pattern of connections is more complex, with no two-dimensional topography, which might indicate that it is coding higher dimensions. It would be interesting to investigate the alternative possibility of a discrete, two-dimensional collection of neurons that actually unfolds into a higher-dimensional stimuli space.

Chapter 5

Dimensionality reduction of electropalatographic (EPG) data

In this chapter we apply some of the latent variable models described in chapter 2 to a real-world problem, the dimensionality reduction of electropalatographic data. We obtain results of interest for electropalatography (adaptive data reduction and visualisation) and for latent variable modelling (relative performance of each model).

The chapter is organised as follows. Section 5.1 introduces electropalatography and the convenience of dimensionality reduction of electropalatographic data. Section 5.2 reviews data reduction methods for EPG data and explains the advantages that unsupervised learning methods (in particular latent variable models) have over fixed EPG data reduction methods. Section 5.3 describes the data set employed for the experiments and sections 5.4–5.5 the experiments with the different models employed. The chapter is completed with a discussion of the results in sections 5.6–5.7.

5.1 Introduction

5.1.1 The technique of electropalatography (EPG)

The technique of electropalatography (EPG) (for reviews, see e.g. Hardcastle et al., 1989 or chapter 10 of Hardcastle and Hewlett, 1999) is well established as a relatively noninvasive, conceptually simple and easy-to-use tool for the investigation of lingual activity in both normal and pathological speech. It records details of the timing and location of lingual contacts with the hard palate during continuous speech. Typically, the subject wears an artificial palate moulded to fit the upper palate with a number of electrodes mounted on the surface to detect lingual contact (62 in the Reading EPG system; Hardcastle et al., 1991a); two such pseudopalates are shown in fig. 5.1. The EPG signal is sampled at 100–200 Hz. Thus, for a given utterance, the sequence of raw EPG data consists of a stream of binary vectors with both spatial and temporal structure, such as the one showed in fig. 5.2. Careful observation of both temporal and spatial details of contact across the entire palatal region can be very helpful to identify many phonetically relevant details of lingual activity.

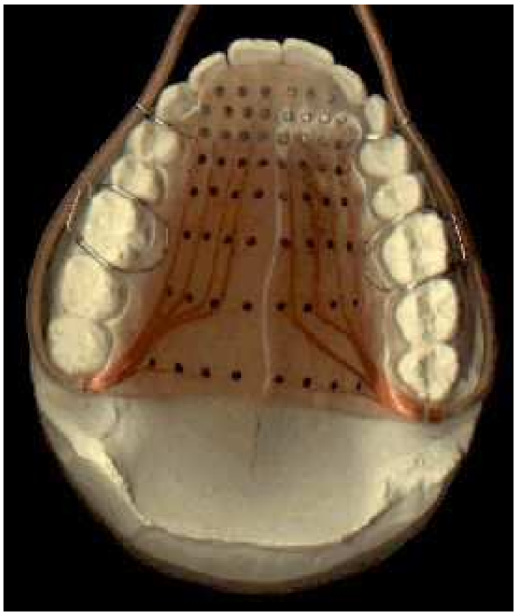
The technique is suitable for investigation of those sounds that involve measurable amounts of lingual contact with the hard palate. Varying degrees of contact occur for all English lingual obstruents, such as /t, d, k, g, s, z, ʃ, ʒ, ʧ, ʤ, tr, dr/, the palatal approximant /j/, nasals /n, ŋ/, lateral approximant /l/ (with varying degrees of contact depending on the nature of the following vowel) and relatively closed vowels and diphthongs such as /iː, ɪ, eɪ, ɑi, ɛ/. There is usually little contact with back vowels such as /u, ʊ, ɑʊ, ɔː/ and relatively open vowels such as /æ, ɑː, ɒ, ə, ʌ/.

EPG has been successfully used to study a number of phenomena in phonetic descriptive work, in studies of lingual coarticulation[1] and in the diagnosis and treatment of a variety of speech disorders (Hardcastle et al., 1991a, 1989). In the latter case, EPG has been useful in two main areas:

- Assessment: EPG provides insights into possible origins of auditorily perceived errors.

This chapter is mainly based on references Carreira-Perpiñán and Renals (1998a,b).

[1] Coarticulation refers to the simultaneous or overlapping movements of different articulatory organs or different parts of the same organ resulting in sounds being influenced by their phonetic context. It can be *anticipatory*, e.g. a stop being affected by the nature of a preceding vowel; or *carryover*, e.g. the opposite case. See also section 10.1.1.5.



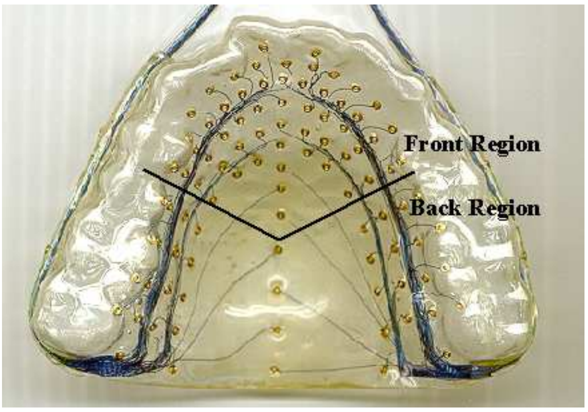


Figure 5.1: EPG pseudopalates. *Left*: Reading EPG system (Arnfield, Feb. 1, 2000), from the Speech Laboratory of the University of Reading. *Right*: UCLA Phonetics Lab system (UCLA, Feb. 1, 2000). Note the different electrode layout. Pictures used with permission of the Reading Speech Lab and the UCLA Phonetics Lab, respectively.

- Therapy: real-time visual feedback can be effective in the remediation of certain types of intractable speech problems.

For descriptive purposes, particular speech gestures can be characterised by **quasi-static patterns** of contact, such as those shown in fig 5.3. These patterns typify the contacts that have been found to occur at particular phases during the production of various speech sounds, and actual contact patterns that occur during normal spontaneous speech can be described with reference to these quasi-static patterns; Hardcastle et al. (1991a) give a more thorough description. Figure 5.3 shows some representative EPGs for the typical stable phase of different phonemes; these are to be compared with the EPG prototypes of fig. 5.6.

The disadvantages of EPG are:

- Currently, the technique records the location of the contact only: there is no direct information about the proximity of the tongue to the palate[2] or the overall shape of the tongue, nor is there any way of inferring directly which part of the tongue is producing the particular contact pattern. Therefore, in the numerous cases where the vocal tract configuration involves little or no contact with the hard palate, the technique provides little information.
- Since the electrodes are discrete points on the surface of the artificial palate, continuous lingual contact cannot be assumed when two adjacent electrodes are contacted (this is particularly relevant in the lateral parts of the palate).
- Each artificial palate must be individually manufactured from dental moulds of the speaker.
- Sometimes the artificial palate in the speaker's mouth may hinder their ability to produce normal speech.

Due to these limitations, it is advisable to adopt a multichannel approach to the investigation of lingual activity and to supplement EPG data with information from movement tracking devices, aerodynamic data and an audio signal representation. Nevertheless, despite these limitations it has provided useful information on many aspects of articulation, particularly in the study of lingual coarticulation.

[2]Although some systems for obtaining three-dimensional palate diagrams are being developed (Jones and Hardcastle, 1995).

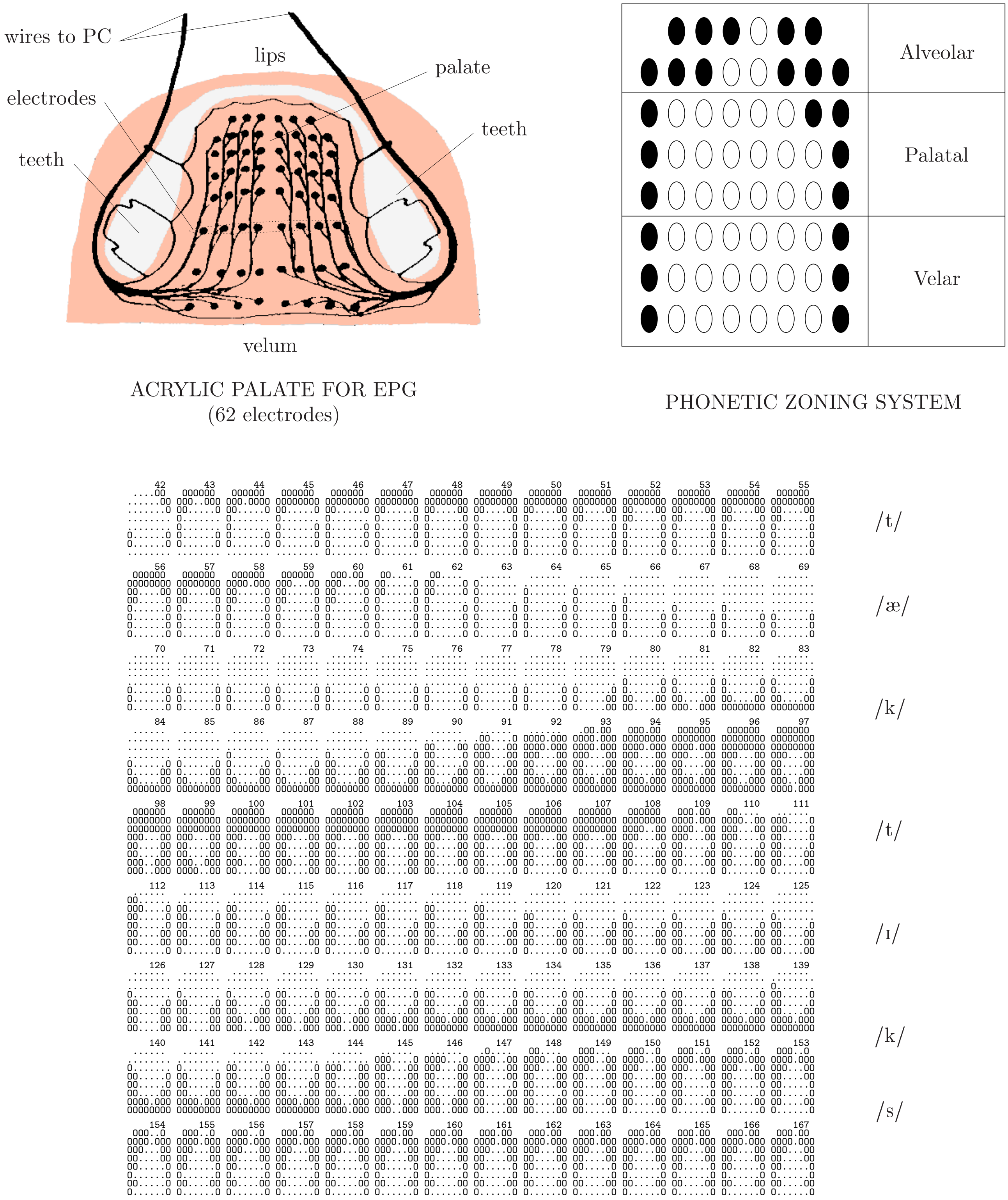


Figure 5.2: The Reading EPG system. *Upper left*: placement of electrodes on the artificial palate. *Upper right*: division of electrodes into zones: rows 1–2 alveolar zone, rows 3–5 palatal zone, rows 6–8 velar zone. Lingual contact is indicated by full, black dots (as in fig. 5.6) for a sample phoneme. *Lower*: full EPG printout of the word "tactics." In each palatal diagram the alveolar region is at the top, and the velar region at the bottom; lingual contact is indicated by zeroes. Interval between frames is 10 ms. Adapted with permission from Nicolaidis et al. (1993) (copyright John Wiley & Sons Limited).

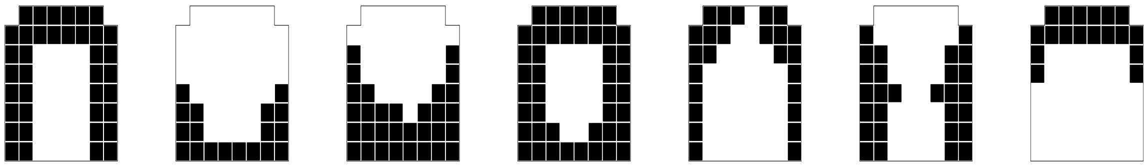


Figure 5.3: Quasi-static EPG contact patterns found in normal speech (from left to right): alveolar stop pattern; velar stop pattern; palatal stop pattern; double alveolar-velar pattern; alveolar grooved pattern; palatal grooved pattern; apical pattern. From Hardcastle et al. (1991a).

5.1.2 The tongue and its mechanical constraints

The human tongue is a muscular hydrostat (Kier and Smith, 1985), i.e., an organ composed almost entirely by muscle, lacking typical systems of skeletal support. Examples of muscular hydrostats are the tongues of mammals, the arms and tentacles of cephalopod molluscs and the trunks of elephants.

The main property of hydrostats is that their volume is constant: any decrease in one dimension will cause a compensatory increase in at least one other dimension. The action of different systems of muscles (grouped according to their direction: transverse, radial, circular, longitudinal and oblique) produces a variety of movements including elongation, shortening, bending, torsion, leverage and stiffening. The musculature acts both as effector of the movement and as support for the movement (the latter being the role of the skeleton). In the case of the tongue, the hard palate acts as support for the tongue in most consonants (e.g. alveolars) and some vowels, which facilitates production of tongue shapes different from those produced by the free-standing tongue.

Potentially the tongue has many degrees of freedom owing to its lack of bony skeleton (Stone, 1991); however, a number of studies suggest that tongue movements in speech may be appropriately modelled using a few elementary articulatory parameters (e.g. Nguyen et al., 1994, 1996 and references therein). As figure 5.2 shows, an EPG sequence presents redundancy at two levels, due to physical constraints of the tongue-palate system:

- Intraframe, or **spatial**, redundancy arises due to the limited number of possible tongue configurations—only a tiny fraction of the potential $2^{62} \approx 4.6 \cdot 10^{18}$ EPG patterns can be produced by the articulatory system. For example, the tongue cannot fold into arbitrary shapes or go through the palate. This suggests the use of a static dimension-reduction method in which the temporal interdependence of different vectors is disregarded.

- Interframe, or **temporal**, redundancy arises due to the correlations between neighbouring frames: the position of the tongue (and thereby the EPG signal) changes slowly with time, compared to the acoustic signal. This suggests the use of a model for vector time series, e.g. a hidden Markov model (HMM), coupled with a static dimension-reduction method.

If the appropriate constraints were known, then it would be possible to represent the EPG signal using fewer dimensions.

5.1.3 Dimensionality reduction of EPG data

In this chapter we approach the problem of finding a dimensionality reduction mapping for EPG data, not from a physical point of view, but from a machine learning one. That is, we consider a data set consisting of a number of EPG patterns and try to learn a mapping into a low-dimensional space from the data, without any a priori assumption about it (other than a specific, but quite general, form). We will use the latent variable modelling approach described in chapter 2. In common with most work on EPG data reduction we will concentrate on dimensionality reduction at the spatial level only, thus ignoring the constraints arising from coarticulation and other dynamics of the articulatory system. Of course, one can always analyse the temporal evolution of the dimension-reduced frames.

The primary advantage of dimensionality reduction for EPG data is that it makes a sequence of EPG data more amenable to analysis, since direct manipulation of the raw EPG sequence is cumbersome due to its high dimensionality (Hardcastle et al., 1991b). A reduced-dimension representation in terms of a few variables can be inspected, plotted against time or against each other, etc. more easily. Additional advantages are the consequent reductions in storage space and data transmission rate—albeit of little relevance, since the EPG data has relatively low requirements for them.

The resultant dimensionality reduction may suggest a value for the intrinsic dimensionality of the EPG data, i.e., the number of degrees of freedom of the tongue-palate system in what concerns the generation of EPG patterns.

In chapter 10 we use latent variable models for EPG and acoustic data to look into the problem of the acoustic-to-articulatory mapping, in which values for articulatory parameters such as vocal tract area functions, lip positions and jaw dynamics are inferred from the acoustic signal. However, the EPG signal alone is an incomplete articulatory description, omitting such details as nasalisation and vocalisation. Hence, the mapping from phonemes to EPG patterns is not one-to-one since certain phonemes (e.g. /æ/ and /ɑ/) can produce the same EPG patterns.

5.2 Review of data reduction methods for EPG data

5.2.1 Fixed data reduction: EPG indices

Several techniques to extract features or other kind of condensed information from the EPG data have been developed (reviewed in Hardcastle et al., 1991b; Jones and Hardcastle, 1995), such as:

- *Totals* display: place of articulation measure to monitor dynamic changes in different contact regions of the palate. The palate is divided into reference zones (e.g. alveolar: rows 1–2; palatal: 3–5; velar: 6–8) and the total number of contacts in each zone is plotted as a function of time. It is useful to illustrate global differences in contact patterns between normal and pathological speech.

- *Centre of gravity* plot: useful to estimate the differences during different phases of articulatory gestures, e.g. the approach and release phase of stop consonants. The centre of gravity index was designed to locate the highest concentration of activated electrodes in a given EPG frame. Preferential weight is given to the anterior zone along the sagittal axis.

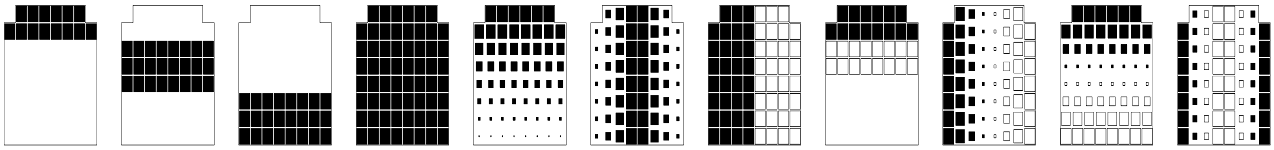
- *Frequency of contact* with each electrode over a period of time, expressed as a percentage and usually calculated in a row-by-row basis. For example, Farnetani et al. (1989) propose a coarticulation index that measures the difference in electrode activation for a phonetic segment (e.g. a consonant) when uttered in two different contexts (e.g. two different VCV sequences).

The majority of these techniques are based on linear combinations of the EPG vector components. We refer to such linear combinations as *linear indices* and represent them using a D-dimensional basis vector $\mathbf{v}$. The score of a D-dimensional EPG pattern $\mathbf{t}$ with respect to index $\mathbf{v}$ is thus the projection of $\mathbf{t}$ on $\mathbf{v}$, i.e., $\mathbf{v}^T\mathbf{t} = \sum_{d=1}^{D} v_d t_d$. Multiplying an index by a constant does not add any new information, so that $\mathbf{v}$, $2\mathbf{v}$ and $-2\mathbf{v}$ are all equivalent.

Many of these numerical indices have a fixed form, prescribed in advance. This form often reflects a phonetician's a priori beliefs about the relevance of different palate regions; for example, most work on EPG data reduction uses predetermined articulatory regions on the artificial palate (Byrd et al., 1995; Hardcastle et al., 1991b) via a linear combination of the individual contacts (see fig. 5.4). Alternatively, they may have been designed in some other way. For example, Nguyen (1995) applies the discrete cosine transform (DCT) to the EPG frames considered as images to obtain several linear indices, corresponding to low spatial frequencies. In particular, he proposes four indices (shown in fig. 5.4): SSU (Scaled SUm of activated electrodes), LRA (Left-Right Asymmetry index), APA (Alveolar-PAlatal index) and LME (Lateral-MEdian index), corresponding to spatial frequencies of 1, 2, 3 and 6, respectively. These indices are the most easily interpretable DCT basis vectors and also get the larger weights in the linear decomposition of a typical EPG frame. These indices are exactly or approximately equivalent to the traditional indices TOT, ASY, ALV and (scaled) LAT, respectively. Even though the DCT-based approach makes no a priori assumption about the data, it still has the disadvantage that the set of indices is fixed across data sets—because the indices are the basis vectors of the DCT, which for a given image size is fixed.

Therefore, ad hoc indices require the experimenter to make a priori assumptions (usually about the production of speech), thereby ignoring two kinds of variability: interspeaker variability (in palate size and shape[3], and in accent or other speaker-specific characteristics) and intraspeaker variability, e.g. different speech styles (colloquial, slow, etc.). Ad hoc methods are not robust and will not perform well in situations where the speech

[3]The Reading system is less vulnerable to this because the artificial palate is custom-fitted, with electrodes placed in anatomical landmarks of the individual.



ALV PAL VEL TOT = SSU COG LAT ASY FRIC LRA APA LME

Figure 5.4: A selection of typical EPG data reduction indices reported in the literature (basis vectors of an orthogonal, linear projection): alveolar (ALV), palatal (PAL), velar (VEL), total of contacts (TOT, equivalent to the scaled sum of activated electrodes, SSU), centre of gravity (COG), laterality (LAT), asymmetry (ASY), frication (FRIC), left-right asymmetry (LRA), alveolar-palatal (APA) and lateral-median (LME).

deviates from the standard: impaired speakers, different speech styles or unusual accents. This makes desirable the use of automatic methods to extract structure from a data set without requiring any prior knowledge about it, such as factor analysis (which looks for linear correlations), neural networks (which can find more complex relationships), etc. Such techniques could also be useful in determining relevant articulatory regions empirically; this would allow for interspeaker differences in the details of electrode placement with respect to anatomical configurations or speaker-specific articulatory patterns (Byrd et al., 1995), rather than using fixed electrode locations for all speakers, as is customary. Thus, one of the goals of this chapter is to find EPG cues in an empirical way, without referring to predetermined indices based on phonetic knowledge (such as those illustrated in fig. 5.4).

5.2.2 Adaptive data reduction

Adaptive dimensionality reduction for EPG data has been investigated using both linear systems and neural networks (Nguyen et al., 1996; Holst et al., 1995). Linear approaches have been based on a rotated principal components analysis (referred to as factor analysis by Nguyen et al., 1996). Two layer feed-forward neural network autoassociators used by Nguyen et al. (1996) and Holst et al. (1995) infer a set of basis vectors that spans a subspace similar[4] to that defined by the principal components (Bourlard and Kamp, 1988; Baldi and Hornik, 1989), as discussed in section 4.5. These adaptive methods lack a natural interpretation as a probability model (although one may be forced upon PCA, see section 2.6.2). In this thesis we are concerned with methods that explicitly define a full probability model.

5.2.3 Graphical representation of EPG indices

Graphically, we can represent both EPG patterns and indices as in figures 5.4 and 5.6: each small rectangle corresponds to one component of the D-dimensional vector (starting in the top left corner and ending in the bottom right one), where $D = 62$. The colour of the rectangle, black or white, indicates the positive or negative sign of the component, and the size of the rectangle is proportional to the magnitude of the component. The shape of the figure (an 8×8 grid where the top corners are unused) follows that of the palate, the alveolar part being in the top and the velar part in the bottom. In the conventional pictorial representation of EPG patterns, only binary data are allowed: each position is either empty or contains a full, black rectangle. Thus, we extend this representation to EPG indices by allowing any real value in each position. However, the reader should bear in mind that in an EPG pattern all components are either 0 or 1, and in an EPG index they can be any real number (positive or negative). Thus, fig. 5.4 depicts EPG indices while fig. 5.6 depicts EPG patterns.

Concerning phonemic categories, it should be clear that, for practical reasons, no general labelling of the data was performed, as explained in section 5.3. However, we use the labels of fig. 5.6 (identified perceptually by a phonetician) with the specific purpose of assessing our results in sections 5.4.2 to 5.5.

5.3 Data set description: the ACCOR database

For the experiments reported in this chapter we have used a subset[5] of the EUR-ACCOR database (described in appendix B). The EPG data consists of 62-bit frames sampled at 200 Hz. We have selected EPG frames

[4] If the neural network units have linear activation functions, then both spaces are exactly the same.

[5] We are grateful to Alan Wrench for providing us with the data.

corresponding to the English language, in which each of 6 different subjects (FG, HD, KM, PD, RK, SN; see table B.2) recorded 14 different utterances (table B.1). The data set was divided into 6 parts, one for each speaker, and the data of each speaker was split into a training and a test set: utterances 1, 3–5 and 9–14 were put into the training set and utterances 2 and 6–8 into the test set. The criterion followed was to have a balance between the number of different EPG frames (i.e., excluding repeated patterns) between the test and training set: roughly 80% for training and 20% for test (depending on each speaker).

All the data used were unlabelled. Labelling each EPG frame with the phoneme it corresponds to is a time-consuming process, requiring manual annotation of the EPG sequence in relation to the acoustic signal (and perhaps other signals) by a phonetician; automatic annotation is possible using a speech recognition program but prone to errors. This is another reason to perform unsupervised learning of the EPG data, so that articulatory categories can be extracted directly from the data in the form of factors, prototypes or, more generally, a latent-variable space.

5.4 Experimental results

We fitted the following continuous latent variable models to the EPG data: factor analysis, principal component analysis and GTM. All these models assume continuous variables, while the EPG data are binary. We discuss this issue in section 5.6.2. We also fitted a model for binary data, a mixture of multivariate Bernoulli distributions.

5.4.1 Scatterplots of principal components of the EPG data

Figure 5.5 shows projections of the data set corresponding to speaker RK on planes defined by two principal components. Observe the obvious departures from normality in many of them, including clustering and nonlinearity, which implies that the joint distribution of the EPG variables is not multivariate normal[6] and that the relation between some variables is not linear. This means that linear-normal models such as PCA or factor analysis will not be adequate.

5.4.2 Factor analysis and principal component analysis

For all the experiments we report, maximum likelihood factor analysis was performed using an EM algorithm (Rubin and Thayer, 1982) with random starting points. Once the relative increase in log-likelihood was smaller than 10^{-7}, the iterative procedure was stopped and varimax rotation was applied to the factors.

As mentioned in section 2.6.2, for a fixed data set we may regard PCA as an *additive* technique, insofar that a PCA of order L produces the same principal components as a PCA of order $L-1$ plus a new, additional principal component. This property of additivity does not hold in principle for either PCA followed by (varimax) rotation or factor analysis followed or not by (varimax) rotation. However, for the EPG data set we used, we found that this property of *factor additivity* holds with good approximation: after varimax rotation, the factor loadings obtained in a factor analysis of order $L-1$ appear again with little modification in a factor analysis of order L. This has an important consequence: the factors can be ordered by a *relevance* criterion, in the same way that the principal components can be ordered by the proportion of directional variance they explain. So, a factor analysis of order 1 produces the first factor; a factor analysis of order 2 finds that same factor plus a second one, and so on. In this situation, it is possible to refer individually to each factor as having an independent status.

Figure 5.7 shows the factor loadings obtained for a factor analysis of order 9, ordered by relevance for speaker RK. Considering the loading vectors as linear indices, it is apparent that many of the indices of figure 5.4 may be associated with the factor loadings or with linear combinations of these loadings (e.g. ALV, PAL and VEL with factors 3, 7 and 1, respectively).

This association may be regarded as an empirical justification of these phonetic indices, since they account for a large proportion of the correlation between variables, as well as possessing a straightforward interpretation. However, if we compare figure 5.7 with figure 5.8 (corresponding to speaker HD) we see that, although some factors are approximately equal for both speakers (e.g. factor 8 in RK is factor 2 in HD), there are significant differences, revealing different articulation trends. For example, speaker HD tends to produce EPG patterns which are much less symmetrical than those of RK. Such variations are difficult to track with fixed indices;

[6]A normally distributed projection may be hiding a non-normal data set, and besides Diaconis and Freedman (1984) prove that most projections from an arbitrarily distributed data set will be approximately normal. But, clearly, a projection which is not normally distributed necessarily implies that the original data set is also non-normal.

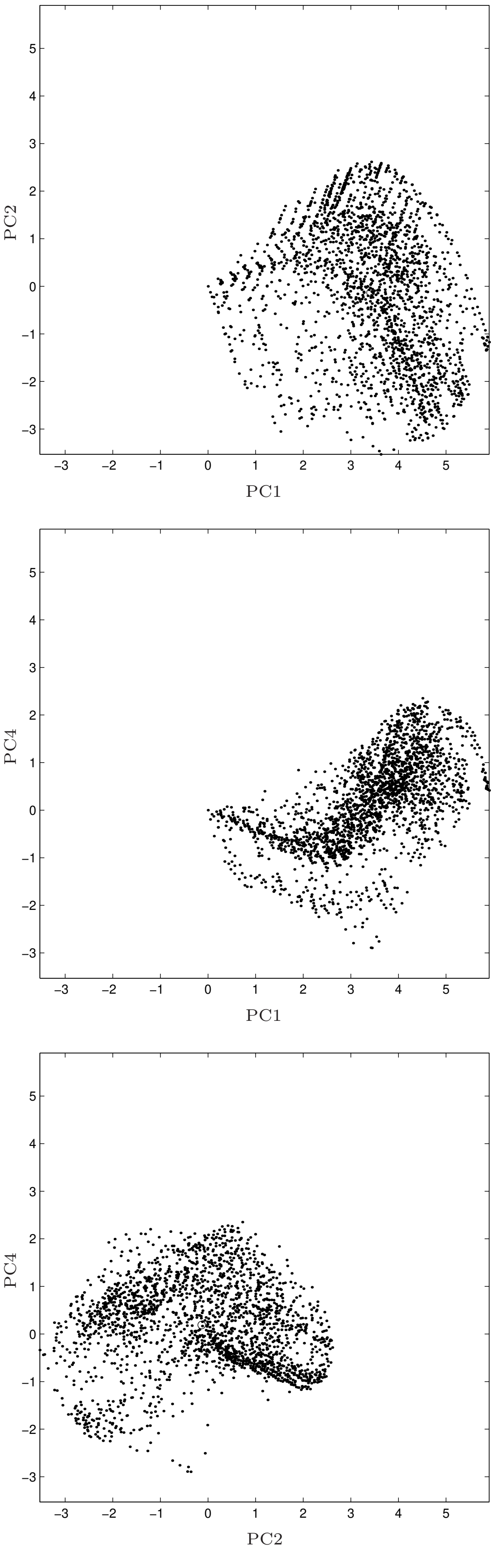


Figure 5.5: Projections of the EPG database corresponding to speaker RK on different principal component planes. The mean vector is marked as a circle (∘). The departure from normality is obvious.

e.g. the asymmetry index will give no extra information for a speaker that does not produce any asymmetric patterns.

We also applied principal component analysis (computed via singular value decomposition) to the same training sets as factor analysis. The first 9 principal components[7] for speaker RK are shown in figure 5.9. Most of the basis vectors contain both positive and negative values, thus lacking a simple interpretation. After a varimax rotation (figure 5.10) most basis vectors contain a majority of positive values, with several being similar to some of the factors of fig. 5.7. This is a consequence of the uniquenesses matrix $\mathbf{\Psi}$ being relatively isotropic (a multiple of the identity matrix), in which case factor analysis is equivalent to PCA. However, in a general situation the two methods give different results.

The factor analysis and PCA representations may be evaluated in terms of log-likelihood and reconstruction error. The left side of figures 5.11 and 5.12 show the log-likelihood of the factor analysis and PCA models for the training and test data sets of speaker RK and for different numbers of factors or principal components, respectively. In both models the log-likelihood value was computed as (see eqs. (2.20) and (2.29)):

$$\mathcal{L}(\mathbf{\Theta}) = -\frac{N}{2}\left(D \ln 2\pi + \ln|\mathbf{\Sigma}(\mathbf{\Theta})| + \operatorname{tr}\left(\mathbf{S}\mathbf{\Sigma}(\mathbf{\Theta})^{-1}\right)\right) \tag{5.1}$$

where $\mathbf{S} = \frac{1}{N}\sum_{n=1}^{N}(\mathbf{t}_n - \bar{\mathbf{t}})(\mathbf{t}_n - \bar{\mathbf{t}})^T$ is the sample covariance matrix of the data set, with $\bar{\mathbf{t}} = \frac{1}{N}\sum_{n=1}^{N}\mathbf{t}_n$ being the sample mean. $\mathbf{\Sigma}(\mathbf{\Theta})$ is the covariance matrix under the model specified with parameters $\mathbf{\Theta}$, i.e., $\mathbf{\Sigma}(\mathbf{\Lambda}, \mathbf{\Psi}, \boldsymbol{\mu}) = \mathbf{\Lambda}\mathbf{\Lambda}^T + \mathbf{\Psi}$ for factor analysis and $\mathbf{\Sigma}(\mathbf{\Lambda}, \sigma^2, \boldsymbol{\mu}) = \mathbf{\Lambda}\mathbf{\Lambda}^T + \sigma^2\mathbf{I}$ for PCA, according to the probabilistic PCA model of section 2.6.2. Factor analysis is systematically better than PCA, as the theory predicts, because PCA is a particular case of factor analysis in which the uniquenesses are forced to be equal. That is, factor analysis is a more complete model than PCA. Not shown in the figure is the fact that the maximum achievable log-likelihood is identical for PCA (using the limit of 62 principal components) and factor analysis (using 51 factors, the identifiability limit, see section 2.8.2.1).

The right side of figures 5.11 and 5.12 shows the reconstruction error of the factor analysis and PCA models, respectively. For PCA, the projection on the L principal components was used[8]; for factor analysis, the projection using the posterior mean (2.19) was used. The reconstruction error was defined in the usual way:

$$E_2 = \frac{1}{N}\sum_{n=1}^{N}\|\mathbf{t}_n - (\mathbf{\Lambda}\mathbf{A}(\mathbf{t} - \boldsymbol{\mu}) + \boldsymbol{\mu})\|_2^2 \tag{5.2}$$

where for PCA, $\mathbf{\Lambda} = \mathbf{A}^T$ contains the first L principal components and for factor analysis, $\mathbf{\Lambda}$ contains the factor loadings and $\mathbf{A}$ is defined in (2.18). According to the squared reconstruction error criterion PCA is systematically better than factor analysis; note that PCA is optimal among linear mappings according to this criterion on the training set. Further, the directional variance (along each component or factor) decreases monotonically in PCA, while in factor analysis it does not—although it has a tendency to decrease with the relevance order.

5.4.3 GTM

A factor analysis goodness-of-fit test[9] (valid in the asymptotic limit of large samples) is available for the null hypothesis that "the data sample can be explained with L factors" (Everitt, 1984), i.e., that the covariance of the observed variables can be accounted for with L factors. For our training sets (of sizes well above $N = 5\,000$ in all cases) and at a significance level of 5%, this hypothesis was rejected for all $L < 45$. A similar test is

[7] Defined by the ordered eigenvectors $\mathbf{u}_1, \dots, \mathbf{u}_L$ of the sample covariance matrix rather than by the columns of matrix $\mathbf{\Lambda}$ in eq. (2.28), although for interpretation purposes the scaling introduced in the columns of $\mathbf{\Lambda}$ is irrelevant.

[8] Again, by principal components we mean the ordered eigenvectors of the sample covariance matrix rather than the columns of matrix $\mathbf{\Lambda}$ in eq. (2.28). Thus, we disregard the dimensionality reduction mapping induced by the probability model and use the usual PCA orthogonal projection, which is guaranteed to minimise the reconstruction error of any linear dimensionality reduction mapping (section 4.5).

[9] This statistical test should be considered only as qualitative evidence for two reasons:

- Traditional hypothesis tests are not generally valid under a Bayesian perspective, due to the fact that they use the likelihood and not the posterior distribution. That is, they disregard the prior distribution of a given sample—which may be very difficult to estimate anyway. This means that the conventional practice of quoting P-values for assessing the level of rejection of a hypothesis is actually of qualitative value only. Berger (1985, pages 145–157) and Sivia (1996, pages 87–88) further discuss this issue.
- This test requires normality of the population, as most parametric tests do (Kanji, 1993, page 3), which does not hold for our EPG data sets (section 5.4.1).

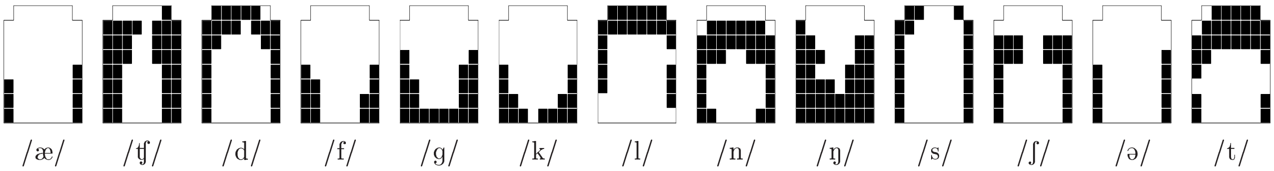


Figure 5.6: Representative EPGs for the typical stable phase of different phonemes.

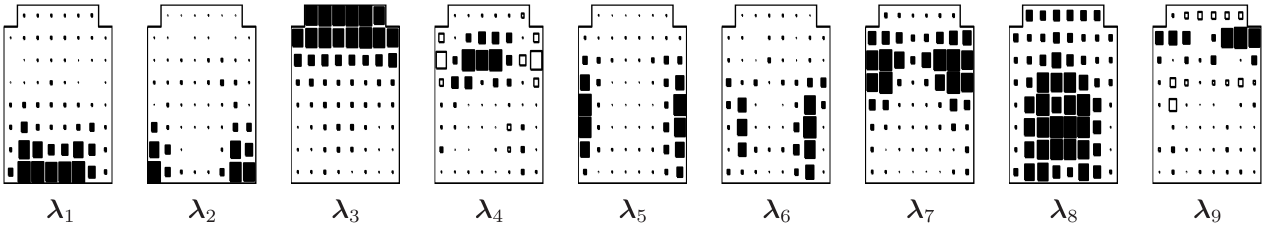


Figure 5.7: Factors for speaker RK after varimax rotation.

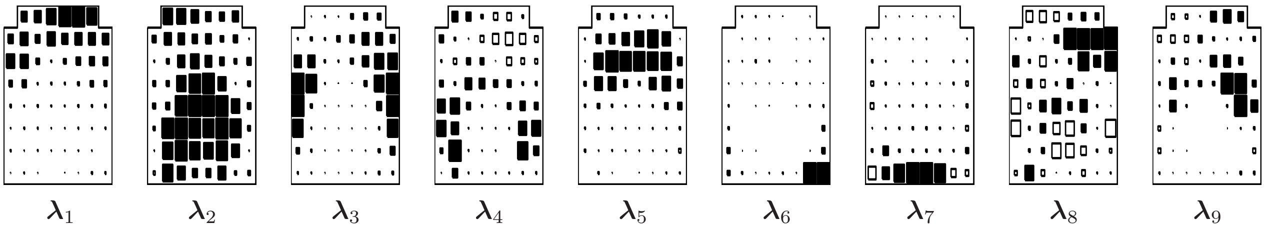


Figure 5.8: Factors for speaker HD after varimax rotation.

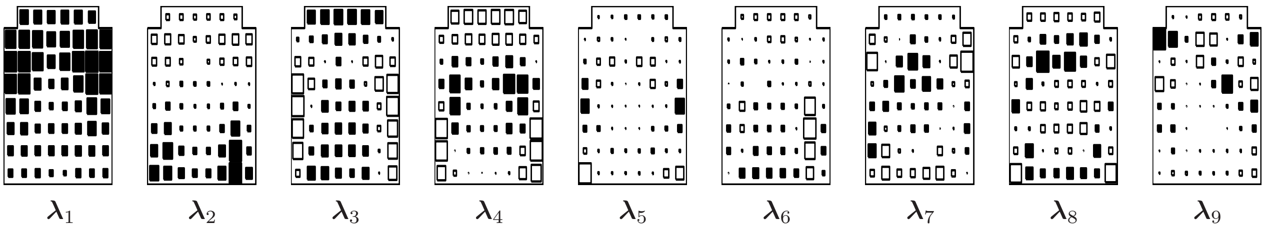


Figure 5.9: Principal components for speaker RK.

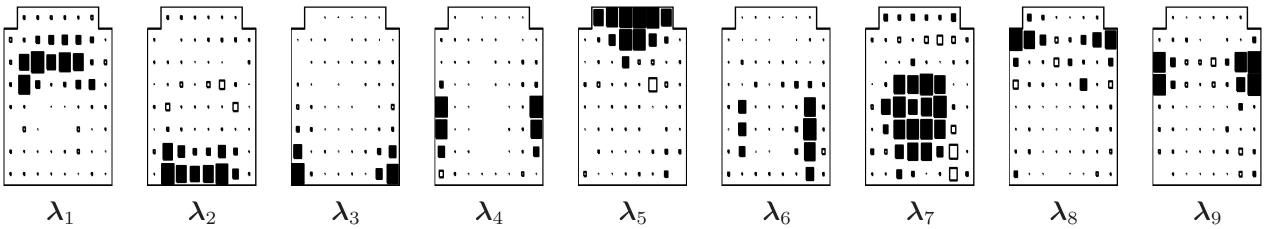


Figure 5.10: Principal components for speaker RK after varimax rotation.

available for PCA and the null hypothesis was rejected for all values of L at the same significance level. This suggests that linear methods are not powerful enough and are able to extract only a fraction of the information contained in the EPG data. Besides, the scatterplots of section 5.4.1 indicate that linear-normal methods will not be adequate.

The generative topographic mapping (GTM) is a latent variable model well suited to visualisation, since the latent space is effectively limited to two dimensions[10]. We trained GTM models[11] with the following parameters: 20×20 grid in two-dimensional latent space ($K = 400$ points), scaled to the $[-1, 1] \times [-1, 1]$ square, and $\sqrt{F} \times \sqrt{F}$ grid in the same square of F Gaussian basis functions of width equal to the separation between basis functions centres; $\sqrt{F}$ varied from 3 to 14. Training was achieved by an EM algorithm, stopping when the relative increment in log-likelihood was smaller than 10^{-4}; the starting point was not random, but derived from the first principal components of the data set, as described by Bishop et al. (1998b).

Figure 5.13 gives the log-likelihood (left) and the reconstruction error (right) for the GTM models (speaker RK; the vertical axes have the same scale as in figs. 5.11–5.12). We can see that:

- While the log-likelihood for the training set increases monotonically with increasing F, that of the test set reaches a maximum at around $F = 49$ and starts decreasing again. This means that overfitting is occurring for $F > 49$: the model is learning the training set too well but cannot generalise to the test set properly. The phenomenon of overfitting did not appear in the previous models, probably due to the small number of parameters (although the log-likelihood curves for the test set do flatten if many factors or principal components are used). Note that the reconstruction error continues decreasing monotonically past $F = 49$.

- The log-likelihood of the GTM model (even with a relatively small number of basis functions, to avoid overfitting) is better than the highest possibly attainable log-likelihood of both PCA and factor analysis (i.e., using any number of principal components or factors). The reconstruction error using as reduced-dimension representative the mode of the posterior distribution[12] is very small, comparable to that of a PCA with 10 principal components or more (i.e., a latent space of dimension $L = 10$).

We remark that GTM is using a latent space of only dimension 2, while the curves for factor analysis and PCA correspond to latent spaces of higher dimension.

GTM is then a much better generative model for the EPG data set than PCA or factor analysis, due to the fact that the mapping between latent and data space used by GTM is an expansion in radial basis functions, which have been proven to be able to approximate any smooth function to arbitrary accuracy given enough basis functions (Park and Sandberg, 1993; Scarselli and Tsoi, 1998). However, the number of parameters of a GTM model is typically much higher than that of the other models we have analysed.

As we mentioned in section 2.9, the reduced-dimension representative for a data point $\mathbf{t}$ is obtained as a particularly informative point in latent space according to the posterior distribution for $\mathbf{t}$. For factor analysis and PCA this posterior distribution is normal (more or less broad, depending on the covariance matrix) and there is no doubt as to what reduced-dimension representative to select, because the normal is unimodal: the mean (equal to the mode). This is not necessarily the case for GTM, though: in principle, the posterior distribution for a given data point can be multimodal, which makes difficult to select a single reduced-dimension representative (see section 2.9). However, after our GTM model was trained, we found that the posterior distribution was unimodal and sharply peaked for over 90% of the data points. Thus, almost every EPG pattern can be associated with a unique point in two-dimensional latent space. Clearly, the coordinates of the reduced-dimension representative could be considered as (nonlinear) EPG data reduction indices.

5.4.4 Mixture models

5.4.4.1 Mixtures of factor analysers

We modelled the same training sets with a mixture of factor analysers. Again, a maximum likelihood estimate for the parameters of the model was found via an EM algorithm[13] (Ghahramani and Hinton, 1996), with random starting point and stopping when the relative increment in log-likelihood was smaller than 10^{-4}. The

[10]Although theoretically it may use a latent space of any dimension L, in practice one uses $L \leq 2$, since the computational complexity is approximately exponential in L due to the curse of the dimensionality (section 2.6.5).

[11]We are grateful to Markus Svensén's for making available (from `http://www.ncrg.aston.ac.uk/GTM`) his Matlab implementation of GTM.

[12]Using as reduced-dimension representative the mean of the posterior distribution gave a slightly larger error in all cases.

[13]We are grateful to Zoubin Ghahramani for making available (from `http://www.gatsby.ucl.ac.uk/~zoubin`) his Matlab implementation of mixtures of factor analysers.

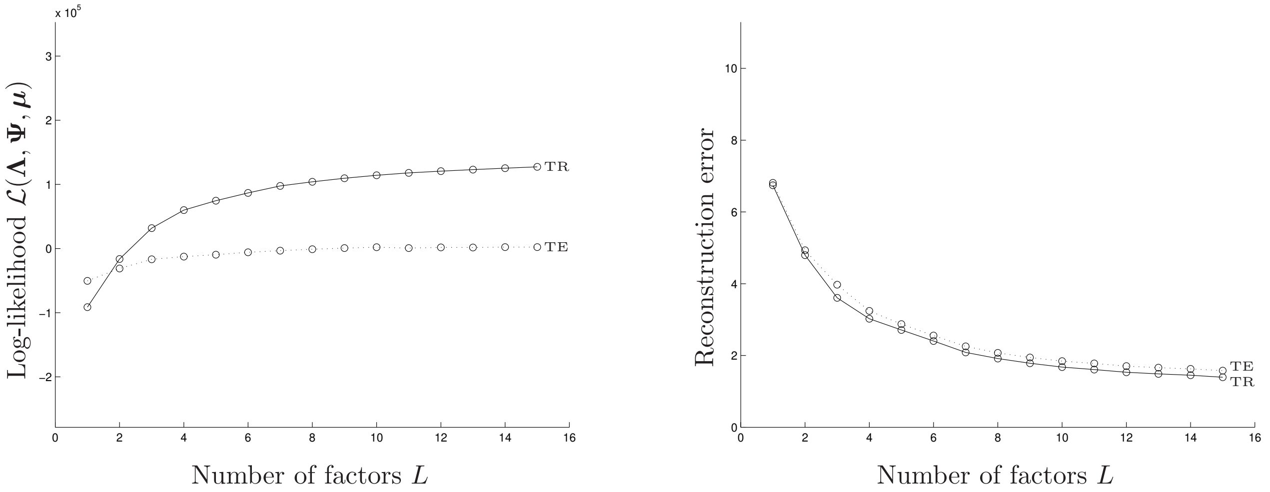


Figure 5.11: Log-likelihood (left) and reconstruction error (right) of the factor analysis model for speaker RK. TR and TE refer to training and test set, respectively.

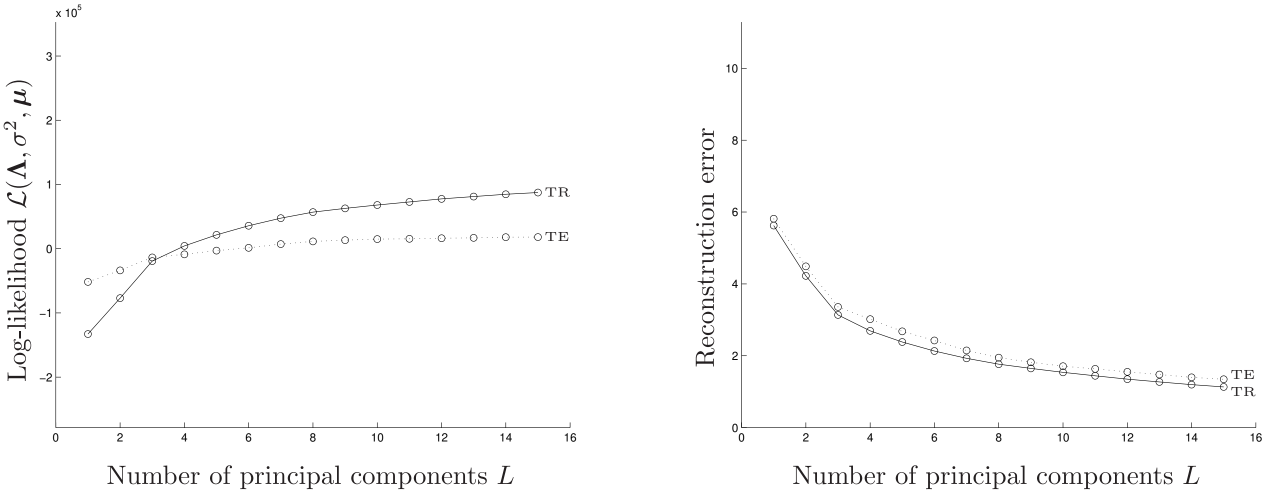


Figure 5.12: Log-likelihood (left) and reconstruction error (right) of the PCA model for speaker RK. TR and TE refer to training and test set, respectively. Vertical scale as in fig. 5.11.

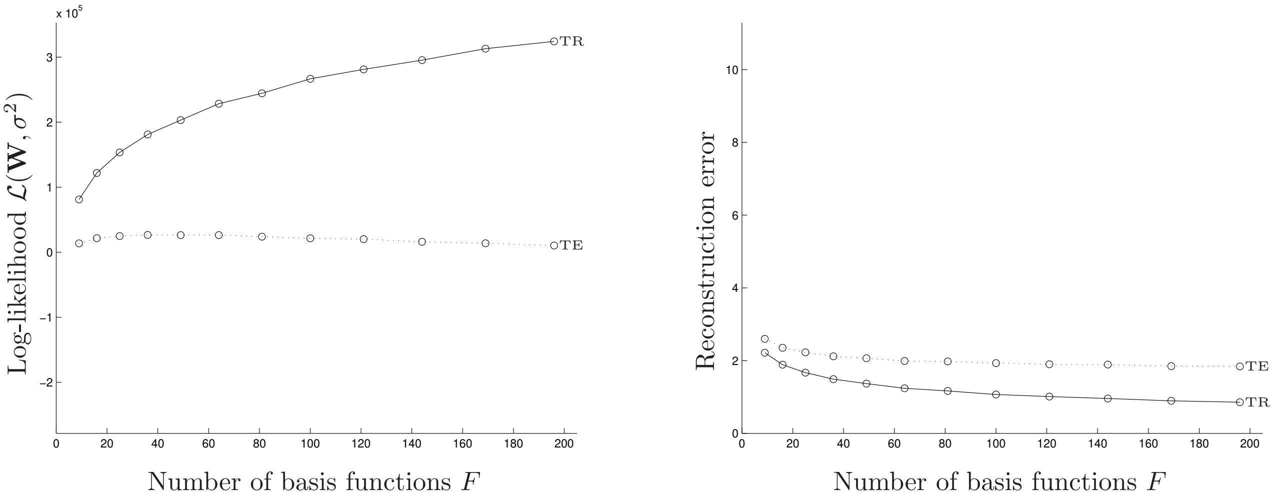


Figure 5.13: Log-likelihood (left) and reconstruction error (right) of the GTM model for speaker RK. TR and TE refer to training and test set, respectively. Vertical scale as in fig. 5.11.

parameters are now the mixing proportions π_m of the mixture (that correspond to the prior probabilities of the different mixture components) and the means $\boldsymbol{\mu}_m$ and factor loadings $\boldsymbol{\Lambda}_m$ of each factor analyser.

Figure 5.14 shows the results for a mixture with 4 components, each of them a factor analysis of first order. The means coincide with some of the typical EPG patterns of fig. 5.6 (and with some of the prototypes found by the mixture of multivariate Bernoulli distributions of the next section). The corresponding loading vectors coincide with or are approximately a linear combination of several of the factor loadings of fig. 5.7. This implies that the mixture does not find new factors, but that it places different factors in different locations of data space, adapting to the local behaviour of the correlations. For example, component number 2 is located in the area of vowels like /æ/ and is associated with a factor stressing velar patterns with some left-right asymmetry.

In all our experiments, a first-order factor analysis was used for each component (which made varimax rotation unnecessary), varying only the number of components in the mixture. The reason for not using higher order factor analysers is that the estimation algorithm systematically tends to a singularity of the likelihood surface, where some of the uniquenesses tend to zero and the $\boldsymbol{\Psi}$ matrix is not positive definite. This is a well-known problem in the literature of factor analysis and finite mixture distributions (Bartholomew, 1987; Everitt and Hand, 1981), called a *Heywood case*. See section 5.6.1 for possible solutions to this.

As before, the left side of figure 5.15 shows the log-likelihood of the mixture of factor analysers model for all data sets of speaker RK, and the right side the reconstruction error (M is the number of components in the mixture). The log-likelihood of the mixture is always superior to that of simple factor analysis (the latter being a more restricted maximum likelihood model), and the reconstruction error is always smaller as well, although there is not much difference. The mixtures considered do not result in a significantly improved performance over factor analysis.

5.4.4.2 Mixtures of multivariate Bernoulli distributions

We trained a mixture of multivariate Bernoulli distributions with an EM algorithm (Everitt and Hand, 1981; Wolfe, 1970), using random starting values for the parameters and stopping when the relative increment in log-likelihood was smaller than 10^{-4}. In this case, the only parameters to be estimated (apart from the mixture proportions) are the Bernoulli probabilities $\mathbf{p}_m = (p_{m1}, \dots, p_{mD})^T$ of each component, which can be considered as the prototype vectors for that component, because the mean of a Bernoulli distribution coincides with its $\mathbf{p}$ value. Since each Bernoulli probability (of the corresponding electrode being activated) is in the range $[0, 1]$, the prototypes obtained are interpretable as EPGs without the necessity of any transformation, such as varimax rotation.

Again, we observed that for this data set the mixtures also have approximately the property of additivity mentioned in section 2.6.2, and so we can order the prototypes obtained. A mixture with $M = 9$ components produced the prototypes shown in fig. 5.16, in decreasing order of relevance from left to right. Those prototypes are easily recognisable as some of the typical EPG patterns of fig. 5.6, e.g. $\mathbf{p}_2$ with /ʃ/, $\mathbf{p}_5$ with /l/ or $\mathbf{p}_6$ with /g/. Thus, the mixture may be doing a good job at estimating the density of the distribution of the EPG data in $D = 62$ dimensions, but—as noted in section 2.9.3.3—it does not achieve dimensionality reduction. The most it can do is to assign each data vector to its most likely component (in the sense of highest posterior probability) and reconstruct it as the prototype ($\mathbf{p}_m$ vector) of that component. This is essentially vector quantisation.

Figure 5.17 (left) shows the log-likelihood for speaker RK (where M is the number of components used). The addition of new components after the first 4 or 5 does not increase much the log-likelihood. Figure 5.17 (right) shows the reconstruction error, which is quite large compared to most of the other methods, confirming that this model is not good for vector reconstruction.

Also, the phenomenon of overfitting appears here again: the log-likelihood for the test set reaches a plateau at about $M = 4$.

As noted in section 3.2.2, EM estimation of a mixture of multivariate Bernoulli distributions will always give a positive likelihood from almost every starting point. In particular, boundary values (where some p_{md} can be 0 or 1) will not give rise to a likelihood equal to 0 (and a log-likelihood equal to $-\infty$) for points in the training set. However, this is not necessarily the case for other data sets. This is an undesirable feature arising from the Bernoulli distribution itself: a Bernoulli distribution of parameter $p = 0$ (1) can never generate a 1 (0). The other models studied (based on the normal distribution) assign to any point in the domain a strictly positive probability, however small this may be. Thus, for each speaker, a few points from the test set (usually less than 2%) were removed in a number of occasions in order to obtain a finite log-likelihood in figure 5.17.

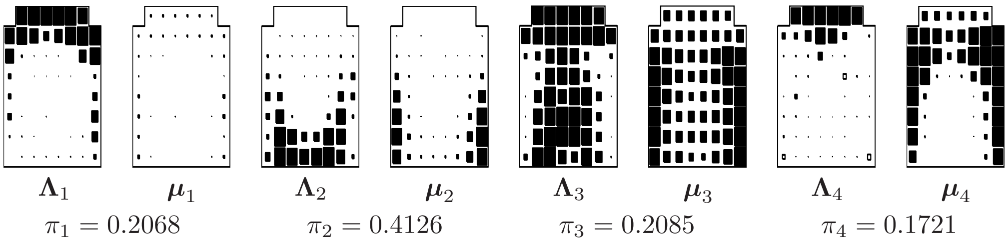


Figure 5.14: Means $\boldsymbol{\mu}_m$ and factor loadings $\mathbf{\Lambda}_m$ for a mixture of $M = 4$ factor analysers, each of $L = 1$ factor (for speaker RK). Below each pair $(\boldsymbol{\mu}_m, \mathbf{\Lambda}_m)$ is the mixing proportion π_m of component m.

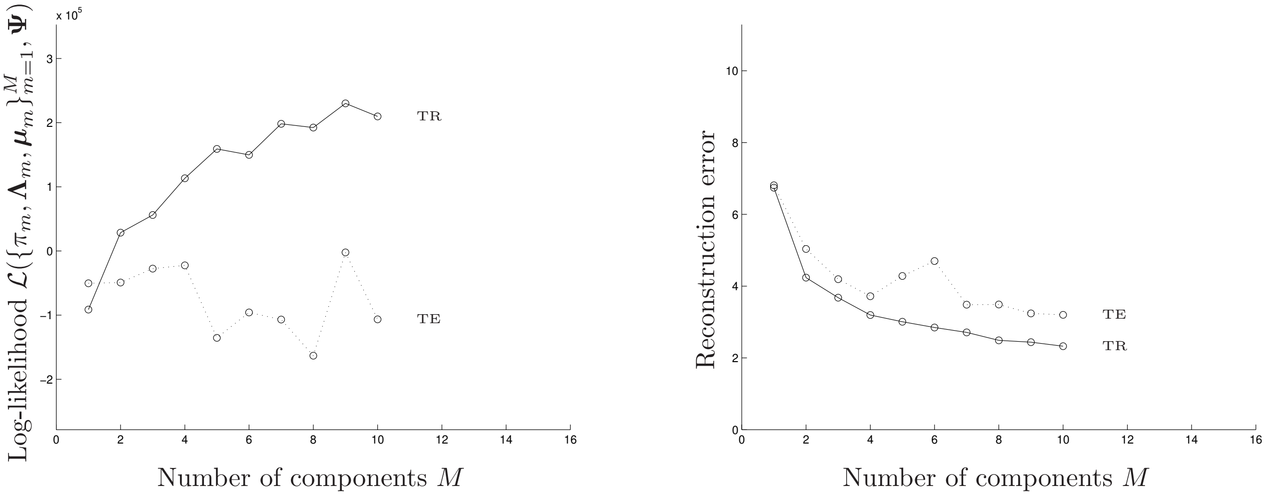


Figure 5.15: Log-likelihood (left) and reconstruction error (right) of the mixture of factor analysers model for speaker RK. TR and TE refer to training and test set, respectively. Vertical scale as in fig. 5.11.

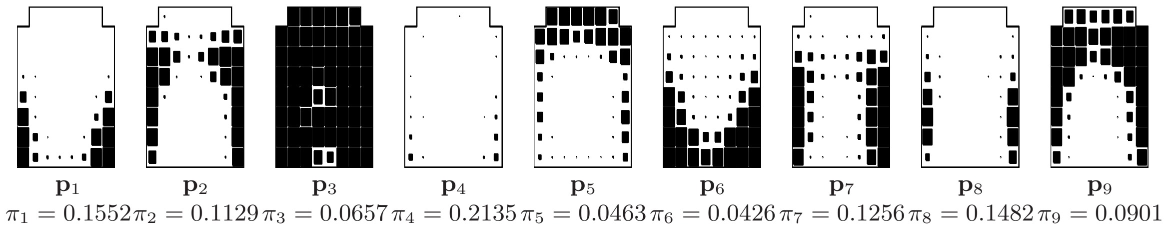


Figure 5.16: Prototypes for a mixture of multivariate Bernoulli distributions for speaker RK.

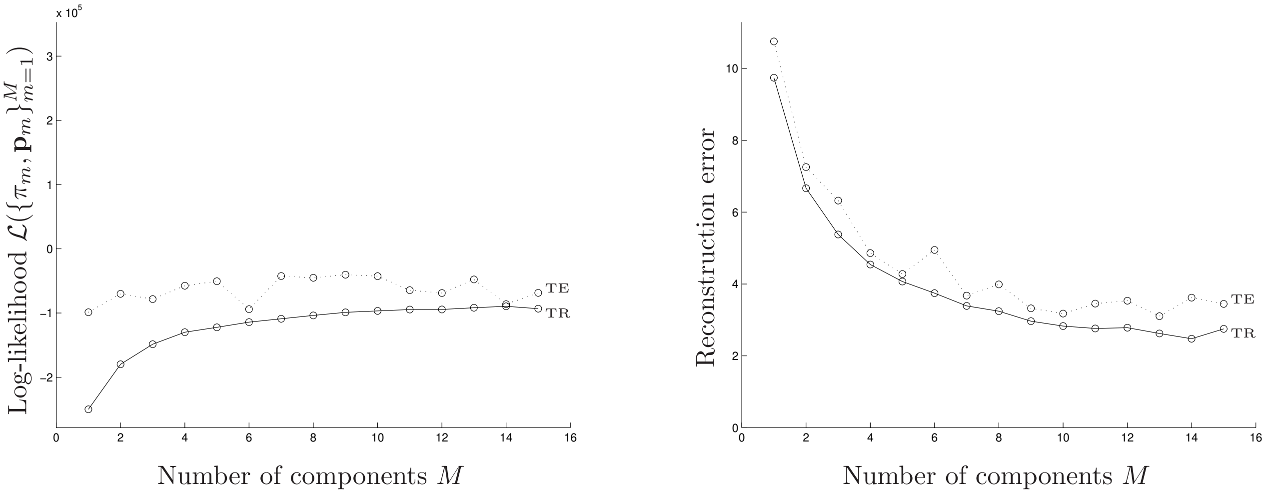


Figure 5.17: Log-likelihood (left) and reconstruction error (right) of the Bernoulli mixture model for speaker RK. TR and TE refer to training and test set, respectively. Vertical scale as in fig. 5.11.

5.5 Two-dimensional visualisation of EPGs

We consider now the issue of graphically representing a set of EPG frames, possibly a sequence obtained during an utterance. When there are more than two indices, the usual way to do this is by plotting the variation of all the indices with the time in an X-Y plot (called *trajectory* or *contact profile*) where time goes in abscissas and the index or indices of interest in ordinates (Byrd et al., 1995; Hardcastle et al., 1991b). However, when there are only two indices a more natural representation is possible: both index values are used as the X-Y coordinates in a plane, with points consecutive in time being linked by an arrow. If the two indices are powerful enough to adequately discriminate between different articulatory classes, the two-dimensional representation will be a *map* of articulatory regions.

There are statistical methods specifically suited to find the best projection of a multidimensional data set in the sense that it maximises a certain criterion. For example, discrimination between classes (Fisher's linear discriminant), departure from normality (projection pursuit) or a stress measure (multidimensional scaling); see chapter 4. These methods do not propose a model for the high-dimensional data. In contrast, our latent variable models are trained not to find an optimal low-dimensional projection but to model best the data according to the maximum likelihood criterion. Nevertheless, the two-dimensional representation found by them (in particular by GTM) turns out to be good[14], as one might intuitively expect.

Figure 5.18 shows such a representation using pairs of factors for speaker RK. In the left picture, factors 1 and 2 were used, while in the right one factors 3 and 4 were used. The labels correspond to the typical EPG patterns of figure 5.6, while the trajectory corresponds to the highlighted fragment of the following utterance: "I prefer **Kant to** Hobbes for a good bedtime book," with phonemic transcription /aɪ prɪ'fɜ 'kænt tə 'hɒbz fər ə 'gʊd 'bedtaɪm 'bʊk/. In each picture, the trajectory was obtained by projecting onto two-dimensional latent space each of the EPG frames of the utterance fragment and linking consecutive ones by a straight line. For visualisation purposes, we are free to choose the best pair from the factors extracted. In fact, factors 3-4 give a poor view of the data while factors 1-2 give a better one.

Figure 5.19 shows, for the case of two-dimensional GTM with $F = 49$ basis functions, the two-dimensional latent space with the same typical EPG patterns and utterance fragment as in fig. 5.18. Now, points in two-dimensional latent space which correspond to consecutive EPG frames in the utterance have been labelled with a correlative number to show more clearly the path followed by the reduced-dimension representative. The GTM map presents a much clearer layout of the articulatory classes than the best pair of factors did.

These two-dimensional maps obtained by latent variable models can be useful to analyse articulatory dynamics. Since the inverse mapping $\mathbf{F}$ defined in section 2.9 is continuous for factor analysis and for GTM (although see below), and since the EPG signal is assumed to be a continuous function of the time (due to the mechanical constraints of the tongue-palate system), discontinuities in the latent space trajectory must be due to abrupt jumps in the sampled EPG sequence in data space. This is the case for the transition from /æ/ to /nt/ in the utterance mentioned (fig. 5.20). The reason is that the EPG sampling frequency (equal to 200 Hz), while generally appropriate, is too low in situations where the tongue moves very fast and this results in missing EPG frames. The latent variable two-dimensional maps are able to detect this situation. Note, however, that the concept of continuity is not well defined because:

- The EPG frames are binary. Still, for a high enough sampling frequency, one can expect most neighbouring frames to be very close (in both the Euclidean and Hamming distance sense).
- The latent space in GTM is discretised (see sections 2.6.5 and 2.9.2). However, owing to the fact that for most data points the posterior distribution was unimodal and sharply peaked, examination of the EPG sequence of the utterance showed that nearby points in data space were most times projected on neighbouring points in latent space.

Two-dimensional maps are also useful in speech therapy, speech assessment or language learning applications. For example, in speech therapy, most systems (such as the IBM SpeechViewer; Pratt et al., 1993), provide real-time visual and auditory feedback on a range of speech parameters. Auditory feedback is available as normal or reduced-speed playback. Visual feedback includes cartoon graphics, intended for the speaker, and technical displays, useful for the therapist to monitor the details of the speaker's performance or to show aspects of speech patterning. The latter include displays over time of the fundamental frequency or the amplitude, 3D spectrograms, etc. If articulatory regions are drawn on the two-dimensional latent space, the speaker will receive instantaneous feedback in an intuitive way about the correctness of the articulatory gesture that

[14] We realise that our comparisons here are of qualitative value only. However, making them quantitative would take us too far away, as it would require defining an index of *goodness of two-dimensional representation* and comparing our latent variable models with some of the other methods. A further obstacle is that our data are unlabelled.

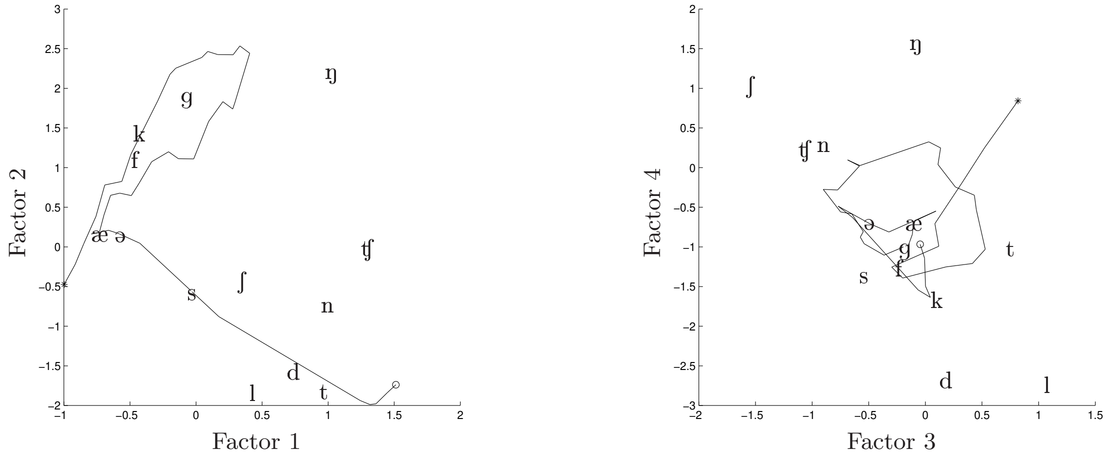


Figure 5.18: Two-dimensional plot of the trajectory of the utterance fragment "I prefer **Kant to** Hobbes for a good bedtime book" using factor analysis for speaker RK (left: latent space of factors 1 and 2; right: latent space of factors 3 and 4). The start and end points are marked as $*$ and $\circ$, respectively. The phonemes are located on the figure by projecting the prototypes of figure 5.6 using equation (2.19).

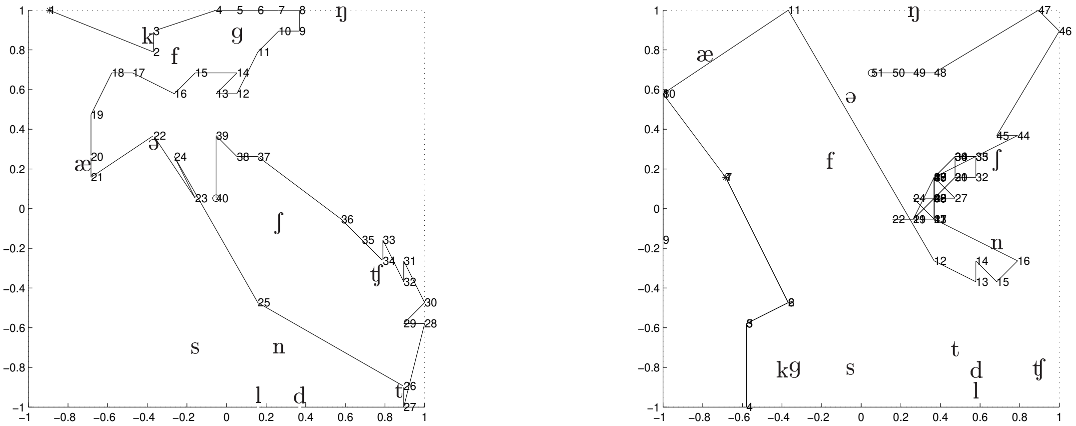


Figure 5.19: Two-dimensional plot of the trajectory of the utterance fragment "I prefer **Kant to** Hobbes for a good bedtime book" using GTM with $F = 49$ basis functions and a 20×20 latent space grid (left: speaker RK; right: speaker HD). The start and end points are marked as $*$ and $\circ$, respectively. Points are numbered correlatively. The phonemes are located on the figure by projecting the indices of figure 5.6 using the posterior mean of the GTM model.

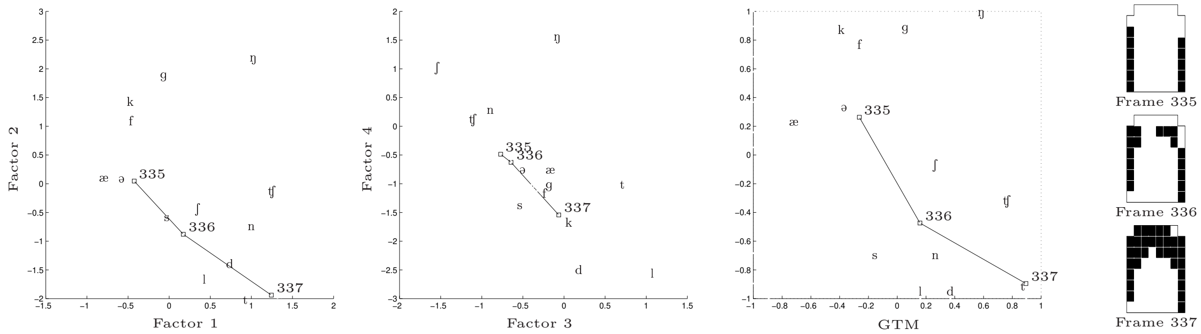


Figure 5.20: Discontinuities in the latent space due to discontinuities in the EPG sequence of the utterance fragment "I prefer **Kant to** Hobbes for a good bedtime book." For speaker RK, the transition from /æ/ to /nt/ occurs from frame 335 to frame 336. The right part of the figure shows the three consecutive EPG frames. The plots on the left part show the corresponding trajectory in the two-dimensional map of factor analysis (factors 1 vs. 2, left; factors 3 vs. 4, centre) and GTM (right).

he is trying to produce, a difficult task to achieve with the raw data. Hardcastle et al. (1991a) give a specific account of the uses of electropalatography (using both raw and reduced data) in speech therapy.

5.6 Discussion

5.6.1 Method comparison

Figures 5.21 and 5.22 show the log-likelihood and reconstruction error, respectively, for the training and test sets of speaker RK and all the models studied. The curves for other speakers were also very similar, with the ranking of the methods being virtually the same for all our six speakers[15].

Our primary criterion is the log-likelihood in the test set, since it measures the generalisation power of the generative model (although in a Bayesian approach a prior distribution over the parameters and the test set should be considered too). Reconstruction error is of secondary importance from a modelling perspective, since it treats equally the true data and the noise—although a good model in terms of likelihood will usually have a low reconstruction error as well.

For the cases studied, overfitting in the log-likelihood can appear if the number of parameters of the model is large enough (e.g. GTM in the right side of fig. 5.21), but the reconstruction error presents a steady decrease in both the training and test sets for any number of parameters (fig. 5.22).

Linear models (factor analysis and PCA) The training set curves confirm the theory: factor analysis outperforms PCA in log-likelihood (PCA being a particular case of factor analysis) while PCA outperforms factor analysis in reconstruction error (since PCA is the linear transformation with minimal squared reconstruction error). Interestingly, PCA outperforms factor analysis in log-likelihood in the test set for speakers RK and HD. We can conclude that for our data set the PCA model is almost as good as the factor analysis one. Both methods are computationally straightforward.

Mixtures of latent variable models have the advantage of providing different latent variable models for different data space regions, thus adapting to local structure of the data. This often is a good modelling assumption, in that data are usually due to a number of different, concurrent processes. The mixtures of factor analysers, using a latent space of only one dimension, have a performance similar to that of factor analysis with several factors. This makes them promising for higher dimensions of the latent space. However, their practical application is limited as long as a training algorithm which efficiently avoids singularities in the log-likelihood surface is not available. One could perhaps overcome this by constraining the domain of some parameters, although the question arises of how much to constrain them. More generally, one could use regularisation techniques (Girosi et al., 1995) in the form of a prior on the parameters. This requires some sort of approximation due to the impossibility of solving exactly the marginalisation of the hyperparameters. Recent work on Bayesian regularisation for mixtures of factor analysers includes the use of variational approximations (Ghahramani and Beal, 2000) and of Gibbs sampling (Utsugi and Kumagai, 2001). Not all mixtures are affected by this problem, either because of the nature of the likelihood surface (e.g. mixtures of multivariate Bernoulli distributions, section 3.2.2) or because a direct algorithm to find maxima of it is available (e.g. mixtures of PCAs, section 2.6.2).

Mixtures of multivariate Bernoulli distributions fare worse than all the other methods in all categories despite being the only model purely for binary data. Computationally they are also more costly than linear models. Their use for EPG data modelling is thus discouraged.

The generative topographic mapping (GTM), a nonlinear latent variable model, appears to be the best model in all categories. Taking the point of $F = 49$ radial basis functions (which gives the maximum likelihood in the test set), it clearly outperforms all the other models in log-likelihood and also reconstruction error (PCA needs a latent space of more than $L = 10$ dimensions to achieve a comparable reconstruction error). GTM achieves this with a latent space of only $L = 2$ dimensions, while the other methods were tested up to $L = 15$ (except the mixtures of factor analysers). The major drawback of GTM is its computational complexity, exponential in the dimension of the latent space, which limits it to $L = 2$ dimensions in practice.

[15] A Friedman test (for n-wise comparisons) (Kanji, 1993, test 73, p. 113) showed significant n-wise differences (at a significance level of 1%) between the effect of the methods. Although this test does not require normality of the population (being a nonparametric test), again this evidence should be only qualitative under the Bayesian viewpoint (see comments in footnote 9, section 5.4.3).

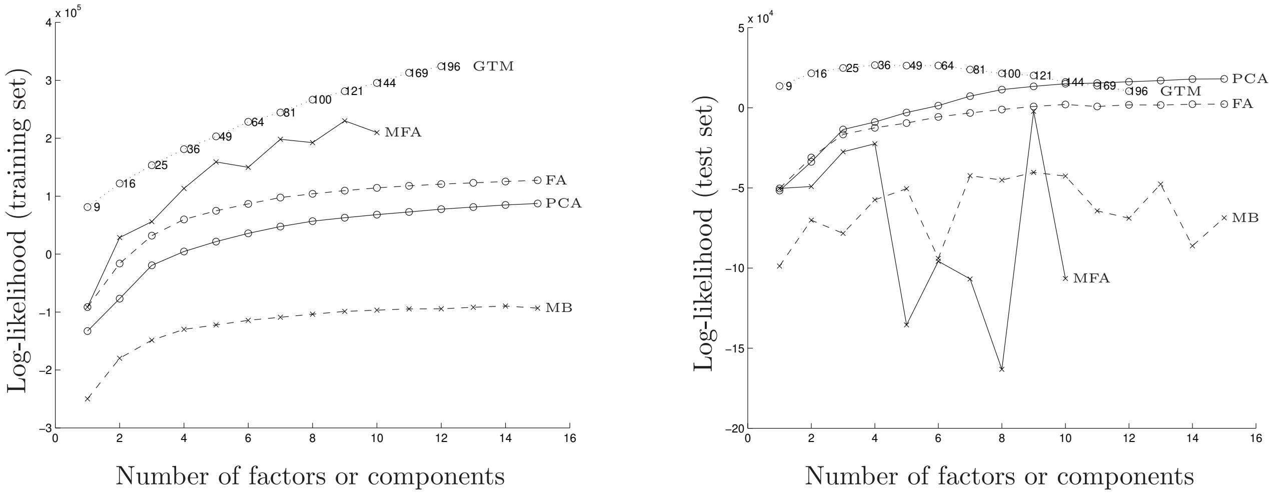


Figure 5.21: Comparison between methods in terms of log-likelihood for speaker RK (left: training set; right: test set): factor analysis (FA), principal component analysis (PCA), generative topographic mapping (GTM), mixtures of factor analysers (MFA) and mixtures of multivariate Bernoulli distributions (MB). The x axis refers to the order of the factor analysis or principal component analysis, the number of mixture components in the case of mixture models and the square root of the number of basis functions in the case of GTM. Compare with the left graph of figures 5.11–5.13, 5.15 and 5.17.

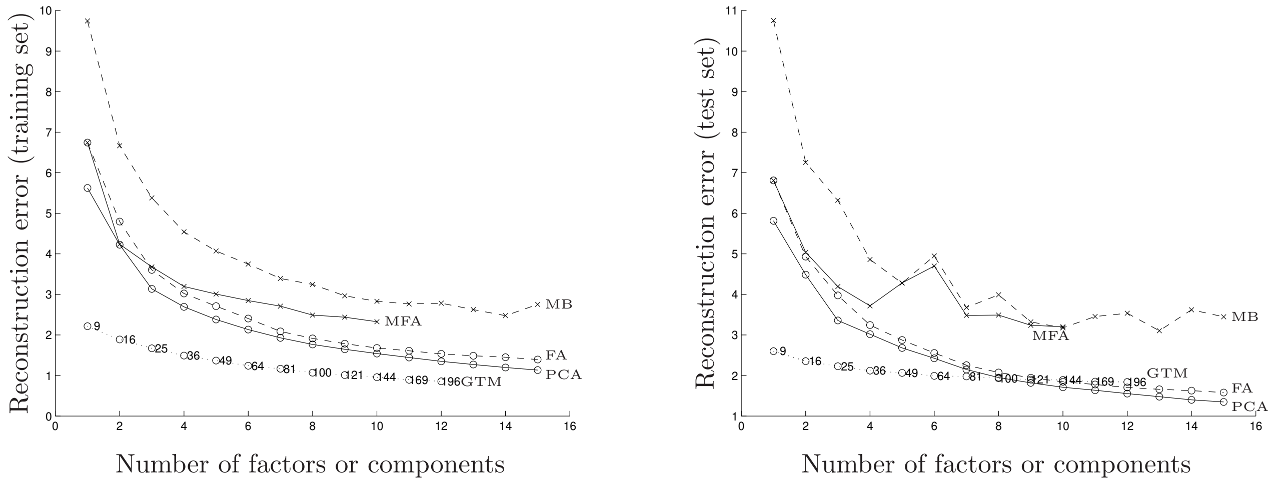


Figure 5.22: Comparison between methods in terms of reconstruction error for speaker RK (left: training set; right: test set): factor analysis (FA), principal component analysis (PCA), generative topographic mapping (GTM), mixtures of first order factor analysers (MFA) and mixtures of multivariate Bernoulli distributions (MB). The x axis refers to the order of the factor analysis or principal components analysis, the number of mixture components in the case of mixture models and the square root of the number of basis functions in the case of GTM. Compare with the right graph of figures 5.11–5.13, 5.15 and 5.17.

Nevertheless, two-dimensional visualisation can be of interest in many speech applications, as we noted in section 5.5.

5.6.2 Model validity

5.6.2.1 Validity of a priori indices

The results from factor analysis show that:

- The similarity between the factors found and some of the linear data reduction indices proposed in the literature (Hardcastle et al., 1991b; Jones and Hardcastle, 1995), which have been designed using a priori assumptions, gives empirical support to the validity of the latter: such indices effectively capture a good deal of linear correlation in the data.
- However, the fact that the factors differ according to the EPG database employed indicates that fixed indices will not perform well on all databases (corresponding to different speakers, different speech styles, etc.). In order to capture as much structure as possible from a given database it is necessary to let the data speak for itself.

5.6.2.2 Model identifiability

The identifiability issue was discussed in sections 2.8 and 3.3: PCA and factor analysis are identifiable (up to (varimax) rotation); mixtures of multivariate Bernoulli distributions are not identifiable in theory but they generally are in practice; and no results are known for mixtures of factor analysers or GTM.

5.6.2.3 Model additivity

In section 2.6.2 we introduced the concept of model additivity for PCA and factor analysis, and we found that a similar phenomenon was observed for mixtures of multivariate Bernoulli distributions. There we claimed that often—whether the model is identifiable (e.g. factor analysis) or not (e.g. mixtures of multivariate Bernoulli distributions)—the factors or prototypes found in a model of order L are very approximately those of a model of order $L - 1$ plus a new one. Here we add some more facts, based on our experience with the data sets analysed in this chapter:

- Sometimes one does not obtain a new, stable factor or prototype but an intermediate one which will unfold into two new, stable ones in higher-order models.
- Past a certain order the models begin to produce repeated versions of some of the prototypes (typically the more relevant ones). This indicates that that order is probably too high and that a smaller one should be tried.
- While standard PCA is additive (the principal components being the eigenvectors of the sample covariance matrix ordered by the magnitude of the associated eigenvalue), PCA followed by varimax rotation is not. The components change considerably between adjacent orders.

In our data sets the phenomenon of additivity holds, but we do not have a theoretical basis for a general situation.

5.6.2.4 Continuous versus binary models

It may be argued a priori that using continuous variables to model binary data (or, in general, discrete data) is not a good idea, since binary D-dimensional vectors occupy only the 2^D corners of a hypercube (see section 4.3.1). However, a posteriori the model based on binary variables (the Bernoulli mixtures) performed worse than the ones based on continuous variables in terms of both log-likelihood and reconstruction error. There are other applications where models based on continuous variables have performed well with discrete data, such as self-organising maps for text categorisation (Kohonen, 1995), or some generative models for handwritten character recognition (Revow et al., 1996; Hinton et al., 1997).

Another common strategy to model binary data with underlying continuous variables is to squash the outputs of the continuous model to the $[0, 1]$ interval (e.g. with the logistic function $\sigma(\cdot)$) and use the resulting values as the parameters of a Bernoulli distribution. In fact, Bishop et al. (1998a) proposed such an extension for GTM (section 2.6.5.1), where each binary observed variable t_d receives a noise model $t_d|\mathbf{x} \sim \mathcal{B}(\sigma(f_d(\mathbf{x})))$

(and no variance parameter). However, the M step becomes more complex, requiring iterative, nonlinear optimisation, and we did not experiment with this model. A more crude possibility than using a soft squashing nonlinearity is to simply add a clipping nonlinearity to the outputs of the continuous model (e.g. Lee and Sompolinsky, 1999 do this to the outputs of a PCA model), although one would expect worse results.

Binary data do present a computational problem for linear methods, because the sample covariance matrix can be singular. This can be easily solved by adding a small amount of noise to the training data.

5.6.3 Training set preprocessing

For some data sets, particularly if they have a small number of EPG patterns, it can happen that the data matrix (having as row vectors the EPG patterns) is rank defective. This is due to the relatively high-dimensionality of the EPG patterns and the fact that, of all the frames contained in an utterance, only a small fraction are distinct. For example, the training sets for speakers FD, KM and PD had ranks 54, 50 and 56, respectively. In such cases, straightforward application of factor analysis and mixtures of factor analysers would fail (the sample covariance matrix being singular) and some of the other methods may also produce bad results. An obvious way to circumvent this problem is to first apply a PCA to the original matrix, eliminating all (nearly-)zero eigenvalues, and then apply any of the other methods. Another possibility is to add a small amount of noise to the data.

An important question is: given a sequence of EPG frames collected from an utterance, should all repetitions of EPG frames be removed? There are several reasons for and against removing repeated patterns:

- Practically, removing repeated patterns is possible both in batch and online learning algorithms and leads to a large improvement in training speed (since typically the size of the training set reduces to about 20% of the original one). Also, it happens that typically a handful of specific patterns (e.g. those corresponding to /d/ and vowels such as /æ/ and /ə/) can account for up to 30% of the EPG frames and obscure other less frequent patterns.

- However, in doing so one is equalising the natural prior distribution of the data. Also, if one removes repeated patterns and then adds some noise to the remaining patterns for numerical stability, the size of the remaining data set may be insufficient for methods with many parameters (e.g. GTM) and lead to overfitting.

We also trained the models presented in this work with non-repeated data and found little difference in the prototypes or factors found (not so in the mixing proportions, which are essentially related to the prior distribution of the components). This may reduce the importance of this issue.

5.6.4 Number of parameters

So far we have not said anything about the complexity of the models used, measured by the number of independent parameters to be fitted. The usual rule of thumb is that the sample size should be large enough for the estimation to be statistically sound, at least as large as the number of free parameters. Since our training sets have more than $N = 5\,000$ vectors for all speakers, this holds for all the models we have employed, except for GTM with more than 100 basis functions (which produced overfitting, as we saw in section 5.4.3). Table 5.1 lists the number of free parameters of each model. The general formulas are given by:

$$\textbf{PCA: } D(L+1) \qquad \textbf{FA: } D(L+2) - \frac{L(L-1)}{2} \qquad \textbf{GTM: } (F+1)D+1$$

$$\textbf{MFA: } DM(L+1) - \frac{ML(L-1)}{2} + M + D - 1 \qquad \textbf{MB: } M(D+1)-1$$

where $D = 62$ is the dimension of the data space, L is the number of principal components or factors, F is the number of basis functions in GTM (which we took equal to 81) and M is the number of components in a mixture. For the mixtures of factor analysers we took $L = 1$. So factor analysis and PCA have approximately the same number of parameters, while GTM can have many even for a dimension $L = 2$ of the latent space, due to the mapping between latent and data space being a radial basis functions expansion.

5.6.5 Computational considerations

Table 5.2 gives some average training and testing times for the methods considered and for typical values for the parameters (dimension of latent space, number of components of mixture, etc.). Training means here

L	PCA	FA
1	124	186
2	186	247
3	248	307
4	310	366
5	372	424
6	434	481
7	496	537
8	558	592
9	620	646
10	682	699
11	744	751
12	806	802
13	868	852
14	930	901
15	992	949

F	GTM
9	621
16	1055
25	1613
36	2295
49	3101
64	4031
81	5085
100	6263
121	7565
144	8991
169	10 541
196	12 215

M	MFA	MB
1	186	62
2	311	125
3	436	188
4	561	251
5	686	314
6	811	377
7	936	440
8	1 061	503
9	1 186	566
10	1 311	629
11	1 436	692
12	1 561	755
13	1 686	818
14	1 811	881
15	1 936	944

Table 5.1: Comparison between methods in terms of number of free parameters: factor analysis (FA), principal component analysis (PCA), generative topographic mapping (GTM), mixtures of factor analysers (MFA) and mixtures of multivariate Bernoulli distributions (MB).

	PCA	FA	GTM	MFA	MB
Training	9.1	13.5	4 000	1 253	670
Testing	$9 \cdot 10^{-5}$	$9 \cdot 10^{-5}$	$3.2 \cdot 10^{-3}$	$5.2 \cdot 10^{-3}$	$5.6 \cdot 10^{-3}$

Table 5.2: Comparison between methods in terms of CPU consumption. Times are given in seconds. For training, the data set size was $N = 6892$. The testing time is per EPG pattern. The methods were run with the following parameters: PCA: $L = 9$ principal components. Factor analysis (FA): $L = 9$ factors. GTM: latent space of dimension $L = 2$, with $K = 400$ latent space points and $M = 49$ basis functions. Mixtures of factor analysers (MFA): $M = 9$ components, each a factor analysis of $L = 1$ factor. Mixtures of multivariate Bernoulli distributions (MB): $M = 9$ components.

the process of producing a maximum likelihood estimate for all the model parameters given a training set (of size $N = 6\,892$ patterns). Testing is the process of producing a reduced-dimension representative of a given EPG pattern. All the times correspond to Matlab implementations of the various methods in a Sun Ultra-170 workstation[16].

We can see that for an EPG sampling frequency of 200 Hz, all the methods are able to produce a reduced-dimension representative of the current sample in real time. This would allow the use of any of them in a system for speech therapy or speech assessment. Training requires a relatively long time for GTM, a moderate time for the mixtures of multivariate Bernoulli distributions and mixtures of factor analysers, and virtually no time for factor analysis and PCA. C implementations of the methods considered would reduce considerably both the computation time and the memory requirements.

5.6.6 Intrinsic dimensionality of the EPG data

All the models described need to know in advance either the dimension of the latent space L or the number of components in the mixture M. This is a common disadvantage of most dimensionality reduction methods and is directly related to the determination of the intrinsic dimensionality of the EPG data. That is, the number of degrees of freedom of the articulatory system that generates the EPG patterns: the tongue and the hard palate. This is a matter currently being investigated (e.g. Stone, 1991; Nguyen et al., 1994, 1996 and references therein); a dimension of 5 to 10 is usually suggested. Alternatively, one can try to infer this intrinsic dimension directly from the data set by standard statistical techniques of model selection (see e.g. Bishop, 1995, pp. 371–377); or by methods to estimate the (fractal) dimension, such as box counting—although in

[16]This is a slower computer than that used in section 10.2.4.

practice these methods are generally limited to small dimensions (see e.g. Falconer, 1990, pp. 38ff and Peitgen et al., 1992, pp. 212ff, 721ff).

In the models analysed in this chapter, one can often get a rough idea of what the appropriate number of latent variables or components is by examining the factors or prototypes found. For example, if in a mixture of multivariate Bernoulli distributions several prototypes look very similar (e.g. see fig. 5.16), it is obvious that one should try a mixture with fewer components even if the log-likelihood decreases a bit. Also, plotting the log-likelihood in the training and test sets versus the number of latent variables or components can help to avoid overfitting by selecting L or M such that the log-likelihood with the test set is maximal.

The fact that the dimension of the latent space or the number of components found in any of these ways may be optimal with respect to the model does not guarantee that it actually matches the intrinsic dimensionality of the EPG data, because the model may be bad for this data. Nevertheless, the two-dimensional representation produced by GTM and its performance in log-likelihood and reconstruction error (compared to that of the other methods, which use more latent variables) suggest that the intrinsic dimensionality of the EPG data may be substantially smaller than that claimed by other studies.

Determining the intrinsic dimensionality of the EPG data would be a significant step towards the solution of the acoustic-to-articulatory mapping and the construction of a realistic mechanical model of the vocal tract. However, a purely phenomenological, or black-box, approach—where all hope for finding a simple model is abandoned—may be more fruitful from a practical point of view. That is, a low-dimensional model of high complexity (with a high number of parameters) may offer no help in understanding the physical behaviour of the articulatory system yet be able to mimic it to a desired level of accuracy. In chapter 10 we look further into the acoustic-to-articulary mapping problem.

5.7 Conclusion

We have shown that latent variable models, owing to their ability to produce a reduced-dimension representation of a data set, can be useful to present the relatively high-dimensional information contained in the EPG sequence in a way which is easier to understand and handle, e.g.:

- The loading vectors and principal component vectors of factor analysis and PCA, respectively, can be used as conventional EPG data reduction indices, with the advantage of adapting better to the database under consideration.
- A two-dimensional latent space representation can be used for the analysis of EPG sequences in several applications, such as speech therapy, language learning and articulatory dynamics.

This is a significant advantage over general probability models and mixtures of them, which are suitable for classification or clustering but not for dimensionality reduction.

The superior performance of a two-dimensional GTM model over the other methods indicates that the EPG data may be appropriately accounted for with a very small number of degrees of freedom as long as a nonlinear, complex model is used.

A direction for future work is the development of a dimension-reduced representation for EPG data that takes into account their temporal dependence. Integration of well-known modelling tools for time-varying data, such as hidden Markov models, with static dimensionality reduction methods, such as those discussed here, seems a promising avenue for research.

Our Matlab implementation of factor analysis, varimax rotation, principal component analysis and imaging tools for EPG data is freely available in the Internet (see appendix C).

Part III

Sequential data reconstruction

The central assumption of this thesis is the fact that multivariate data often embody relations between variables so that a lower-dimensional representation is possible. We dealt with the problem of finding such a representation (i.e., dimensionality reduction) in parts I and II, and we saw probabilistic and non-probabilistic methods, respectively, for that. Finding a low-dimensional representation can be seen as inferring the values of some invented variables given the values of the variables that are actually observed. In part III we deal with a somewhat different problem: given the values of some (not all) the observed variables, infer the values of the remaining observed variables. This problem is called missing data reconstruction. Again, the solution of this problem is partly possible due to the redundancy caused by the relations between observed variables. Such redundancy is necessary but not sufficient in general and an extra form of redundancy is necessary to solve the problem, as we will see.

The relations between variables mentioned earlier usually are due to the existence of a forward (univalued) mapping, whose inverse mapping is then typically multivalued. The pattern recognition literature sometimes uses the terms inverse mapping and inverse problem interchangeably, but in general they mean different things. Chapter 6 is devoted to reviewing inverse problems and clarifying when they become equivalent to inverting a mapping. Chapter 7 presents the problem of missing data reconstruction in general and proposes a method to solve it. Its title, "sequential data reconstruction," reflects the fact that most of the time we will consider a particular and important case of missing data reconstruction, but the ideas are readily generalisable. The current implementation of the method relies on an algorithm, explained in chapter 8, to find all the modes of an arbitrary Gaussian mixture. Chapters 9 and 10 present experimental results using synthetic and real-world data sets, respectively.

Chapter 6

Inverse problems and mapping inversion

6.1 Introduction

The concepts of "inverse problem" and "mapping inversion" are often used interchangeably in the machine learning literature, although they denote different things. The aim of this chapter is: to introduce the area of (Bayesian) inverse problem theory; to compare it with general Bayesian analysis and in particular with latent variable models; and to differentiate it from the problems of mapping inversion and mapping approximation.

Section 6.2 defines inverse problem theory, explains the reasons for non-uniqueness of the solution and reviews Bayesian inverse problem theory, including the topics of the choice of prior distributions, stability and regularisation. It also describes several examples of inverse problems in some detail to clarify the theory and its interpretation. Section 6.3 compares Bayesian inverse problem theory with general Bayesian analysis and in particular with latent variable models. Section 6.4 defines (statistical) mapping inversion, mapping approximation and universal mapping approximators.

6.2 Inverse problem theory

6.2.1 Introduction and definitions

Inverse calculations involve making inferences about models of physical systems from data[1]. The scientific procedure to study a physical system can be divided into three parts:

1. *Parameterisation of the system*: discovery of a minimal set of model parameters[2] whose values completely characterise the system.

2. *Forward modelling*: discovery of the physical laws allowing, for given values of the parameters, predictions of some observable or data parameters to be made.

3. *Inverse modelling*: use of actual measurements of the observed parameters to infer the values of the model parameters. This inference problem is termed the **inverse problem**.

The model parameters conform the model space $\mathcal{M}$, the observable parameters conform the data space $\mathcal{D}$ and the union of both parameter sets conforms the parameter space $\mathcal{X} = \mathcal{D} \times \mathcal{M}$. See section 6.2.3.1 for a further interpretation of the model parameters.

Usually the forward problem is a well-defined single-valued relationship (i.e., a function in the mathematical sense) so that given the values of the model parameters the values of the measured parameters are uniquely

[1]Tarantola (1987) claims that inverse problem theory in the wide sense has been developed by people working with geophysical data, because geophysicists try to understand the Earth's interior but can only use data collected at the Earth's surface. However, inverse problems appear in many other areas of physics and engineering, some of which are briefly reviewed in section 6.2.4.

[2]The term *parameters* is used in inverse problem theory to mean both the variables and the parameters of a model, as these terms are usually understood in machine learning. Throughout this chapter, we will keep the notation and naming convention which is standard in inverse problem theory. In section 6.3 we discuss the point of view of probabilistic models.

identified. This is often due to causality in the physical system. But often this forward mapping is many-to-one, so that the inverse problem is one-to-many: given values of the observed parameters, there is more than one model (possibly an infinite number) that corresponds to them.

Thus, if $\mathbf{g} : \mathcal{M} \to \mathcal{D}$ is the forward mapping, then $\mathbf{d} = \mathbf{g}(\mathbf{m})$ is unique given $\mathbf{m}$, but its inverse $\mathbf{g}^{-1}(\mathbf{d})$ can take several values for some observed $\mathbf{d} \in \mathcal{D}$.

The example in section 6.2.4.1 illustrates this abstract formulation.

6.2.1.1 Types of inverse problems

Inverse problems can be classified as:

Continuous Most inverse problems are of this type. The model to be estimated is a continuous function in several variables. For example, the mass density distribution inside the Earth as a function of the space coordinates.

Discrete There is a finite (actually numerable) number of model parameters to be estimated. Sometimes the problem itself is discrete in nature, e.g. the location of the epicentre in the example 6.2.4.1, which is parameterised by the epicentre coordinates X and Y. But most times, the problem was originally continuous and was discretised for computational reasons. For example, one can express the mass density distribution inside the Earth as a parameterised function in spherical coordinates (perhaps obtained as the truncation of an infinite parameterised series) or as a discrete grid (if the sampling length is small enough).

In this chapter we deal only with discrete inverse problems. Tarantola (1987) discusses both discrete and continuous inverse problems.

6.2.1.2 Why the nonuniqueness?

Nonuniqueness arises for several reasons:

- Intrinsic lack of data: for example, consider the problem of estimating the density distribution of matter inside the Earth from knowledge of the gravitational field at its surface. Gauss' theorem shows that there are infinitely many different distributions of matter density that give rise to identical exterior gravitational fields. In this case, it is necessary to have additional information (such as a priori assumptions on the density distribution) or additional data (such as seismic observations).

- Uncertainty of knowledge: the observed values always have experimental uncertainty and the physical theories of the forward problem are always approximations of the reality.

- Finiteness of observed data: continuous inverse problems have an infinite number of degrees of freedom. However, in a realistic experiment the amount of data is finite and therefore the problem is underdetermined.

6.2.1.3 Stability and ill-posedness of inverse problems

In the Hadamard sense, a well-posed problem must satisfy certain conditions of existence, uniqueness and continuity. Ill-posed problems can be numerically unstable, i.e., sensitive to small errors in the data (arbitrarily small changes in the data may lead to arbitrarily large changes in the solution).

Nonlinearity has been shown to be a source of ill-posedness (Snieder and Trampert, 1999), but linearised inverse problems can often be ill-posed too due to the fact that realistic data is finite. Therefore, inverse problems in general might not have a solution in the strict sense, or if there is a solution, it might not be unique or might not depend continuously on the data. To cope with this problem, stabilising procedures such as regularisation methods are often used (Tikhonov and Arsenin, 1977; Engl et al., 1996). Bayesian inversion is, in principle, always well-posed (see section 6.2.3.5). Mapping inversion (section 6.4) is also a numerically unstable problem but, again, probabilistic methods such as the one we develop in chapter 7 are well-posed.

6.2.2 Non-probabilistic inverse problem theory

For some inverse problems, such as the reconstruction of the mass density of a one-dimensional string from measurements of all eigenfrequencies of vibration of the string, an exact theory for inversion is available (Snieder and Trampert, 1999). Although these exact nonlinear inversion techniques are mathematically elegant, they are of limited applicability because:

- They are only applicable to idealistic situations which usually do not hold in practice. That is, the physical models for which an exact inversion method exists are only crude approximations of reality.
- They are numerically unstable.
- The discretisation of the problem caused by the fact that the data are only available in a finite amount makes the problem underdetermined.

Non-probabilistic inversion methods attempt to invert the mathematical equation of the forward mapping (for example, solving a linear system of equations by using the pseudoinverse). These methods cannot deal with data uncertainty and redundancy in a natural way, and we do not deal with such methods here. A more general formulation of inverse problems is obtained using probability theory.

6.2.3 Bayesian inverse problem theory

The standard reference for the Bayesian view of (geophysical) inversion is Tarantola (1987), whose notation we use in this section; the standard reference for the frequentist inverse theory is Parker (1994); other references are Scales and Smith (1998) and Snieder and Trampert (1999).

In the Bayesian approach to inverse problems, we use physical information about the problem, plus possibly uninformative prior distributions, to construct the following two models:

- A joint prior distribution $\rho(\mathbf{d}, \mathbf{m})$ in the parameter space $\mathcal{X} = \mathcal{D} \times \mathcal{M}$. This prior distribution is usually factorised as $\rho_{\mathcal{D}}(\mathbf{d})\rho_{\mathcal{M}}(\mathbf{m})$, because by definition the a priori information on the model parameters is independent of the observations. However, it may happen that part of this prior information was obtained from a preliminary analysis of the observations, in which case $\rho(\mathbf{d}, \mathbf{m})$ might not be factorisable. If no prior information is available, then an uninformative prior may be used (see section 6.2.3.2).
- Using information obtained from physical theories we solve the forward problem, deriving a deterministic forward mapping $\mathbf{d} = \mathbf{g}(\mathbf{m})$. If a noise model f (typically normal) is applied, a conditional distribution $\theta(\mathbf{d}|\mathbf{m}) = f(\mathbf{d} - \mathbf{g}(\mathbf{m}))$ may be derived. For greater generality, the information about the resolution of the forward problem is described by a joint density function $\theta(\mathbf{d}, \mathbf{m})$. However, usually $\theta(\mathbf{d}, \mathbf{m}) = \theta(\mathbf{d}|\mathbf{m})\mu_{\mathcal{M}}(\mathbf{m})$, where $\mu_{\mathcal{M}}(\mathbf{m})$ describes the state of null information on model parameters.

Tarantola (1987) postulates that the a posteriori state of information is given by the *conjunction* of the two states of information: the prior distribution on the $\mathcal{D} \times \mathcal{M}$ space and the information about the physical correlations between $\mathbf{d}$ and $\mathbf{m}$. The conjunction is defined as

$$\sigma(\mathbf{d}, \mathbf{m}) \stackrel{\text{def}}{=} \frac{\rho(\mathbf{d}, \mathbf{m})\theta(\mathbf{d}, \mathbf{m})}{\mu(\mathbf{d}, \mathbf{m})} \tag{6.1}$$

where $\mu(\mathbf{d}, \mathbf{m})$ is a distribution representing the state of null information (section 6.2.3.2). Thus, all available information is assimilated into the posterior distribution of the model given the observed data, computed by marginalising the "joint posterior distribution" σ (assuming factorised priors $\rho(\mathbf{d}, \mathbf{m})$ and $\mu(\mathbf{d}, \mathbf{m})$):

$$\sigma_{\mathcal{M}}(\mathbf{m}) = \int_{\mathcal{D}} \sigma(\mathbf{d}, \mathbf{m})\, d\mathbf{d} = \rho_{\mathcal{M}}(\mathbf{m})L(\mathbf{m}) \tag{6.2}$$

where the likelihood function L, which measures the data fit, is defined as:

$$L(\mathbf{m}) \stackrel{\text{def}}{=} \int_{\mathcal{D}} \frac{\rho_{\mathcal{D}}(\mathbf{d})\theta(\mathbf{d}|\mathbf{m})}{\mu_{\mathcal{D}}(\mathbf{d})}\, d\mathbf{d}.$$

Thus, the "solution" of the Bayesian inversion method is the posterior distribution $\sigma_{\mathcal{M}}(\mathbf{m})$, which is unique (although it may be multimodal and even not normalisable, depending on the problem). Usually a maximum a posteriori (MAP) approach is adopted, so that we take the model maximising the posterior probability σ: $\mathbf{m}_{\text{MAP}} \stackrel{\text{def}}{=} \max_{\mathbf{m} \in \mathcal{M}} \sigma_{\mathcal{M}}(\mathbf{m})$.

Likewise, the posterior distribution in the data space is calculated as

$$\sigma_{\mathcal{D}}(\mathbf{d}) = \int_{\mathcal{M}} \sigma(\mathbf{d}, \mathbf{m})\, d\mathbf{m} = \frac{\rho_{\mathcal{D}}(\mathbf{d})}{\mu_{\mathcal{D}}(\mathbf{d})} \int_{\mathcal{M}} \theta(\mathbf{d}|\mathbf{m})\rho_{\mathcal{M}}(\mathbf{m})\, d\mathbf{m}$$

which allows to estimate posterior values of the data parameters (*recalculated data*).

In practice, all uncertainties are described by stationary Gaussian distributions:

- likelihood $L(\mathbf{m}) \sim \mathcal{N}(\mathbf{g}(\mathbf{m}), \mathbf{C}_{\mathcal{D}})$
- prior $\rho_{\mathcal{M}}(\mathbf{m}) \sim \mathcal{N}(\mathbf{m}_{\text{prior}}, \mathbf{C}_{\mathcal{M}})$.

A more straightforward approach that still encapsulates all the relevant features (uninformative priors and an uncertain forward problem) is simply to obtain the posterior distribution of the model given the observed data as

$$p(\mathbf{m}|\mathbf{d}) = \frac{p(\mathbf{d}|\mathbf{m})p(\mathbf{m})}{p(\mathbf{d})} \propto p(\mathbf{d}|\mathbf{m})p(\mathbf{m}). \tag{6.3}$$

This equation should be more familiar to statistical learning researchers.

6.2.3.1 Interpretation of the model parameters

Although the treatment of section 6.2.3 is perfectly general, it is convenient to classify the model parameters into one of two types:

- Parameters that describe the *configuration* or *state* of the physical system and completely characterise it. In this case, they could be called *state variables* as in dynamical system theory. We represent them as a vector $\mathbf{s}$. Examples of such parameters are the location of the epicentre in example 6.2.4.1 or the absorption coefficient distribution of a medium in CAT (example 6.2.4.3). These parameters are independent, in principle, of any measurements taken of the system (such as a particular projection in CAT or a measurement of the arrival time of the seismic wave in example 6.2.4.1).

- Parameters that describe the *experimental conditions* in which a particular measurement of the system was taken. Thus, for each measurement $\mathbf{d}_n$ we have a vector $\mathbf{c}_n$ indicating the conditions in which it was taken. For example, in 2D CAT (example 6.2.4.3) one measurement is obtained from a given X-ray source at plane coordinates x, y and at an angle θ; thus $\mathbf{c}_n = (x_n, y_n, \theta_n)$ and the measurement $\mathbf{d}_n$ is the transmittance. In example 6.2.4.1, one measurement is taken at the location (x_n, y_n) of station n; and so on. If there are N measurements, then the model parameters are $\{\mathbf{c}_n\}_{n=1}^N$ (in addition to the $\mathbf{s}$ model parameters) and one can postulate prior distributions for them to indicate uncertainties in their determination. However, usually one assumes that there is no uncertainty involved in the conditions of the measurement and takes these distributions as Dirac deltas. Of course, the measured value $\mathbf{d}_n$ can still have a proper distribution reflecting uncertainty in the actual measurement. In this way, the estimation problem is simplified, because all $\{\mathbf{c}_n\}_{n=1}^N$ model parameters are considered constant, and only the $\mathbf{s}$ model parameters are estimated.

From a Bayesian standpoint, there is no formal difference between both kinds of model parameters, state $\mathbf{s}$ and experimental conditions $\mathbf{c}$—or between model parameters $\mathbf{m}$ and data parameters $\mathbf{d}$, for that matter—because probability distributions are considered for all variables and parameters of the problem.

Thus, if there are N measurements, the forward mapping is $\mathbf{d} = \mathbf{g}(\mathbf{m})$ with $\mathbf{d} = (\mathbf{d}_1, \dots, \mathbf{d}_N)$ and $\mathbf{m}$ including both kinds of parameters (state variables and experimental conditions): $\mathbf{m} = (\mathbf{s}, \mathbf{c}_1, \dots, \mathbf{c}_N)$. Usually the measurements are taken independently, so that

$$\rho_{\mathcal{D}}(\mathbf{d}) = \prod_{n=1}^{N} \rho_{\mathcal{D},n}(\mathbf{d}_n) \qquad \mu_{\mathcal{D}}(\mathbf{d}) = \prod_{n=1}^{N} \mu_{\mathcal{D},n}(\mathbf{d}_n)$$

and the forward mapping decomposes into N equations $\mathbf{d}_n = \mathbf{g}_n(\mathbf{s}, \mathbf{c}_n)$. Thus $\theta(\mathbf{d}|\mathbf{m}) = \prod_{n=1}^N \theta_n(\mathbf{d}_n|\mathbf{s}, \mathbf{c}_n) = \prod_{n=1}^N f(\mathbf{d}_n - \mathbf{g}_n(\mathbf{s}, \mathbf{c}_n))$ and the likelihood function factorises as $L(\mathbf{m}) = \prod_{n=1}^N L_n(\mathbf{m})$ with

$$L_n(\mathbf{m}) \stackrel{\text{def}}{=} \int_{\mathcal{D}_n} \frac{\rho_{\mathcal{D},n}(\mathbf{d}_n)\theta_n(\mathbf{d}_n|\mathbf{m}, \mathbf{c}_n)}{\mu_{\mathcal{D},n}(\mathbf{d}_n)} \, d\mathbf{d}_n.$$

6.2.3.2 Choice of prior distributions

The controversial matter in the Bayesian approach is, of course, the construction of prior distributions. Usual ways to do this are (Jaynes, 1968; Kass and Wasserman, 1996):

- Define a measure of information, such as the entropy, and determine the distribution that optimises it (e.g. maximum entropy).

- Define properties that the noninformative prior should have, such as invariance to certain transformations (Jeffreys' prior). For example, for a given definition of the physical parameters $\mathbf{x}$, it is possible to find a unique density function $\mu(\mathbf{x})$ which is form invariant under the transformation groups which leave the fundamental equations of physics invariant.
- The previous choices, usually called "objective" or "noninformative," are constructed by some formal rule, but it is also possible to use priors based on subjective knowledge.

In any case, the null information distributions are obtained in each particular case, depending on the coordinate systems involved, etc. However, the choice of a noninformative distribution for a continuous, multidimensional space remains a delicate problem. Bernardo and Smith (1994, pp. 357–367) discuss this issue.

6.2.3.3 Bayesian linear inversion theory

Assuming that all uncertainties are Gaussian, if the forward operator is linear, then the posterior distribution σ will also be Gaussian. This is equivalent to factor analysis, which is a latent variable model where the prior distribution in latent space is Gaussian, the mapping from latent onto data space is linear and the noise model in data space is Gaussian.

Linear inversion theory is well developed and involves standard linear algebra techniques: pseudoinverse, singular value decomposition and (weighted) least squares problems. Tarantola (1987) gives a detailed exposition.

6.2.3.4 Bayesian nonlinear inversion theory

Almost all work in nonlinear inversion theory, particularly in geophysics, is based on linearising the problem using physical information. Usual linearisation techniques include the Born approximation (also called the single-scattering approximation), Fermat's principle and Rayleigh's principle (Snieder and Trampert, 1999).

6.2.3.5 Stability

In the Bayesian approach to inverse problems, it is not necessary in principle to invert any operators to construct the solution to the inverse problem, i.e., the posterior probability σ. Thus, from the Bayesian point of view, no inverse problem is ill-posed (Gouveia and Scales, 1998).

6.2.3.6 Confidence sets

Once a MAP model has been selected from the posterior distribution σ, confidence sets or other measures of resolution can be extracted from $\sigma(\mathbf{m}_{\text{MAP}})$. Due to the mathematical complexity of this posterior distribution, only approximate techniques are possible, including the following ones:

- The forward operator $\mathbf{g}$ is linearised about the selected model $\mathbf{m}_{\text{MAP}}$, so that the posterior becomes normal: $\sigma(\mathbf{m}) \sim \mathcal{N}(\mathbf{m}_{\text{MAP}}, \mathbf{C}'_{\mathcal{M}})$. The posterior covariance matrix $\mathbf{C}'_{\mathcal{M}}$ is obtained as $\mathbf{C}'_{\mathcal{M}} = (\mathbf{G}^T\mathbf{C}_{\mathcal{D}}^{-1}\mathbf{G} + \mathbf{C}_{\mathcal{M}}^{-1})^{-1}$, where $\mathbf{G}$ is the derivative[3] of $\mathbf{g}$ with respect to the model parameters evaluated at $\mathbf{m}_{\text{MAP}}$. This has the same form as the posterior covariance matrix in latent space of a factor analysis, as in eq. (2.59), where $\mathbf{G}$ would be the factor loadings matrix $\mathbf{\Lambda}$.
- Sampling the posterior distribution with Markov chain Monte Carlo methods (Mosegaard and Tarantola, 1995).

6.2.3.7 Occam's inversion

In Occam's inversion (Constable et al., 1987), the goal is to construct the smoothest model consistent with the data. This is not to say that one believes a priori that models are really smooth, but rather that a more conservative interpretation of the data should be made by eliminating features of the model that are not required to fit the data. To this effect, they define two measures:

[3]The linearised mapping $\mathbf{g}(\mathbf{m}_{\text{MAP}}) + \mathbf{G}(\mathbf{m} - \mathbf{m}_{\text{MAP}})$ is usually called Fréchet derivative in inverse problem theory, and is the linear mapping tangent to $\mathbf{g}$ at $\mathbf{m}_{\text{MAP}}$.

- A measure of data fit (irrespective of any Bayesian interpretation of the models):

$$d(\mathbf{m},\mathbf{d}) \stackrel{\text{def}}{=} (\mathbf{g}(\mathbf{m})-\mathbf{d})^T \mathbf{C}_{\mathcal{D}}^{-1}(\mathbf{g}(\mathbf{m})-\mathbf{d}),$$

  that is, the Mahalanobis distance between $\mathbf{g}(\mathbf{m})$ and $\mathbf{d}$ with matrix $\mathbf{C}_{\mathcal{D}}^{-1}$.

- A measure of model smoothness: $\|\mathbf{Rm}\|$, where $\mathbf{R}$ is a Tikhonov roughening operator (Tikhonov and Arsenin, 1977), e.g. a discrete second-difference operator, such as

$$\mathbf{R} = \begin{pmatrix} -2 & 1 & 0 & 0 & \dots & 0 & 0 & 0 \\ 1 & -2 & 1 & 0 & \dots & 0 & 0 & 0 \\ 0 & 1 & -2 & 1 & \dots & 0 & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & 0 & \dots & 1 & -2 & 1 \\ 0 & 0 & 0 & 0 & \dots & 0 & 1 & -2 \end{pmatrix}. \tag{6.4}$$

Then, Occam's inversion finds a model being both smooth and fitting well the data by solving the optimisation problem:

$$\min_{\mathbf{m}\in\mathcal{M}} \|\mathbf{Rm}\| \text{ subject to } d(\mathbf{m},\mathbf{d}) \le \epsilon$$

for some tolerance ϵ. Practically, due to the distance d being a quadratic form, this can be conveniently implemented as a weighted least-squares problem with a Lagrange multiplier to control the tradeoff between model smoothness and data fit: for fixed λ, solve the weighted, regularised least-squares problem

$$\min_{\mathbf{m}\in\mathcal{M}} (\mathbf{g}(\mathbf{m})-\mathbf{d})^T \mathbf{C}_{\mathcal{D}}^{-1}(\mathbf{g}(\mathbf{m})-\mathbf{d}) + \lambda(\mathbf{m}-\mathbf{m}_{\text{prior}})^T \mathbf{R}^T\mathbf{R}(\mathbf{m}-\mathbf{m}_{\text{prior}}). \tag{6.5}$$

Then, increase λ until $(\mathbf{g}(\mathbf{m})-\mathbf{d})^T \mathbf{C}_{\mathcal{D}}^{-1}(\mathbf{g}(\mathbf{m})-\mathbf{d}) > \epsilon$.

Clearly, Occam's inversion is a particular case of Bayesian inversion, in which the uncertainty distributions are taken as Gaussians (with the appropriate covariance matrix) and the prior distribution over the models is used as a smoothness regularisation term by taking $\mathbf{C}_{\mathcal{M}}^{-1} = \lambda\mathbf{R}^T\mathbf{R}$. However, Bayesian inversion is more general than Occam's inversion in that the prior distributions allow to introduce physical knowledge into the problem. Gouveia and Scales (1997) compare Bayes' and Occam's inversion in a seismic data problem.

6.2.3.8 Locally independent inverse problems

In the general statement of inverse problems, the data parameters $\mathbf{d}$ depend on all the model parameters $\mathbf{m}$ which in turn are a continuous function of some independent variables, such as the spatial coordinates $\mathbf{x}$. Thus $\mathbf{m} = \mathbf{m}(\mathbf{x})$ and $\mathbf{d} = \mathbf{g}(\mathbf{m})$. A single datum parameter depends on the whole function $\mathbf{m}(\cdot)$, even if that datum was measured at point $\mathbf{x}$ only.

Sometimes we can assume locally independent problems, so that a datum measured at point $\mathbf{x}$ depends only on the value of $\mathbf{m}(\mathbf{x})$, not on the whole function $\mathbf{m}$ for all $\mathbf{x}$. If we have N measurements $\{\mathbf{d}_n\}_{n=1}^N$ at points $\{\mathbf{x}_n\}_{n=1}^N$ and we discretise the problem, so that we have one model parameter $\mathbf{m}_n$ at point $\mathbf{x}_n$, for $n = 1,\dots,N$, then:

$$\theta(\mathbf{d}|\mathbf{m}) = \prod_{n=1}^N \vartheta(\mathbf{d}_n|\mathbf{m}_n) \Longrightarrow \theta(\mathbf{d},\mathbf{m}) \propto \rho_{\mathcal{M}}(\mathbf{m}) \prod_{n=1}^N \vartheta(\mathbf{d}_n|\mathbf{m}_n) \tag{6.6}$$

where the distribution ϑ is the same for all parameters because the function $\mathbf{g}$ is now the same for all values of $\mathbf{m}$. This is equivalent to inverting the mapping $\mathbf{x} \to \mathbf{m}(\mathbf{x}) \xrightarrow{\mathbf{g}} \mathbf{d}(\mathbf{m}(\mathbf{x}))$ at data values $\mathbf{d}_1,\dots,\mathbf{d}_N$ and obtaining values $\mathbf{m}_1 = \mathbf{g}^{-1}(\mathbf{d}_1),\dots,\mathbf{m}_N = \mathbf{g}^{-1}(\mathbf{d}_N)$. This approach is followed in the example of section 6.2.4.2. We deal with problems of this kind in section 6.4.1 and give examples. However, this simplification cannot be applied generally. For example, for the CAT problem (section 6.2.4.3) a measurement depends on all the points that they ray travels through.

If we also assume an independent prior distribution for the parameters, $\rho_{\mathcal{M}}(\mathbf{m}) = \prod_{n=1}^N \varrho_{\mathcal{M}}(\mathbf{m}_n)$, then the complete inverse problem factorises into N independent problems:

$$\theta(\mathbf{d}|\mathbf{m}) \propto \prod_{n=1}^N \varrho_{\mathcal{M}}(\mathbf{m}_n)\vartheta(\mathbf{d}_n|\mathbf{m}_n).$$

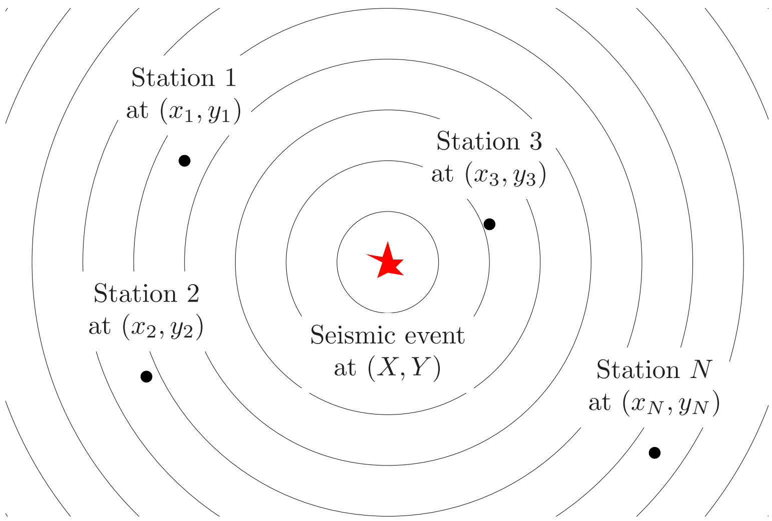


Figure 6.1: A seismic event takes place at time $\tau = 0$ at location (X, Y) and the seismic waves produced are recorded by several seismic stations of Cartesian coordinates $\{(x_n, y_n)\}_{n=1}^N$ at times $\{d_n\}_{n=1}^N$. Determining the location of the epicentre from the wave arrival times is an inverse problem.

6.2.4 Examples of inverse problems

To clarify the concepts exposed, we briefly review some examples of inverse problems, some of which have a rich literature.

6.2.4.1 Locating the epicentre of a seismic event

We consider the simplified problem[4] of estimating the epicentral coordinates of a seismic event (e.g. a nuclear explosion), depicted in fig. 6.1. The event takes place at time $\tau = 0$ at an unknown location (X, Y) on the surface of the Earth (considered flat). The seismic waves produced by the explosion are recorded in a network of N seismic stations of Cartesian coordinates $\{(x_n, y_n)\}_{n=1}^N$, so that $\mathbf{d}_n = d_n$ is the observed arrival time of the seismic wave at station n. The waves travel at a velocity v in all directions.

The model parameters to be determined from the data parameters $\{d_n\}_{n=1}^N$ are:

- State parameters: the coordinates of the epicentre (X, Y).
- Experimental condition parameters: the coordinates of each station (x_n, y_n), the time of the event τ and the wave velocity v.

Thus $\mathbf{m} = (\mathbf{s}, \mathbf{c}_1, \dots, \mathbf{c}_N) = (X, Y, \tau, v, x_1, y_1, \dots, x_N, y_N)$. Assuming that the station coordinates, the time of the event and the wave velocity are perfectly known, we can drop them and avoid defining prior distributions for them, so that $\mathbf{m} = (X, Y)$.

Given (X, Y), the arrival times of the seismic wave at the stations can be computed exactly as $\mathbf{d}_n = \mathbf{g}_n(X, Y) = \frac{1}{v}\sqrt{(x_n - X)^2 + (y_n - Y)^2}$ for $n = 1, \dots, N$, which solves the forward problem. Determining the epicentre coordinates (X, Y) from the arrival times at the different stations is the inverse problem, whose complete solution is given by Tarantola (1987).

6.2.4.2 Retrieval of scatterometer wind fields

A satellite can measure the amount of backscatter generated by small ripples on the ocean surface, in turn produced by oceanic wind fields. The satellite scatterometer consists of a line of cells, each capable to detect backscatter at the location to which it is pointing. At a given position in space of the satellite, each cell records a *local measurement*. A *field* is defined as a spatially connected set of local measurements obtained from a swathe, swept by the satellite along its orbit. For example, for the ESA satellite ERS-1, which follows a polar

[4] This example is adapted from problem 1.1 of Tarantola (1987, pp. 85–91).

orbit, the swathe contains 19 cells and is approximately 500 km wide. Each cell samples an area of around 50×50 km, with some overlap between samples.

The *backscatter* $\boldsymbol{\sigma}^0$ is a three-dimensional vector because each cell is sampled from three different directions by the fore, mid and aft beams, respectively. The near-surface *wind vector* $\mathbf{u}$ is a two-dimensional, quasicontinuous function of the oceanic spatial coordinates (although see comments about the wind continuity in section 7.9.6). Both $\boldsymbol{\sigma}^0$ and $\mathbf{u}$ contain noise, although the noise in $\boldsymbol{\sigma}^0$ is dominated by that of $\mathbf{u}$. A backscatter field is written as $\boldsymbol{\Sigma}^0 = (\boldsymbol{\sigma}_i^0)$ and a wind field as $\mathbf{U} = (\mathbf{u}_i)$.

The forward problem is to obtain $\boldsymbol{\sigma}^0$ from $\mathbf{u}$ and is single-valued and relatively easy to solve. The inverse problem, to obtain the wind field from the backscatter, is one-to-many and no realistic physically-based local inverse model is possible. The aim of the inversion is to produce a wind field $\mathbf{U}$ that can be used in data assimilation for numerical weather prediction (NWP) models. The most common method for inversion is to use lookup tables and interpolation. Following the standard Bayesian approach of inverse problem theory described in section 6.2.3, Cornford and colleagues[5] (Cornford et al., 1999a; Nabney et al., 2000; Evans et al., 2000) model the conditional distribution $p(\boldsymbol{\Sigma}^0|\mathbf{U})$ and the prior distribution of the wind fields $p(\mathbf{U})$. The prior is taken as a zero-mean normal, $p(\mathbf{U}) \sim \mathcal{N}(\mathbf{0}, \mathbf{K_U})$. The conditional distribution of the backscatter field $\boldsymbol{\Sigma}^0$ given the wind field $\mathbf{U}$ can be factorised into the individual distributions at each point in the region (i.e., at each cell) as $p(\boldsymbol{\Sigma}^0|\mathbf{U}) = \prod_i p(\boldsymbol{\sigma}_i^0|\mathbf{u}_i)$ because theoretically there is a single-valued mapping $\mathbf{u} \rightarrow \boldsymbol{\sigma}^0$. However, rather than using a physical forward model to obtain a noise model $p(\boldsymbol{\sigma}^0|\mathbf{u})$, which is difficult, they use Bayes' theorem to obtain $p(\boldsymbol{\sigma}^0|\mathbf{u}) \propto p(\mathbf{u}|\boldsymbol{\sigma}^0)/p(\mathbf{u})$, the factor $p(\boldsymbol{\sigma}^0)$ being constant for a given data, and they model $p(\mathbf{u}|\boldsymbol{\sigma}^0)$ as a mixture density network (Bishop, 1994), which is basically a universal approximator for conditional densities (see section 7.11.3). Applying Bayes' theorem again, the posterior distribution is

$$p(\mathbf{U}|\boldsymbol{\Sigma}^0) \propto p(\mathbf{U}) \prod_i \frac{p(\mathbf{u}_i|\boldsymbol{\sigma}_i^0)}{p(\mathbf{u}_i)}.$$

Given a backscatter field $\boldsymbol{\Sigma}^0$, the corresponding wind field is determined by MAP: a mode of $p(\mathbf{U}|\boldsymbol{\Sigma}^0)$ is found using a conjugate gradients method.

The fact that this inverse problem can be factorised into independent mapping inversion problems (see section 6.4.1) and the quasicontinuous dependence of the wind on the space coordinates make this problem amenable to the technique described in chapter 7.

6.2.4.3 Computerised tomography

The aim of computerised tomography (Herman, 1980) is to reconstruct the spatially varying absorption coefficients within a medium (e.g. the human body) from measurements of intensity decays of X-rays sent through the medium. Typically, X-rays are sent between a point source and a point receiver which counts the number of photons not absorbed by the medium, thus giving an indication of the integrated attenuation coefficient along that particular ray path (fig. 6.2). Repeating the measurement for many different ray paths, conveniently sampling the medium, the spatial structure of the attenuation coefficient can be inferred and so an image of the medium can be obtained.

The transmittance ρ_n (the probability of a photon of being transmitted) along the nth ray is given by:

$$\rho_n \stackrel{\text{def}}{=} \exp\left(-\int_{R_n} \mathbf{m}(\mathbf{x}(s_n))\, ds_n\right)$$

where:

- $\mathbf{m}(\mathbf{x})$ is the linear attenuation coefficient at point $\mathbf{x}$ and corresponds to the probability per unit length of path of a photon arriving at $\mathbf{x}$ being absorbed.
- R_n is the ray path, identified by the coordinates of the X-ray source and its shooting angle (ray paths of X-rays through an animal body can be assimilated to straight lines with an excellent approximation).
- ds_n is the element of length along the ray path.
- $\mathbf{x}(s_n)$ is the current point considered in the line integral along the ray (in Cartesian or spherical coordinates).

[5] Papers and software can be found in the NEUROSAT project web page at `http://www.ncrg.aston.ac.uk/Projects/NEUROSAT`.

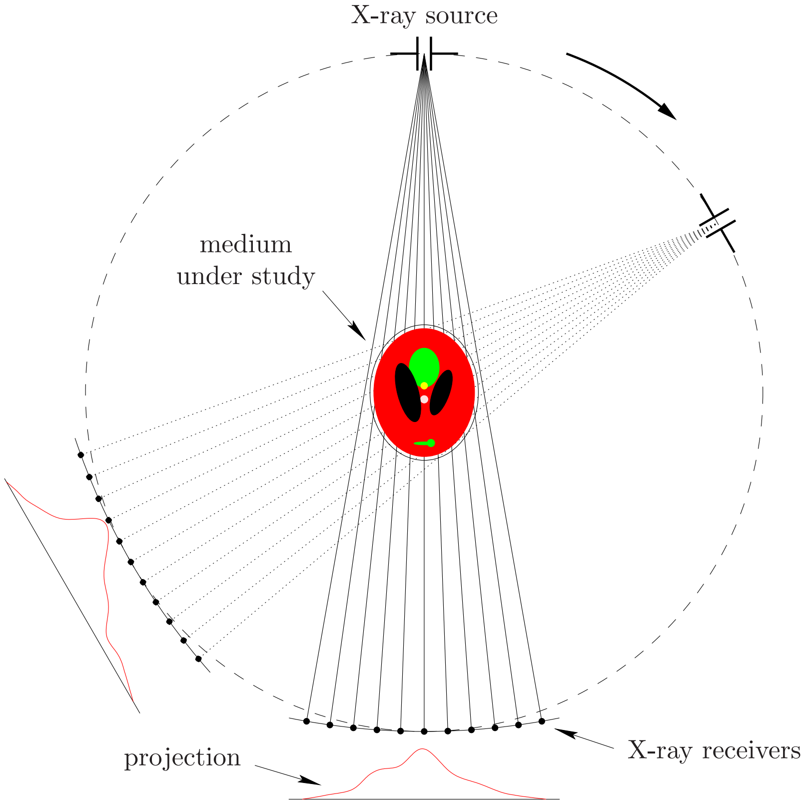


Figure 6.2: Setup for 2D X-ray tomography. A source sends a beam of X-rays through the object under study. Each individual X-ray is attenuated differently according to its path through the object. The X-rays are measured by an array of receivers, thus providing with a projection of the object. By rotating the source or having several sources surrounding the object we obtain several projections. Reconstructing the object density from these projections is the inverse problem of tomography.

Defining the data

$$d_n \stackrel{\text{def}}{=} -\ln \rho_n = \int_{R_n} \mathbf{m}(\mathbf{x}(s_n))\, ds_n \tag{6.7}$$

gives a linear relation between the data d_n and the unknown function $\mathbf{m}(\mathbf{x})$. Eq. (6.7) is the Radon transform of the function $\mathbf{m}(\mathbf{x})$, so that the tomography problem is the problem of inverting the Radon transform.

Thus, here the model parameters are the attenuation $\mathbf{m}(\mathbf{x})$ in the continuous case, or $\mathbf{m} = (m_{ijk})$ in a discretised version, and the observed parameters are the measured log-transmittance d_n. Given $\mathbf{m}(\mathbf{x})$, the forward problem is solved by the linear equation (6.7).

Similar problems appear in non-destructive testing and in geophysics. For example, in *geophysical acoustic tomography* the aim is to compute the acoustic structure of a region inside the Earth from seismic measurements. This allows, for example, to detect gas or oil deposits or to determine the radius of the Earth's metallic core. Acoustic waves are generated by sources at different positions inside a borehole and the travel times of the first wave front to receivers located in other boreholes around the region under study are recorded. The main difference with X-ray tomography is that the ray paths are not straight (they depend on the medium structure and are diffracted and reflected at boundaries), which makes the forward problem nonlinear. Acoustic tomography is an inverse scattering problem, in which one wants to determine the shape or the location of an obstacle from measurements of waves scattered by the obstacle.

Unlike example 6.2.4.2, this inverse problem does not factorise into independent mapping inversion problems and thus is not approachable by the technique of chapter 7.

6.3 Inverse problems vs Bayesian analysis in general

In Bayesian analysis in general, a parametric model is a function $p(\mathbf{x}; \mathbf{\Theta})$ where $\mathbf{x}$ are the variables of interest in the problem and $\mathbf{\Theta}$ the parameters, which identify the model. One is interested in inferences about both

Bayesian inverse problem theory	Latent variable models
Data space $\mathcal{D}$	Observed or data space $\mathcal{T}$
Model space $\mathcal{M}$	Latent space $\mathcal{X}$
Prior distribution of models $\rho_{\mathcal{M}}(\mathbf{m})$	Prior distribution in latent space $p(\mathbf{x})$
Forward mapping $\mathbf{g} : \mathcal{M} \to \mathcal{D}$	Mapping from latent space onto data space $\mathbf{f} : \mathcal{X} \to \mathcal{T}$
Uncertainty in the forward mapping $\theta(\mathbf{d}\|\mathbf{m}) = f(\mathbf{d} - \mathbf{g}(\mathbf{m}))$	Noise model $p(\mathbf{t}\|\mathbf{x}) = p(\mathbf{t}\|\mathbf{f}(\mathbf{x}))$

Table 6.1: Formal correspondence between continuous latent variable models and Bayesian inverse problem theory.

the parameters and the data, e.g. prediction via conditional distributions $p(x_2|x_1)$, etc. Bayesian inference is often approximated by fixing the parameters to a certain value given a sample $\{\mathbf{x}_n\}_{n=1}^N$ of the data, e.g. via maximum a posteriori (MAP) estimation:

$$\mathbf{\Theta}_{\text{MAP}} \stackrel{\text{def}}{=} \arg\max_{\mathbf{\Theta}} p(\mathbf{\Theta}|\mathbf{x}) = \arg\max_{\mathbf{\Theta}} p(\mathbf{x}|\mathbf{\Theta})p(\mathbf{\Theta}). \tag{6.8}$$

The model parameters $\mathbf{m}$ and the observed parameters $\mathbf{d}$ of inverse problem theory correspond to the parameters $\mathbf{\Theta}$ and the problem variables $\mathbf{x}$, respectively, of the Bayesian analysis in general. Bayesian inference about the model parameters $\mathbf{m}$, as shown in section 6.2.3, coincides with equation 6.8. But the emphasis is solely in inferences about the parameter estimates, i.e., how to find a single value of the model parameters that hopefully approximates well the physical reality. Thus, inverse problem theory is a *one-shot inversion problem*: use as much data as required to find a single value $\mathbf{m}_{\text{MAP}}$. Even if a second inversion is performed, we would expect the new inverse value of the model parameters to be close to the previous one—assuming that the system has not changed, i.e., assuming it is stationary or considering it in a fixed moment of time.

6.3.1 Inverse problems vs latent variable models

As an interesting example of the differences in interpretation between inverse problem theory and general Bayesian inference, let us consider continuous latent variable models as defined in chapter 2. As mentioned before, estimating the parameters of any probabilistic model can be seen as an inverse problem. But even when these parameters have been estimated and fixed, there is a formal parallelism between latent variable models and Bayesian inverse problem theory (in fact, all the estimation formulas for factor analysis mirror those of linear inverse problem theory). In latent variable models, we observe the data variables $\mathbf{t} \in \mathcal{T}$ and postulate a low-dimensional space $\mathcal{X}$ with a prior distribution $p(\mathbf{x})$, a mapping from latent space onto data space $\mathbf{f} : \mathcal{X} \to \mathcal{T}$ and a noise model in data space $p(\mathbf{t}|\mathbf{x}) = p(\mathbf{t}|\mathbf{f}(\mathbf{x}))$. Thus, the *model* here means the whole combined choice of the prior distribution in latent space, $p(\mathbf{x})$, the noise model, $p(\mathbf{t}|\mathbf{x})$, and the mapping from latent onto data space, $\mathbf{f}$, as well as the dimensionality of the latent space, L. And all these elements are equipped with parameters (collectively written as $\mathbf{\Theta}$) that are estimated from a sample in data space, $\{\mathbf{t}_n\}_{n=1}^N$. Table 6.1 summarises the formal correspondence between continuous latent variable models and Bayesian inverse problem theory.

The choice of model parameters to describe a system in inverse problem theory is not unique in general, and a particular choice of model parameters is a *parameterisation* of the system, or a *coordinate system*. Two different parameterisations are equivalent if they are related by a bijection. Physical knowledge of the inverse problem helps to choose the right parameterisation and the right forward model. However, what really matters is the combination of both the prior over the models and the forward mapping, $\rho_{\mathcal{M}}(\mathbf{m})\theta(\mathbf{d}|\mathbf{m})$, because this gives the solution to the Bayesian inversion. The same happens in latent variable models: what matters is the density in observed space $p(\mathbf{t})$, which is the only observable of the problem, rather than the particular conceptualisation (latent space plus prior plus mapping) that we choose. Thus, we are reasonably free to choose a simple prior in latent space if we can have a universal approximator as mapping $\mathbf{f}$, so that a large class of $p(\mathbf{t})$ can be constructed. Of course, this does not preclude using specific functional forms of the mapping and distributions if the knowledge about the problem suggests so.

We said earlier that inverse problem theory is a one-shot problem in that given a data set one inverts the forward mapping once to obtain a unique model. In continuous latent variable models, the latent variables are interpreted as *state variables*, which can take any value in their domain and that determine the observed variables of the system (up to noise). That is, when the system is in the state $\mathbf{x}$, the observed data is $\mathbf{f}(\mathbf{x})$ (plus the noise). The model remains the same (same parameters $\boldsymbol{\Theta}$, same prior distribution, etc.) but the marginal distribution of the latent and observed variables $p(\mathbf{x}, \mathbf{t})$ can be used to make inferences about $\mathbf{t}$ given $\mathbf{x}$ (forward problem, related to the deterministic and now estimated mapping $\mathbf{f} : \mathcal{X} \to \mathcal{T}$) and about $\mathbf{x}$ given $\mathbf{t}$ (inverse problem, or dimensionality reduction), for different values of $\mathbf{x}$ and $\mathbf{t}$. Thus, once the latent variable model has been fixed (which requires estimating any parameters it may contain, given a sample in data space) it may be applied any number of times to different observed data and give completely different posterior distributions in latent space, $p(\mathbf{x}|\mathbf{t})$—unlike in inverse problem theory, where different data sets are expected to correspond to the same model.

The particular case of independent component analysis (ICA), discussed in section 2.6.3, cannot be considered as an inverse problem because we do not have a forward model to invert. Even though we are looking for the inverse of the mixing matrix $\boldsymbol{\Lambda}$ (so that we can obtain the sources $\mathbf{x}$ given the sensor outputs $\mathbf{t}$), $\boldsymbol{\Lambda}$ is unknown. ICA finds a particular linear transformation $\mathbf{A}$ and a nonlinear function f that make the sources independent, but we do not either invert a function (the linear transformation $\boldsymbol{\Lambda}$) or estimate an inverse from input-output data ($\{\mathbf{x}_n, \mathbf{t}_n\}_{n=1}^N$).

6.4 Mapping inversion

Consider a function[6] $\mathbf{f}$ between sets $\mathcal{X}$ and $\mathcal{Y}$ (usually subsets of $\mathbb{R}^D$):

$$\begin{aligned} \mathbf{f} : \quad \mathcal{X} \quad &\longrightarrow \quad \mathcal{Y} \\ \mathbf{x} \quad &\mapsto \quad \mathbf{y} = \mathbf{f}(\mathbf{x}). \end{aligned}$$

Mapping inversion is the problem of computing the inverse $\mathbf{x}$ of any $\mathbf{y} \in \mathcal{Y}$:

$$\begin{aligned} \mathbf{f}^{-1} : \quad \mathcal{Y} \quad &\longrightarrow \quad \mathcal{X} \\ \mathbf{y} \quad &\mapsto \quad \mathbf{x} = \mathbf{f}^{-1}(\mathbf{y}). \end{aligned}$$

$\mathbf{f}^{-1}$ may not be a well-defined function: for some $\mathbf{y} \in \mathcal{Y}$, it may not exist or may not be unique. When $\mathbf{f}^{-1}$ is to be determined from a training set of pairs $\{(\mathbf{y}_n, \mathbf{x}_n)\}_{n=1}^N$, perhaps obtained by sampling $\mathcal{X}$ and applying a known function $\mathbf{f}$, the problem is indistinguishable from **mapping approximation** from data: *given a collection of input-output pairs, construct a mapping that best transforms the inputs into the outputs.*

A **universal mapping approximator** (UMA) for a given class of functions $\mathcal{F}$ from $\mathbb{R}^L$ to $\mathbb{R}^D$ is a set $\mathcal{U}$ of functions which contains functions arbitrarily close (in the squared Euclidean distance sense, for definiteness) to any function in $\mathcal{F}$, i.e., any function in $\mathcal{F}$ can be approximated as accurately as desired by a function in $\mathcal{U}$. For example, the class of multilayer perceptrons with one or more layers of hidden units with sigmoidal activation function or the class of Gaussian radial basis function networks are universal approximators for continuous functions in a compact set of $\mathbb{R}^D$ (see Scarselli and Tsoi, 1998 for a review). The functions in $\mathcal{U}$ will usually be parametric and the optimal parameter values can be found using a learning algorithm. There are important issues in statistical mapping approximation, like the existence of local minima of the error function, the reachability of the global minimum and the generalisation to unseen data. But for our purposes in this last part of the thesis what matters is that several kinds of UMAs exist, in particular the multilayer perceptron (MLP), for which practical training algorithms exist, like backpropagation.

A multivalued mapping assigns several images to the same domain point and is therefore not a function in the mathematical sense. Among other cases, multivalued mappings arise when computing the inverse of an injective mapping (i.e., a mapping that maps different domain points onto a same image point)—a very common situation. That is, if the direct or forward mapping verifies $\mathbf{f}(\mathbf{x}_1) = \mathbf{f}(\mathbf{x}_2) = \mathbf{y}$ then both $\mathbf{x}_1$ and $\mathbf{x}_2$ are inverse values of $\mathbf{y}$: $\mathbf{f}^{-1}(\mathbf{y}) \supseteq \{\mathbf{x}_1, \mathbf{x}_2\}$. UMAs work well with univalued mappings but not with multivalued mappings. In chapter 7 we give a method for reconstruction of missing data that applies as a particular case to multivalued mappings, and we compare it to UMAs as well as other approaches for mapping approximation like vector quantisation (section 7.11.4) and conditional modelling (section 7.11.3).

6.4.1 Inverse problems vs mapping inversion

Mapping inversion is a different problem from that of inverse problem theory: in inverse problem theory, one is interested in obtaining a unique inverse point $\mathbf{x}$ which represents a model of a physical system—of which

[6]We use the term *function* in its strict mathematical sense: a correspondence that, given an element $\mathbf{x} \in \mathcal{X}$, assigns to it one and only one element $\mathbf{y} \in \mathcal{Y}$. We use the term *mapping* for a correspondence which may be multivalued (one-to-many).

we observed the values $\mathbf{y} \in \mathcal{Y}$. In mapping inversion, we want to obtain an inverse mapping $\mathbf{f}^{-1}$ that we can use many times to invert different values $\mathbf{y} \in \mathcal{Y}$.

In section 6.2.3.8 we saw that the Bayesian inverse problem theory could be recast to solve such a mapping inversion problem. We rewrite eq. (6.6) with a simplification of the notation and decompose the model parameters $\mathbf{m}$ into parameters of *state* $\mathbf{s}$ and *experimental conditions* $\mathbf{c}_n$, $\mathbf{m} = (\mathbf{s}, \mathbf{c}_1, \dots, \mathbf{c}_N)$:

$$p(\mathbf{s}, \mathbf{c}_1, \dots, \mathbf{c}_N | \mathbf{d}_1, \dots, \mathbf{d}_N) \propto p(\mathbf{s}, \mathbf{c}_1, \dots, \mathbf{c}_N) \prod_{n=1}^{N} p(\mathbf{d}_n | \mathbf{s}, \mathbf{c}_n)$$

where $p(\mathbf{d}_n | \mathbf{s}, \mathbf{c}_n) = f(\mathbf{d}_n - \mathbf{g}(\mathbf{c}_n; \mathbf{s}))$ and $\mathbf{g}$ is the known forward mapping. This can be interpreted as a function $\mathbf{g}$ which maps $\mathbf{c}_n$ into $\mathbf{d}_n$ and has trainable parameters $\mathbf{s}$. However, $\{\mathbf{c}_n\}_{n=1}^N$ are *unknown parameters themselves, not data.* We are interested in constructing an inverse mapping $\mathbf{g}^{-1}$ given a data set consisting of pairs of values $\{(\mathbf{x}_n, \mathbf{y}_n)\}_{n=1}^N$ so that $\mathbf{y}_n = \mathbf{g}(\mathbf{x}_n)$ for a mapping $\mathbf{g}$ (not necessarily known). Clearly, in these terms the distinction between inverse and forward mapping disappears and the problem, as before, becomes a problem of mapping approximation.

Practitioners of inverse problem theory may object that by considering the forward mapping unknown we are throwing away all the physical information. But the theorems about universal approximation of mappings and about universal approximation of probability density functions support the fact that, given enough data, we can obtain (ideally) a good approximation of the joint density of the observed data and thus capture the information about the forward mapping too. This has the added flexibility of making inferences about any group of variables given any other group of variables by constructing the appropriate conditional distribution from the joint density—which includes both the forward and inverse mappings.

Two well-known examples of mapping inversion problems with one-to-many inverse mappings (often referred to as inverse problems in the literature) are the **inverse kinematics problem of manipulators** (the robot arm problem) (Atkeson, 1989) and the **acoustic-to-articulatory mapping problem** of speech (Schroeter and Sondhi, 1994). We describe them and apply to them our own algorithm later in this thesis.

Chapter 7

Sequential data reconstruction

7.1 Introduction

In this chapter we propose a method for sequential data reconstruction, which is also extendable to other kinds of data reconstruction problems. The problem of data reconstruction is very general, including as particular cases those of dimensionality reduction, mapping approximation and regression (we could also consider classification as a data reconstruction problem, but we stick to continuous variables in this thesis). All these problems require to compute some unknown values from other known values.

In section 7.2 we will define precisely the data reconstruction problem we are interested in. Let us introduce it here with an artificial example based on the binary star example of section 2.2.1. Imagine now that the astronomer has an apparatus that allows him to actually measure the (x, y) values of the position of the star, but that this apparatus is doubly imperfect: on the one hand, all measurements are affected by noise; on the other hand, the measured value x can from time to time be missing, and so can the value y other times. After having measured the position of the orbiting star a number of times, the astronomer obtains a collection of (x, y) values where some pairs have both x and y present, some have x missing and y present, some have y missing and x present, and some may have both x and y missing. The astronomer's problem is now to reconstruct as accurately as possible the collection of (x, y) values. To do so, he plots the collection as in figure 7.1 (left), where the filled dots indicate points for which both x and y are known; the vertical lines (drawn of finite length to avoid cluttering) indicate that x is known but not y, and so y can lie anywhere along that vertical line; similarly, the horizontal lines indicate that y is known but not x; and for the case where both x and y are missing we can draw nothing. As an aid, the actual location of the points where x, y or both are missing is drawn as a hollow dot (which is invisible for the astronomer). The task is then to find precisely those dots, e.g. for point 7 to find that $y_7 = y_{\mathrm{L}}$.

Imagine that the astronomer has an estimate of the orbit, obtained perhaps from a previous collection of measurements with no missing values, which we plot in figure 7.1 (right). The task is easier now, since

This chapter is partly based on references Carreira-Perpiñán (1999b, 2000b).

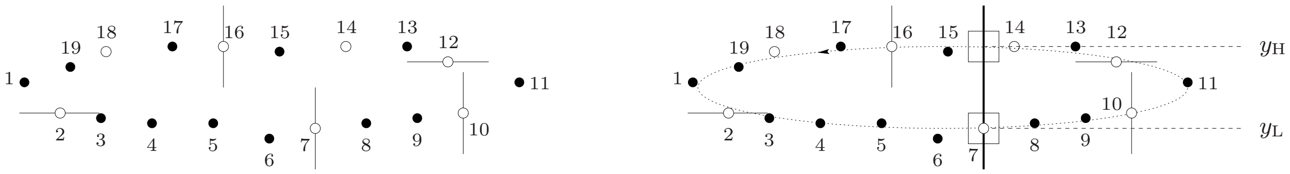


Figure 7.1: The example of the binary system orbit revisited. *Left*: the astronomer has measured a number of (x, y) positions of the orbiting star (black dots), but they are noisy and sometimes one or both of x or y are missing (hollow dots). The vertical line on point 7 indicates that x_7 is present but not y_7, which is somewhere on the line (of infinite length, although drawn finite to avoid cluttering). Likewise, the horizontal line on point 2 indicates that y_2 is present but not x_2, which is somewhere on the line. For points where both x and y are missing, as 14 and 18, we draw nothing. *Right*: superimposing the orbit helps to complete the missing values by finding where the lines meet the orbit and trying to avoid abrupt jumps in consecutive points (e.g. for y_7 choose y_{L} rather than y_{H}).

knowledge of the value of x helps us to determine the value of y and vice versa. For example, for point 7 the vertical line meets the ellipse at $y_7 = y_{\rm H}$ and $y_7 = y_{\rm L}$ (since the mapping $x \to y$ is multivalued). Clearly the latter value is the correct one, because it gives a smooth trajectory $(x_3, y_3), \dots, (x_9, y_9)$ along the lower branch of the ellipse. These two kinds of redundancy—the points lying on a low-dimensional manifold and each point lying close to its predecessor in the sequence—are the basis to solve the data reconstruction problem. The first kind of redundancy, which we will call *pointwise redundancy*, allows us to infer possible values for the missing variables given the present ones; we implement this by building offline a joint probability model of all variables and then deriving a functional relationship from the conditional distribution of the missing variables given the present ones. We implement the second kind of redundancy with a geometric constraint that can be minimised by dynamic programming.

This problem is more complex than that of multivariate regression because:

- We are not predicting one variable from the other one all the time: sometimes x is missing, sometimes y is missing and sometimes both are missing.
- The individual mappings $x \to y$ and $y \to x$ are multivalued: given a value of x, there are (nearly always) two values of y and vice versa.

The rest of this chapter is organised as follows. Section 7.2 defines precisely the data reconstruction problem. Section 7.3 explains how to derive (multivalued) functional relationships from conditional distributions. Section 7.5 explains how to use prior information to constrain the multivalued mappings thus derived. Finding the optimal reconstruction consistent with the multivalued maps and the constraints is described in section 7.6. The remaining sections deal with other topics: a probabilistic interpretation using the Markov assumption (section 7.7), an investigation of the computational complexity of the method (section 7.8), a discussion of its weak points (section 7.9) and of possible applications (section 7.10) and a review of related work (section 7.11). Experimental results are given separately in chapters 9 and 10.

7.2 Definition of the problem of data reconstruction

Generally, we define the problem of data reconstruction as follows.

Definition 7.2.1 (The problem of data reconstruction). Given a data set $\{\mathbf{t}_n\}_{n=1}^N$ where part of the data are missing, reconstruct the whole data set to minimise an error criterion.

Let us examine in detail the elements of this definition.

7.2.1 The data set

The *data set* must have some structure in order that reconstruction be possible, i.e., there must be some dependencies between the different vectors in the set that give rise to redundancy. A typical example is sequential data in which consecutive vectors are close to each other (note that not all sequences satisfy this). Such data can be considered as the result of sampling in time a variable that is a continuous function of the time. We can generalise this notion as follows. Assume $\{\mathbf{t}_n\}_{n=1}^N$ are samples from a continuous function $\mathfrak{F}$ of an independent variable $\mathbf{z}$ at points $\{\mathbf{z}_n\}_{n=1}^N$. We call $\mathbf{z}$ the *experimental conditions*; $\mathbf{z}$ can be the time (when the sample was taken), the position in space (where it was taken), etc. Thus, $\mathbf{t} = \mathfrak{F}(\mathbf{z})$ is the sample point obtained at condition $\mathbf{z}$. We observe $\mathbf{t}$ but not necessarily $\mathbf{z}$. Table 7.1 gives some examples. $\mathfrak{F}$ gives to the collection $\{\mathbf{t}_n\}_{n=1}^N$ the structure or redundancy that allows the reconstruction of missing data.

We now have three dimensionalities: the dimensionality of the space $\mathcal{T}$ of the $\mathbf{t}$ vectors, D; the intrinsic dimensionality of the manifold $\mathcal{M}$ where the $\mathbf{t}$ vectors live, L; and the dimensionality of the variable $\mathbf{z}$, C (we assume that the intrinsic dimensionality of the variable $\mathbf{z}$ coincides with C). It could seem that the intrinsic dimensionality of the $\mathbf{t}$ vectors cannot be larger than that of the $\mathbf{z}$ vectors, since $\mathbf{t}$ is a (continuous) function of $\mathbf{z}$; but L can indeed be larger than C. Consider for example an ant that walks on the trunk of a tree ($L = 2$) and take $\mathcal{T}$ as the Euclidean space ($D = 3$) and $\mathbf{z}$ as the time ($C = 1$); a given trajectory of the ant will be one-dimensional, but the region that the ant is allowed to be on (the trunk) is two-dimensional and in principle we may find it anywhere in that two-dimensional region (either by taking a very long trajectory or by imagining many different trajectories).

Summarising: $\mathcal{T}$ is the space on which we observe data of D dimensions; these data are constrained to live on an L-dimensional manifold $\mathcal{M}$ of $\mathcal{T}$; and we measure a batch of data $\{\mathbf{t}_n\}_{n=1}^N$ under experimental conditions that give them an apparent dimensionality C. Figure 7.2 illustrates the point.

Examples of problem	Experimental conditions $\mathbf{z}$	Observed variables $\mathbf{t}$
Trajectory of a mobile point	time, 1D	spatial coordinates, 3D (Cartesian, spherical)
Spoken utterance	time, 1D	speech feature vector, $\approx$13D (e.g. LSPs, MFCCs, PLPs)
Wind field on the ocean surface	spatial coordinates, 2D	wind velocity vector, 2D
Colour image	spatial coordinates, 2D	RGB level, 3D
Greyscale image	spatial coordinates, 2D	grey level, 1D

Table 7.1: Experimental conditions $\mathbf{z}$ and observed variables $\mathbf{t}$ for several examples of problems.

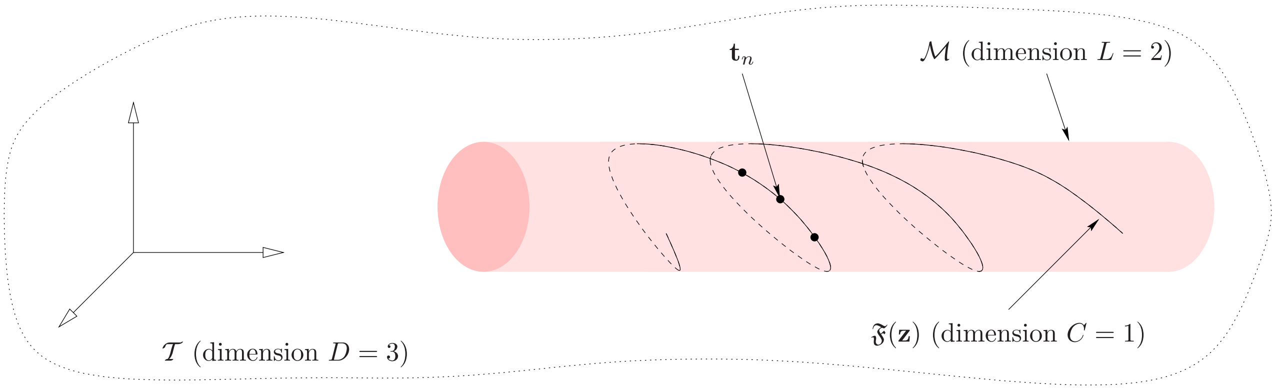


Figure 7.2: Dimensionalities involved in data reconstruction. The data, measured in a D-dimensional space $\mathcal{T}$, live in an L-dimensional manifold $\mathcal{M}$. A particular data set $\mathfrak{F}(\mathbf{z})$ of $\mathcal{M}$ has a dimensionality C equal to that of the experimental conditions $\mathbf{z}$. The dimensionalities verify $C \leq L \leq D$.

We will concentrate on the case where $D > 1$, since the whole topic of this dissertation is the analysis of multidimensional data. The case $D = 1$ (e.g. that of the "greyscale image problem" in table 7.1) does not allow to look at the relationship between variables, since there is only one ($\mathbf{t} = t_1$) and therefore we cannot make use of the redundancy derived from a low-dimensional representation.

In the rest of this thesis we will assume sequential data unless otherwise noted, i.e., $\mathbf{z}$ is the time or some other one-dimensional variable, although the treatment can be generalised to any dimensionality C. We will note $\{\mathbf{t}^{(n)}\}_{n=1}^N$ a sequential data set, where n indicates a temporal order in the data, reserving the notation $\{\mathbf{t}_n\}_{n=1}^N$ for data sets which need not have any sequential order (in which case n is just a label).

7.2.2 The pattern of missing data

In definition 7.2.1 we stated that *part of the data are missing.* This means that some of the ND variables $\{t_{nd}\}_{n,d=1}^{N,D}$ have missing values. We say that the value of a given scalar variable t_{nd} for certain $n \in \{1, \dots, N\}$ and $d \in \{1, \dots, D\}$ is **present** if such value was measured, even if it is (slightly) noisy; otherwise, we say it is **missing**. Abusing the language, we will also speak of present (missing) variables to mean variables whose values are present (missing). The reasons for a value to be missing are multiple: the value was not measured; the value was measured but got lost, erased or corrupted; and so on, depending on the particular problem.

When the values are corrupted in various amounts rather than just being either missing or present, one could consider further categories of uncertainty in a value. This is a beneficial strategy for, e.g., recognition of occluded speech (see the discussion in section 7.10.6). However, for the purposes of data reconstruction, it is not clear what one should do with a value that is neither present (which must not be modified) nor missing (which must be filled in). Therefore we will stick to the present/missing dichotomy. We will also assume that each value has been classified as either present or missing, even though in some applications this task may not be trivial at all.

We can represent the pattern of missing data associated with the data set $\{\mathbf{t}_n\}_{n=1}^N$ with a matrix (or *mask*) $\mathbf{M} = (m_{nd})$ of $N \times D$ such that $m_{nd} = 1$ if the value of t_{nd} is present and $m_{nd} = 0$ otherwise. The matrix $\mathbf{M}$ acts then as a multiplicative mask on the complete data set, i.e., the data set where all values are present, as represented in figure 7.3. We obtain the problem of regression (or mapping approximation, or mapping

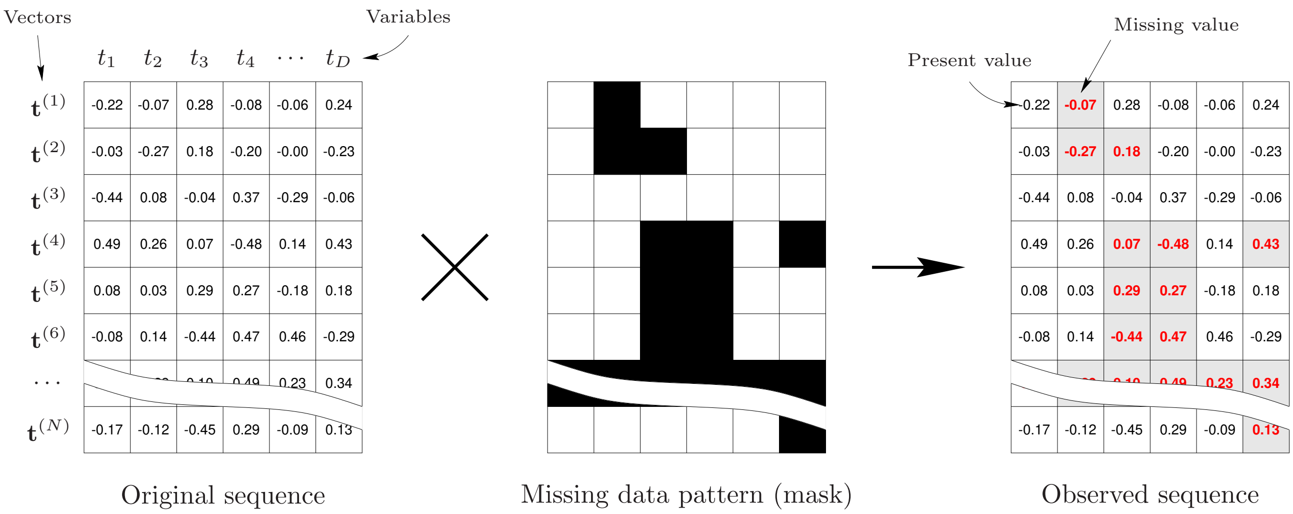


Figure 7.3: Schematic representation of the missing data. The black and white cells in the missing data pattern indicate missing and present values, respectively. The data set is notated as a sequence, but this is not necessary.

inversion) in the particular case where m_{nd} is independent of n (in which case the mask of fig. 7.3 has a columnar structure).

The algorithm described in this chapter is applicable to any pattern of missing data, irrespectively of why or how that pattern came about. However, if the missing data are not missing completely at random (see section 7.11.1.1), then any information about the mechanism of missing data should be taken into account if possible, since it may further constrain the values that the missing variables may take.

We will use the term **complete data (set)** to mean the data (set) as if no values were missing and **reconstructed data (set)** to mean the data (set) where the missing values have been filled in by a reconstruction algorithm.

7.2.3 Reconstruction of the whole data set

To reconstruct the whole data set means to find a single estimate of each missing value and is the objective pursued in this thesis. In some problems, e.g. classification in the presence of missing data, reconstruction may not be the optimal procedure; this is discussed in section 7.9.8.

7.2.4 Error criterion

The question of the error criterion has two aspects, related to the two dimensions of redundancy in the data:

- The intervariable, or pointwise, redundancy implies functional relationships between variables. If the relationship happens to be univalued, then defining it as the mean of the relevant conditional distribution implicitly minimises the quadratic error, as shown in section 7.3.3. If the relationship happens to be multivalued, the question of what error criterion is being minimised is more difficult. Intuitively, if we are able to select the correct branch (or sheet) of the data manifold and then define the functional relationship as the mean of the conditional distribution in that sheet, we would be minimising the quadratic error too; this interesting approach, akin to bump finding, is not the one we follow in this chapter, since we define functional relationships via the modes. Thus, we discuss briefly bump-finding in section 7.9.9 but leave it for future research.
- The interpoint redundancy implies some sort of continuous constraint between consecutive points. We are able to select with complete freedom this constraint, which then defines the error criterion, as shown in section 7.6.

7.2.5 Notation

We will use the following notation to select components (or variables) of a vector. If $\mathbf{t} \in \mathcal{T}$ is a D-dimensional real vector, $\mathbf{t} = (t_1, \dots, t_D)^T$, and $\mathcal{I} \subset \{1, \dots, D\}$ is a set of indices, then $\mathbf{t}_\mathcal{I}$ represents the vector composed of those components of $\mathbf{t}$ whose indices are in $\mathcal{I}$. This notation can be used in probabilistic expressions; for example, if $\mathcal{I}, \mathcal{J} \in \{1, \dots, D\}$ with $\mathcal{I} = \{1, 7, 3\}$ and $\mathcal{J} = \{2, 5\}$ then:

$$\begin{array}{rcc} \mathbf{t}_{\mathcal{I}} & \text{is} & (t_1, t_7, t_3) \\ \mathbf{t}_{\mathcal{J}} & \text{is} & (t_2, t_5) \\ p(\mathbf{t}_{\mathcal{I}}) & \text{is} & p(t_1, t_3, t_7) \\ p(\mathbf{t}_{\mathcal{J}}|\mathbf{t}_{\mathcal{I}}) & \text{is} & p(t_2, t_5|t_1, t_3, t_7) \\ p(\mathbf{t}_{\mathcal{I}}, \mathbf{t}_{\mathcal{J}}) & \text{is} & p(t_1, t_2, t_3, t_5, t_7). \end{array}$$

For definiteness we will consider that all sets of indices are ordered increasingly and contain no repeated elements, although this is not necessary. This notation is convenient to represent arbitrary patterns of missing data, where the present or missing variables at point n are indicated by sets $\mathcal{P}_n$ and $\mathcal{M}_n$, respectively, in which case $\mathcal{P}_n \cap \mathcal{M}_n = \varnothing$ and $\mathcal{P}_n \cap \mathcal{M}_n = \{1, \dots, D\}$ $\forall n = 1, \dots, N$. Abusing the notation, we may sometimes write $\mathbf{t}_{n,\mathcal{I}}$ or $\mathbf{t}^{(n)}_{\mathcal{I}}$ to mean $\mathbf{t}_{n,\mathcal{I}_n}$ or $\mathbf{t}^{(n)}_{\mathcal{I}^{(n)}}$, respectively.

7.2.6 Types of reconstruction

It will be convenient to define four types of reconstruction for later use according to the combinations of the following characteristics:

- The number of candidate reconstructions of a given entity that are provided: **single** or **multiple** reconstruction.
- The entity that is being reconstructed: **pointwise** (or **local**) for reconstruction of a vector $\mathbf{t}^{(n)}$ given information present in $\mathbf{t}^{(n)}$ only and **global** for reconstruction of the whole sequence or data set $\{\mathbf{t}^{(n)}\}_{n=1}^{N}$ given information present in it.

A method that provides single pointwise reconstruction can only provide single global reconstruction; standard regression and mapping approximation are examples such methods. But single global reconstruction can be attained by an appropriate combination of a collection of multiple pointwise reconstructions; the method proposed in this chapter is an example. The kind of reconstruction referred to in definition 7.2.1 is single global reconstruction. We briefly speak about multiple global reconstruction in section 7.7.1.

Any kind of reconstruction is based on a functional relationship of what is missing given what is present and we deal with this next.

7.3 Deriving functional relationships from conditional distributions

This section contains two central ideas. The first one is that one can define a functional relationship $\mathbf{x} \to \mathbf{y}$ ($\mathbf{y}$ as a function of $\mathbf{x}$) from the conditional distribution of $\mathbf{y}$ given $\mathbf{x}$ by picking representative points of this distribution. The second one is that underlying a multimodal conditional distribution there (often) is a multivalued functional relationship and that it is wrong to summarise such a distribution with its expected value. As a result, we will propose the use of all the modes of a conditional distribution to define a multivalued functional relationship and thus to define multiple pointwise reconstruction.

7.3.1 Informative, or sparse, distributions

Consider a real variable y which is a deterministic function of another real variable x, so that for each value x_0 of x there is a unique value $y_0 = f(x_0)$ of the variable y. In probability terms[1] we can say that the conditional probability of y given x is a Dirac delta function located at $y = f(x)$, $p(y|x) = \delta(y - f(x))$ (see fig. 7.4a).

Let us introduce some random variability, so that although the functional relationship is still there, it becomes blurred. Now, for each value x_0 of x there is a range of values of y located in an environment of $y_0 = f(x_0)$ which are possible, although the closer to $f(x_0)$ they are the more likely they are. In probability terms, for each value of x, we have some nondegenerate[2] distribution $p(y|x)$ peaking around $y = f(x)$ and which is the narrower the smaller the uncertainty is about y as depending on x (see fig. 7.4b).

If we were to predict a value for y given the value x_0 of x, it would be reasonable to choose $y = f(x_0)$ or some other value with high probability, which in view of the graph will fall close to $f(x_0)$. In this way we could derive a functional relationship between y and x, which we would expect to be close to the function $y = f(x)$.

[1]With an abuse of notation. More strictly we should write $p(y = y_1|x = x_0) = \delta(y_1 - f(x_0))$.

[2]By *degenerate* we mean a continuous distribution that assigns probability mass only to a zero-measure set, such as a delta distribution. That is, its density is zero almost everywhere.

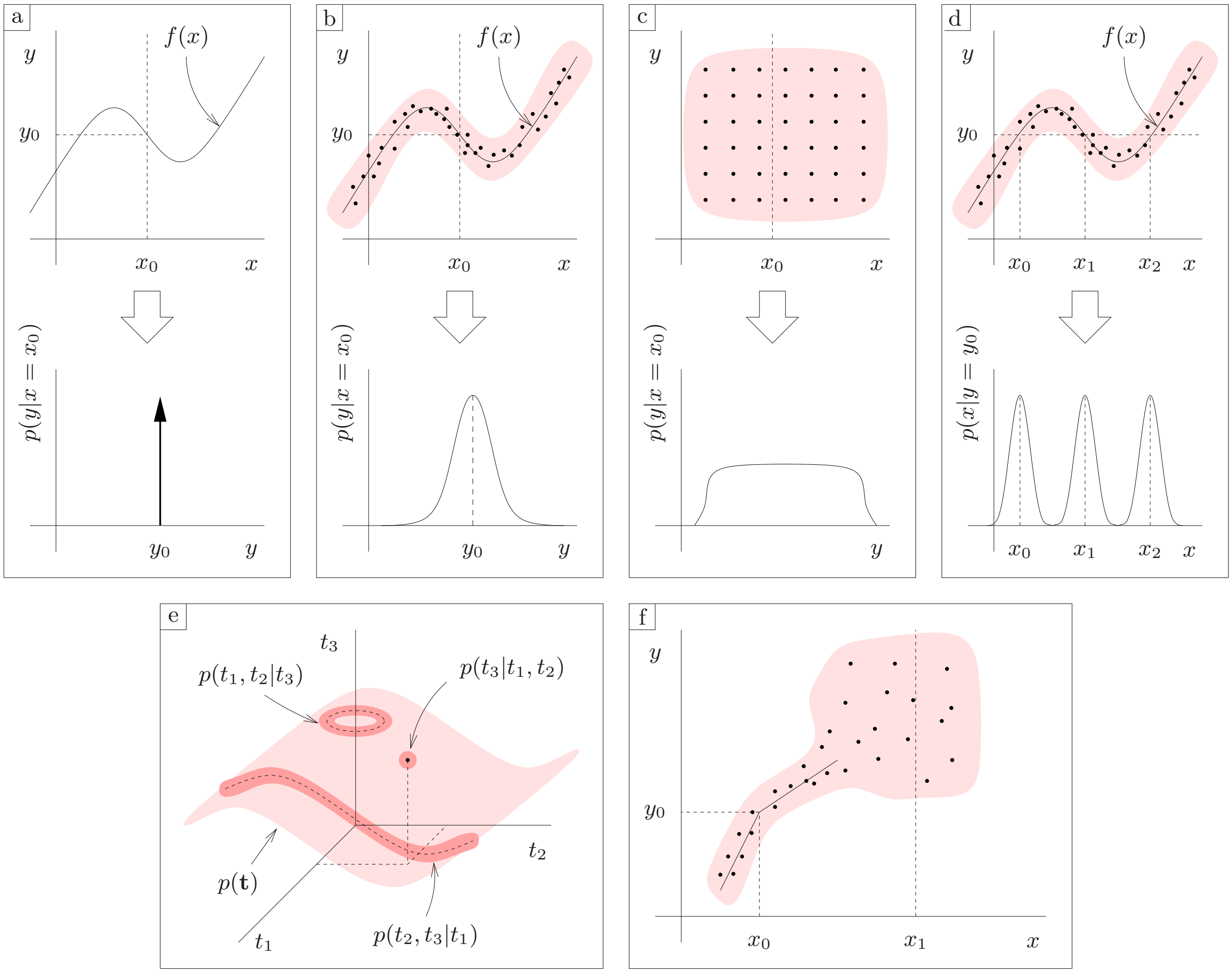


Figure 7.4: Different cases of a conditional distribution in data spaces of two and three dimensions. The shaded areas represent probability density and the black dots a sample typical from that density. In the one-dimensional graphs of $p(y|x)$ and $p(x|y)$, graphs (a) to (e), the densities are not drawn at the same scale. (a) Noiseless, univalued mapping $x \to y$ from $p(y|x)$. (b) Noisy, univalued mapping $x \to y$ from $p(y|x)$. (c) No functional relationship $x \to y$ because $p(y|x)$ is very flat. (d) Noisy, multivalued mapping $y \to x$ from $p(x|y)$. (e) Various cases of maps in three dimensions; $p(\mathbf{t})$ extends over $\mathbb{R}^3$, not just the shaded surface. (f) $p(y|x)$ is very informative around $x = x_0$ but little informative around $x = x_1$. Actual examples are given in figures 9.4–9.5.

Consider a situation where there is not a well-defined relationship of y as a function of x, that is, given a value x_0 of x, we do not expect any particular value of y to be much more likely than others in the domain of interest. Then the conditional distribution $y|x$ will have a rather flat aspect (see fig. 7.4c). We cannot derive a functional relationship in this case.

We conclude that sharply peaked conditional distributions convey more information about a functional relationship between the variables involved than flat-looking ones, to the point that we could construct a mapping $y = g(x)$ where g is a function of a distribution that picks a single *particularly informative* point of the domain of $p(y|x)$. Such an informative point could be the mean or the mode of the distribution.

Consider now the case of a multivalued functional relationship, such as that shown in fig. 7.4d for $y \to x$ from $p(x|y)$. Now we have a conditional distribution with several sharp peaks instead of only one. Although we cannot single out any of these peaks, we can still speak of a functional relationship in the sense that most of the domain of x can be ruled out because its associated probability mass is very low compared to that around the peaks. In this sense the distribution is still much more informative than the flat one.

Finally, consider the case of a multivariate distribution, as in fig. 7.4e, which shows how a functional dependence of a variable t_3 on variables t_1 and t_2, with some random variability, leads to a distribution mainly concentrated on a surface rather than a volume. A conditional distribution $p(t_3|t_1, t_2)$ is approximately defined on a region centred around $f(t_1, t_2)$, thus with dimension 0. A two-dimensional conditional distribution $p(t_1, t_2|t_3)$ is approximately defined on a one-dimensional region, represented by the dotted line (leading to a multivalued relationship). The same happens with the two-dimensional conditional distribution $p(t_2, t_3|t_1)$, approximately defined on the one-dimensional region of the dashed line.

In general, we say that a D-dimensional pdf $p(\mathbf{t})$ (possibly the distribution of $\mathbf{t}$ conditioned on the values of some other variables) is **informative** or **sparse** if the probability mass is concentrated around a low-dimensional manifold. Conversely, if the probability mass is spread over a D-dimensional region then we say that the pdf is **uninformative**. We thus state the principle that *highly informative distributions can be assimilated to (multivalued) functional relationships*.

The concept of sparseness is then a probabilistic extension of that of a low-dimensional manifold[3]. Thus, a D-dimensional pdf defined around a point/curve/surface is sparse for $D \geq 1/2/3$, respectively, and so on. Our definition of sparseness is vague, since the term "concentrated around" is relative (just how much probability mass and how near the low-dimensional manifold?), and thus lends itself to being quantitatively measured in various ways. We attempt one such measure in section 7.12.4, but for our purpose (to derive functional relationships from conditional distributions) this definition suffices.

A conditional distribution $p(y|x)$ may be informative for some values of x and uninformative for other values of x; figure 7.4e gives an example of this. Clearly, if we require that $p(y|x)$ be informative for any value of x, then the joint density $p(x, y)$ must be itself an informative distribution, since $p(y|x) \propto p(x, y)$.

For more generality, the pointwise predictive distribution could be taken as $p(\text{missing}|\text{present})p(\text{present})$ where $p(\text{present})$ is an extra prior distribution quantifying the accuracy to which the present variables have been measured. This prior could be taken as a factorised Gaussian with variances related to the experimental accuracy.

7.3.2 The modes as representative points of a distribution

Choosing representative points from a distribution means collapsing the domain of the continuous random variable in question into a finite number of points. This can only be done without loss of information if the distribution is a mixture of delta functions. In practice, a mixture of deltas will appear as a multimodal distribution, with more or less sharp peaks around the centres of the individual deltas, due to the noise. Intuitively, it would still be possible to recover those centres by identifying the peaks and then, for each peak, locating its centre, perhaps as its mode or mean. In our approach, we select all the modes of the distribution as representative points of it. This has three possible drawbacks, namely the potential existence of spurious modes; the potential coalescence of peaks associated with different centres into a single peak; and the question of choosing the mean of each local peak; we discuss them in section 7.9. For the time being, *let us assume that every mode or peak of a multimodal distribution is closely associated with one and only one centre of an underlying mixture of deltas*.

Our mode-finding algorithms for Gaussian mixtures given in chapter 8 can also compute error bars for each mode, thus estimating locally the error. The global error incurred by reducing the distribution to its modes can be quantified by a sparseness measure (section 7.12.4). However, we will not make use of any of these error estimates in our experiments.

7.3.3 Unimodal distributions: L_2-optimality of the expectation

Let us first consider the simple case of a unimodal distribution. In our earlier assumption of an underlying delta function, we need to pick a single point. This problem depends on what error criterion we want to minimise:

$$\text{Given } p(\mathbf{t}) \text{ with } \mathbf{t} \in \mathcal{T} \subseteq \mathbb{R}^D, \text{ select } \hat{\mathbf{t}} \in \mathcal{T} \text{ as } \hat{\mathbf{t}} = \arg\min_{\hat{\mathbf{t}} \in \mathcal{T}} \mathrm{E}_{p(\mathbf{t})}\left\{d(\mathbf{t}, \hat{\mathbf{t}})\right\} = \arg\min_{\hat{\mathbf{t}} \in \mathcal{T}} \int_{\mathcal{T}} p(\mathbf{t}) d(\mathbf{t}, \hat{\mathbf{t}})\, d\mathbf{t}.$$

That is, we pick the point that minimises the average distance (with respect to the distribution of $\mathbf{t}$) to every other point in the domain of $\mathbf{t}$. For the squared Euclidean distance, the most common case, the optimal point is the mean of the distribution:

$$\begin{aligned} &d(\mathbf{t}, \hat{\mathbf{t}}) \stackrel{\text{def}}{=} \left\|\mathbf{t} - \hat{\mathbf{t}}\right\|_2^2 \\ &\qquad \Rightarrow \mathrm{E}_{p(\mathbf{t})}\left\{\left\|\mathbf{t} - \hat{\mathbf{t}}\right\|_2^2\right\} = \mathrm{E}_{p(\mathbf{t})}\left\{\left\|\mathbf{t} - \boldsymbol{\mu} - (\hat{\mathbf{t}} - \boldsymbol{\mu})\right\|_2^2\right\} = \mathrm{tr}\left(\boldsymbol{\Sigma}\right) + \left\|\hat{\mathbf{t}} - \boldsymbol{\mu}\right\|_2^2 \leq \mathrm{tr}\left(\boldsymbol{\Sigma}\right) \ \forall \hat{\mathbf{t}} \in \mathcal{T} \end{aligned}$$

[3]The astute reader will sense the influence of latent variable models here.

where $\boldsymbol{\mu} \stackrel{\text{def}}{=} \mathrm{E}_{p(\mathbf{t})}\left\{\mathbf{t}\right\}$ and $\boldsymbol{\Sigma} \stackrel{\text{def}}{=} \mathrm{E}_{p(\mathbf{t})}\left\{(\mathbf{t}-\boldsymbol{\mu})(\mathbf{t}-\boldsymbol{\mu})^T\right\}$ are the mean and covariance matrix of $p(\mathbf{t})$, independent of $\hat{\mathbf{t}}$, and we have used $\operatorname{tr}(\mathbf{AB}) = \operatorname{tr}(\mathbf{BA})$. For symmetric distributions the mean and mode coincide, but for skewed distributions they differ. Therefore, for unimodal distributions we need only pick a single representative, according to the desired criterion.

For the L_1 norm, the optimal point is the median. To see it, consider first the case $D = 1$ (DeGroot (1986) gives a different proof):

$$
\begin{aligned}
\mathrm{E}_{p(t)}\left\{\left|t-\hat{t}\right|\right\} &= \int_{-\infty}^{\hat{t}} -(t-\hat{t})p(t)\,dt + \int_{\hat{t}}^{\infty} (t-\hat{t})p(t)\,dt \\
&= -\int_{-\infty}^{\hat{t}} tp(t)\,dt + \int_{\hat{t}}^{\infty} tp(t)\,dt + \hat{t}\left(\int_{-\infty}^{\hat{t}} p(t)\,dt - \int_{\hat{t}}^{\infty} p(t)\,dt\right) \\
&= \mu - 2\int_{-\infty}^{\hat{t}} tp(t)\,dt + \hat{t}(2P(\hat{t})-1)
\end{aligned}
$$

where P is the cdf of t. Differentiating with respect to $\hat{t}$ we obtain a minimum at the median $\hat{t} \stackrel{\text{def}}{=} P^{-1}\left(\frac{1}{2}\right)$:

$$
\begin{aligned}
&\frac{\partial}{\partial \hat{t}}\mathrm{E}_{p(t)}\left\{\left|t-\hat{t}\right|\right\} = -2\hat{t}p(\hat{t}) + 2P(\hat{t}) - 1 + 2\hat{t}p(\hat{t}) = 0 \Rightarrow \hat{t} = P^{-1}\left(\frac{1}{2}\right) \\
&\frac{\partial^2}{\partial \hat{t}^2}\mathrm{E}_{p(t)}\left\{\left|t-\hat{t}\right|\right\} = 2p(\hat{t}) > 0.
\end{aligned}
$$

This is the well known result that the median minimises the sum of absolute errors. It can be generalised to D dimensions for the L_1 distance: the optimal value is given by the componentwise medians, $\hat{t}_d \stackrel{\text{def}}{=} P_d^{-1}\left(\frac{1}{2}\right)$, where P_d is the marginal cdf of t_d. However, this generalisation does not have the natural interpretation of the onedimensional case, since it considers the coordinate directions as more important than arbitrary directions, which is not particularly useful.

A number of approaches exist that derive univalued mappings from the conditional distribution, usually via the mean. For example, for function approximation, Moody and Darken (1989) and Specht (1991) used the mean of the conditional distribution derived from a kernel joint density estimate. For a classification problem with missing data and for function approximation, Ghahramani (1994) and Ghahramani and Jordan (1994) used a mixture model of the joint density and a single value from the conditional distribution: the mean, a random sample or the component centroid with highest posterior probability[4]. (Tresp et al., 1995) used an approximation of the regression of response variables $\mathbf{y}$ as a function of predictor variables $\mathbf{x}$ when some of the latter are missing and the regression $\mathrm{E}\left\{\mathbf{y}|\mathbf{x}\right\}$ is given by a known mapping approximator $\mathbf{x} \xrightarrow{\mathbf{g}} \mathbf{y}$:

$$
\mathrm{E}\left\{\mathbf{y}|\mathbf{x}_{\mathcal{P}}\right\} = \int \mathrm{E}\left\{\mathbf{y}|\mathbf{x}_{\mathcal{P}},\mathbf{x}_{\mathcal{M}}\right\} p(\mathbf{x}_{\mathcal{M}}|\mathbf{x}_{\mathcal{P}})\,d\,\mathbf{x}_{\mathcal{M}} \approx \int \mathbf{g}(\mathbf{x})p(\mathbf{x}_{\mathcal{M}}|\mathbf{x}_{\mathcal{P}})\,d\,\mathbf{x}_{\mathcal{M}}.
$$

$p(\mathbf{x}_{\mathcal{M}}|\mathbf{x}_{\mathcal{P}})$ is obtained from a kernel or Gaussian mixture estimate of $p(\mathbf{x})$ and inserted in the right-hand side integral, which is then computed approximately to give $\mathrm{E}\left\{\mathbf{y}|\mathbf{x}_{\mathcal{P}}\right\}$. This is done to avoid estimating the joint distribution $p(\mathbf{x},\mathbf{y})$ in the first place and so gain some computational efficiency, but—apart from the approximations involved—it remains a conditional mean method.

7.3.4 The expectation of a multimodal distribution considered harmful

This statement should be obvious in itself but it is surprising how often a whole distribution is collapsed onto its mean without being sure that it is unimodal (or mildly multimodal). As fig. 7.5(left) shows, the mean of a multimodal distribution can lie on an area of low probability, thus being a highly unlikely representative of the distribution. Worse still, the mean may lie outside of the support of the random variable when such

[4] The "component centroid with highest posterior probability" is really a fast approximation to the global mode that amounts to vector quantisation, since all interaction between components is ignored. In fact, much research aims at using the "global mode" either for representing a conditional distribution by a single value or more generally to satisfy some optimality criterion for parameter estimation (e.g. maximum likelihood or least squares). Since often there are other modes or local optima, this "global mode" usually is a random mode (like the `rmode` method in section 9.1), depending on the implementation of the maximisation algorithm (typically on the starting point in iterative algorithms). Our mode finding algorithm of chapter 8 is guaranteed to find the global mode because it finds all the modes, at least for the mixtures that verify conjecture 8.1 (thus, in chapters 9–10, the `gmode` method is truly the global mode).

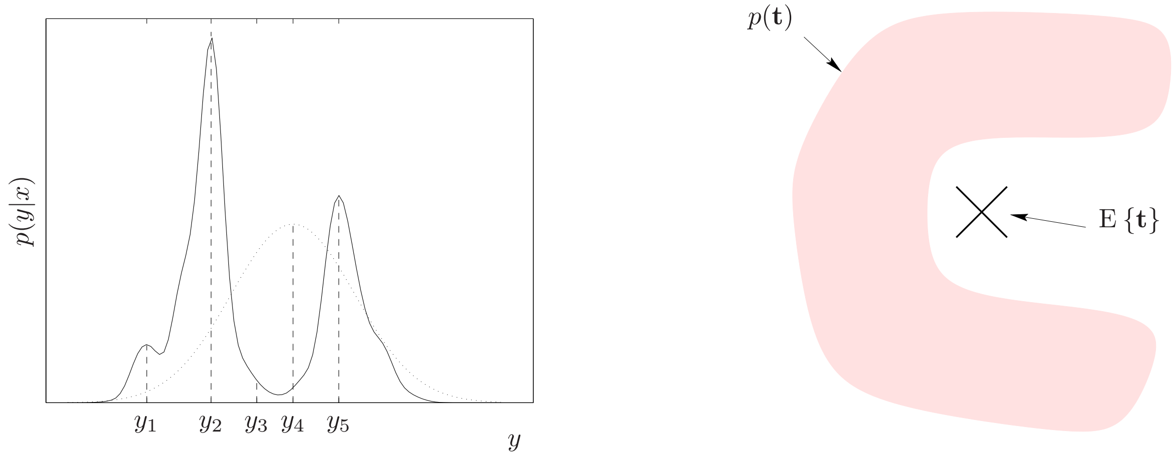


Figure 7.5: The expectation of a multimodal distribution considered harmful. *Left*: a unimodal (dotted line) and a multimodal conditional distribution (solid line). The vertical, dashed lines mark the modes and the means. *Right*: the expectation ($\times$) of a probability distribution on a nonconvex support (shaded area) may lie outside the support.

support is not a convex set[5] (since the mean is a convex sum itself), as shown in fig. 7.5(right); this fact has often been pointed out in the context of inverse problems and mapping inversion (e.g. Jordan, 1990; Jordan and Rumelhart, 1992). Unlike the mean, any mode must lie by definition inside the support of the random variable and have a locally high probability density.

There appear to be two reasons for the popularity[6] of the mean as a representative of a multimodal distribution:

- Its ease of calculation. Calculating the modes of a multivariate distribution is in general difficult because one does not know how many modes there are and because computing each one of them cannot be done analytically.

- The mean remains the optimal choice with respect to the L_2-norm for multimodal distributions, as long as one constrains the choice to be univalued.

But then how does one deal with a multivalued choice? In the absence of additional information, all we can do is to keep all the modes, since any of them is a likely representative of the distribution (a centre of the underlying mixture of deltas). But additional information can help to discard some of those modes; for example, if consecutive points in a sequence are constrained to lie close (as in the example of the introduction). For the time being, we keep all the modes, which implies defining a multivalued functional relationship if the distribution is a conditional distribution.

We conclude that for single pointwise reconstruction, the mean of the conditional distribution of the missing variables on the present ones is the L_2-optimal choice; and for multiple pointwise reconstruction, we propose the modes of that conditional distribution (but see section 7.9). For the same reason as the mean, one should not use a (weighted) average of the modes.

7.3.5 Underconstrained functions

The cases discussed so far concern functional relationships $\mathbf{y} = \mathbf{f}(\mathbf{x})$ where $\mathbf{x}$ and $\mathbf{y}$ can take at most a finite number of values. Theoretically it is possible to have a countably infinite number of values; e.g. $y = \arcsin x$ takes the values $\{m\pi\}_{m=-\infty}^{\infty}$ when $x = 0$. This situation is not a practical problem because in practice the

[5]This can happen even if the distribution is unimodal.

[6]People often consider the central value of a range the most likely or balanced even when there is absolutely no reason for it. Consider, for example, the *average man* of Quetelet in the XIX century, or the following more recent example. An archaeology journalist, David Keys, has put forward the theory that a natural catastrophe happened in 535 a.D. that was responsible for all sorts of environmental and social disruption worldwide, including climatic change, plagues, draughts and political turmoil. He pinpointed the catastrophe as a massive volcanic eruption between Sumatra and Java. Stratigraphic and carbon-14 measurements have indicated that the eruption could have taken place in any date from 1 to 1200 a.D. (David Keys, *Catastrophe*, Ballantine Books, New York, 2000; chapter 32 and p. 316, note 2). In a documentary on his book by the U.K. broadcaster Channel 4 Television, Mr. Keys expressed his satisfaction that 535 a.D. was almost right in the middle of that period.

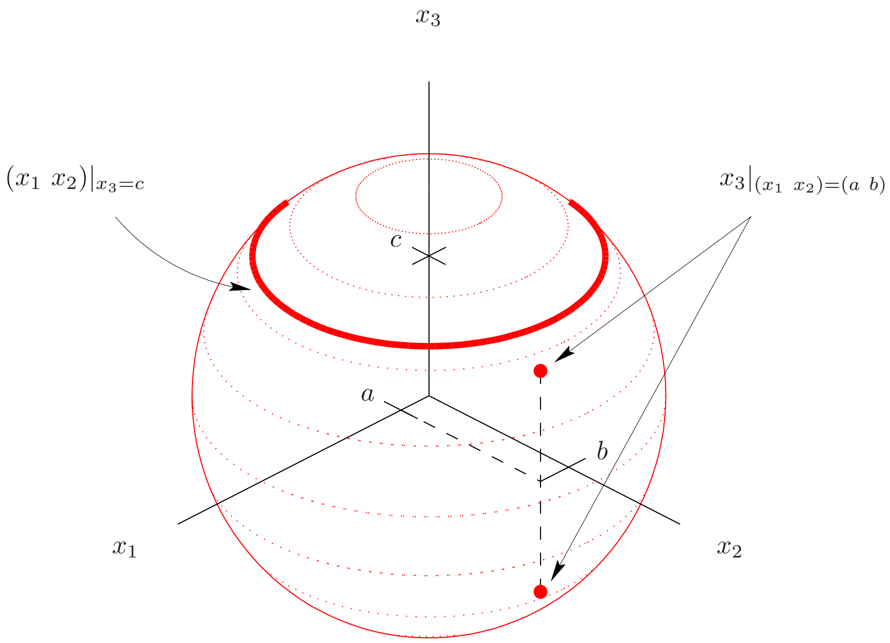


Figure 7.6: Geometric view of missing data reconstruction. The data lie in the spherical 2-manifold $x_1^2 + x_2^2 + x_3^2 = 1$. When only some of the variables are given, the rest can take values in the intersection of that sphere with the appropriate hyperplanes. This figure is similar to figure 7.4f, the difference being that there we used conditional probability distributions and here we use intersections of manifolds with hyperplanes parallel to the coordinate axes.

variables will have a limited support, particularly when the mapping is defined by a sample of data rather than by an equation[7]. A case more likely to appear in practice is when $\mathbf{y}$ is underconstrained given $\mathbf{x}$: $\mathbf{y}$ can take any value in a continuous manifold of dimension $Y \geq 1$. The case of countable values corresponds to a manifold of dimension 0. Geometrically, if the possible joint values of $\mathbf{x}$ and $\mathbf{y}$ span a manifold $\mathcal{M}$ of dimensionality L in $\mathbb{R}^D$, then the set of possible values for $\mathbf{y}$ given a fixed value $\mathbf{x}_0$ of $\mathbf{x}$ is the intersection of $\mathcal{M}$ with the hyperplane $\mathbf{x} = \mathbf{x}_0$. For example, consider the sphere $x_1^2+x_2^2+x_3^2 = 1$: $\mathbf{y} \stackrel{\text{def}}{=} (x_1\ x_2)^T$ as a function of $\mathbf{x} \stackrel{\text{def}}{=} x_3$ takes values continuously in the 1-manifold $x_1^2 + x_2^2 = 1$ (a circle) if $x_3 = 0$; $\mathbf{y} \stackrel{\text{def}}{=} x_3$ as a function of $\mathbf{x} \stackrel{\text{def}}{=} (x_1\ x_2)^T$ takes values discretely in the 0-manifold $\{-1, +1\}$ if $\mathbf{x} = \mathbf{0}$; etc. Figure 7.6 illustrates this point.

Therefore, from a geometric point of view the problem consists of determining the intersection between an arbitrary manifold $\mathcal{M}$ and several coordinate hyperplanes. This has two disadvantages:

- It requires solving nonlinear systems of equations.
- It disregards the noise, i.e., the stochastic variability of the data around and inside that manifold.

In the probabilistic framework the information about the data manifold $\mathcal{M}$ is embedded in the joint probability distribution $p(\mathbf{t})$ of the observed variables, noise is taken care of and the only mathematical operations needed are conditioning (therefore marginalising) and exhaustive mode search, which are computationally tractable for Gaussian mixtures. The choice of a Gaussian mixture as probabilistic model also eliminates the problem of underconstrained mappings, because the number of modes is finite if the Gaussian mixture is finite[8]. Of course, the underconstrained nature of a problem $\mathbf{y} = \mathbf{f}(\mathbf{x})$ remains. What this means is that the method is not affected by it, i.e., the method is providing a partial but robust solution to the problem. The solutions provided will be scattered over the manifold of $\mathbf{y}$. Therefore, the space of solutions is quantised and only an approximation can be given: the closest of this modes to the true solution. At least, the approach guarantees that the solutions obtained are valid in principle in that they lie in the solution manifold.

[7] It is still possible to have a countably infinite number of values even when all variables have a finite support. For example, $y = 1/\arcsin x$ for $y \in (0, 1)$ and $x \in [-1, 1]$ takes the values $y = \left\{\frac{1}{m\pi}\right\}_{m=1}^{\infty}$ when $x = 0$. But, again, such pathological cases are unlikely in practice when learning from data.

[8] This assertion depends on whether conjecture 8.1 is true, but it is probably safe to say that the number of modes is finite. This issue will be dealt with at length in chapter 8.

7.3.6 Universal mapping approximators versus density models

Consider the problem of function approximation: given a training set $\{(\mathbf{x}_n, \mathbf{y}_n)\}_{n=1}^N$, where $\mathbf{y}_n$ is a function of $\mathbf{x}_n$ corrupted with additive zero-mean noise. Then the univalued function $\mathbf{f}: \mathbf{x} \rightarrow \mathbf{y}$ that asymptotically minimises the squared error is the conditional mean of $\mathbf{y}$ given $\mathbf{x}$, as we can prove in a completely analogous way to section 7.3.3. Call $\boldsymbol{\mu}_{\mathbf{y}|\mathbf{x}} \stackrel{\text{def}}{=} \mathrm{E}_{p(\mathbf{y}|\mathbf{x})}\left\{\mathbf{y}\right\} = \mathrm{E}\left\{\mathbf{y}|\mathbf{x}\right\}$ and $\boldsymbol{\Sigma}_{\mathbf{y}|\mathbf{x}} \stackrel{\text{def}}{=} \mathrm{E}_{p(\mathbf{y}|\mathbf{x})}\left\{(\mathbf{y}-\boldsymbol{\mu}_{\mathbf{y}|\mathbf{x}})(\mathbf{y}-\boldsymbol{\mu}_{\mathbf{y}|\mathbf{x}})^T\right\}$ the conditional mean and covariance of $\mathbf{y}$ given $\mathbf{x}$, respectively. Then, for a given univalued function $\mathbf{f}: \mathbf{x} \rightarrow \mathbf{y}$:

$$\begin{aligned}
\mathrm{E}_{p(\mathbf{x},\mathbf{y})}\left\{\|\mathbf{y}-\mathbf{f}(\mathbf{x})\|_2^2\right\} &= \mathrm{E}_{p(\mathbf{x},\mathbf{y})}\left\{\|(\mathbf{y}-\boldsymbol{\mu}_{\mathbf{y}|\mathbf{x}}) + (\boldsymbol{\mu}_{\mathbf{y}|\mathbf{x}}-\mathbf{f}(\mathbf{x}))\|_2^2\right\} \\
&= \mathrm{E}_{p(\mathbf{x},\mathbf{y})}\left\{(\mathbf{y}-\boldsymbol{\mu}_{\mathbf{y}|\mathbf{x}})^T(\mathbf{y}-\boldsymbol{\mu}_{\mathbf{y}|\mathbf{x}})\right\} + \mathrm{E}_{p(\mathbf{x},\mathbf{y})}\left\{\|\boldsymbol{\mu}_{\mathbf{y}|\mathbf{x}}-\mathbf{f}(\mathbf{x})\|_2^2\right\} \\
&\quad + 2\,\mathrm{E}_{p(\mathbf{x},\mathbf{y})}\left\{(\mathbf{y}-\boldsymbol{\mu}_{\mathbf{y}|\mathbf{x}})^T(\boldsymbol{\mu}_{\mathbf{y}|\mathbf{x}}-\mathbf{f}(\mathbf{x}))\right\} \\
&= \mathrm{E}_{p(\mathbf{x})}\left\{\mathrm{E}_{p(\mathbf{y}|\mathbf{x})}\left\{\mathrm{tr}\left((\mathbf{y}-\boldsymbol{\mu}_{\mathbf{y}|\mathbf{x}})^T(\mathbf{y}-\boldsymbol{\mu}_{\mathbf{y}|\mathbf{x}})\right)\right\}\right\} + \mathrm{E}_{p(\mathbf{x})}\left\{\|\boldsymbol{\mu}_{\mathbf{y}|\mathbf{x}}-\mathbf{f}(\mathbf{x})\|_2^2\right\} \\
&\quad + 2\,\mathrm{E}_{p(\mathbf{x})}\left\{\mathrm{E}_{p(\mathbf{y}|\mathbf{x})}\left\{\mathbf{y}-\boldsymbol{\mu}_{\mathbf{y}|\mathbf{x}}\right\}^T(\boldsymbol{\mu}_{\mathbf{y}|\mathbf{x}}-\mathbf{f}(\mathbf{x}))\right\} \\
&= \mathrm{tr}\left(\mathrm{E}_{p(\mathbf{x})}\left\{\mathrm{E}_{p(\mathbf{y}|\mathbf{x})}\left\{(\mathbf{y}-\boldsymbol{\mu}_{\mathbf{y}|\mathbf{x}})(\mathbf{y}-\boldsymbol{\mu}_{\mathbf{y}|\mathbf{x}})^T\right\}\right\}\right) + \mathrm{E}_{p(\mathbf{x})}\left\{\|\boldsymbol{\mu}_{\mathbf{y}|\mathbf{x}}-\mathbf{f}(\mathbf{x})\|_2^2\right\} \\
&= \underbrace{\mathrm{tr}\left(\mathrm{E}_{p(\mathbf{x})}\left\{\boldsymbol{\Sigma}_{\mathbf{y}|\mathbf{x}}\right\}\right)}_{\text{(a)}} + \underbrace{\mathrm{E}_{p(\mathbf{x})}\left\{\|\boldsymbol{\mu}_{\mathbf{y}|\mathbf{x}}-\mathbf{f}(\mathbf{x})\|_2^2\right\}}_{\text{(b)}}.
\end{aligned}$$

Term (b) is minimised at $\mathbf{f}(\mathbf{x}) \stackrel{\text{def}}{=} \boldsymbol{\mu}_{\mathbf{y}|\mathbf{x}}$; term (a), the sum of expected conditional variances, is independent of $\mathbf{f}$ and represents the residual value of the error function at its global minimum.

Therefore, using a universal mapping approximator (introduced in section 6.4) will give us a function very close to the conditional mean (Bishop, 1995). For the purposes of this thesis, we will assume that a universal approximator and the conditional mean are equivalent. However, this assumes an ideal density model for the conditional distribution. In practice, both density models and function approximators are estimated from data and the resulting conditional mean may differ from the approximated function and from the asymptotic conditional mean.

If all is needed is a univalued mapping of a fixed group of predictor variables $\mathbf{x}$ onto a fixed group of response variables $\mathbf{y}$, then the use of universal approximators is appropriate, since they are trained to minimise the quadratic error and thus will converge to the mean. Universal approximators are faster and more efficient, since they model a function of L variables while density estimation models a function of $D + L$ variables. This is analogous to classification using a discriminant function or a probabilistic model: the former models a boundary while the latter models the whole space. The baseline is the dimensionality of the manifold being modelled, since there is where the curse of dimensionality has an effect.

If we need a multivalued mapping of a fixed group of predictor variables $\mathbf{x}$ onto a fixed group of response variables $\mathbf{y}$, then using a single universal approximator is not appropriate, since this will converge to the conditional mean which will be a compromise mapping half way through the branches[9] of the true mapping. It is conceivable to fit each one of these branches by a separate universal approximator, but this has a practical difficulty: to identify the branches or equivalently to separate the training data according to these branches. We discuss this approach in section 7.11.2.1.

If we need to model the mappings between many combinations of variables, universal approximators are not applicable in general because each combination requires a separate universal approximator and the total number of combinations grows exponentially[10], also requiring sufficient training data for each combination. Besides, a combination where few variables are mapped onto many (i.e., many variables are missing) will give rise to a severely unconstrained mapping, which cannot be summarised as a univalued mapping, as the universal approximator would do. Therefore the probabilistic approach to mapping approximation is the choice for multivalued maps and/or arbitrary combinations of predictor and response variables—in spite of the fact that density estimation requires more parameters than mapping approximation.

[9] Intuitively, we can define a *branch* of a forward function $\mathbf{g}$ as a submanifold of the trace of $\mathbf{g}$ where $\mathbf{g}$ is invertible (one-to-one).

[10] Writing a particular mapping as (present variables) $\rightarrow$ (missing variables), the total number of different mappings is equal to the total number of combinations of present variables. If the index d means the number of present variables, then there are $\sum_{d=0}^{D-1}\binom{D}{d} = 2^D - 1$ such mappings, as can be proven by induction (it is simply the cardinal of the set of parts of $\{1,\dots,D\}$ minus 1). This counts a pair of forward and inverse mappings as two different mappings (e.g. $(t_1\ t_3) \rightarrow (t_2\ t_4)$ and $(t_2\ t_4) \rightarrow (t_1\ t_3)$) but counts $(t_1\ t_3) \rightarrow t_2$, $(t_1\ t_3) \rightarrow t_4$ and $(t_1\ t_3) \rightarrow (t_2\ t_4)$ as the same mapping, and includes the mapping $\varnothing \rightarrow (t_1,\dots,t_D)$ but not the mapping $(t_1,\dots,t_D) \rightarrow \varnothing$.

7.3.7 Sampling the predictive distribution

Another method to pick representative points of a distribution is to sample from it. This makes sense when there are few present variables, so that the missing variables are so underdetermined that they can take values on a continuous manifold: in this case, the modes of a Gaussian mixture-based conditional distribution are effectively a particular sample from that manifold. Computationally, sampling is also more attractive than exhaustive mode finding, in particular for Gaussian mixtures. Finally, sampling may also give robustness to poor density models.

However, when the missing variables are constrained to take a finite number of values (as in mapping inversion), this does not make much sense, because—unless the conditional distribution is very sharply peaked around its modes—it will return values that by definition of sample are corrupted by noise: sometimes they can fall in areas far from the main probability mass body or ignore low-probability bumps (which may represent infrequent but legitimate reconstructions). A further, serious problem is how to set the number of samples to obtain, which will certainly locally underestimate or overestimate the true number of values that the missing variables can take. Missing a correct pointwise reconstruction or generating a wrong one may affect the global reconstruction, not just the local one, via the continuity constraint. The experiments of chapter 9 will show that this sampling strategy performs consistently worse than both the mean- and mode-based approaches and is thus not recommended[11].

7.3.8 Summary

We have argued that the conditional distribution of the missing variables given the present ones, if sparse enough, contains information to reconstruct the missing variables by picking representative points of it and thus defines a mapping from present to missing variables. This mapping is in general multivalued and therefore we have proposed to use the modes of the conditional distribution and not the mean as such representative points.

More generally, we reserve the terms **predictive distribution** for a distribution containing information about the missing variables (perhaps different from the conditional) and **candidate (pointwise) reconstructions** for the values used to fill in the missing variables (perhaps different from the modes). This is because most of the ideas in the rest of this chapter, in particular the constraint optimisation, would apply equally well to any pointwise reconstruction method.

7.4 Joint density model of the observed variables

For a generic missing data pattern we need to be able to obtain the conditional distribution for any combination of missing and present variables, which requires estimating a joint density model of the observed variables. The joint density embodies the relation of any subset of observed variables with any other subset of observed variables; all we require is to compute the appropriate conditional distribution, which in turn requires a marginalisation: $p(\mathbf{t}_{\mathcal{M}}|\mathbf{t}_{\mathcal{P}}) = p(\mathbf{t}_{\mathcal{M}}, \mathbf{t}_{\mathcal{P}})/p(\mathbf{t}_{\mathcal{P}})$. Therefore, we are free to choose the method by which we estimate the joint density as long as the estimator allows easy marginalisation. The density model should be estimated offline using a training set different from the one that is to be reconstructed. Typically, the training set will have no missing data, although even if it does, it is possible to train the model using an EM algorithm (e.g. Ghahramani and Jordan, 1994; McLachlan and Krishnan, 1997). This is because the presence of missing data is at the core of the EM algorithm, where missing variables are averaged over; in fact, it basically amounts to replacing $\mathbf{x}_n$ with $\mathbf{t}_{n,\mathcal{M}_n}$ and $\mathbf{t}_n$ with $\mathbf{t}_{n,\mathcal{P}_n}$ in the discussion of section 2.5. See also section 7.11.1.2.

An obvious choice for a density estimation method are latent variable models, since the pointwise redundancy implies an intrinsic low dimensionality of the observed variables (the distribution of the observed variables is sparse in the sense of section 7.3.1). Eq. (2.3) gives the joint density of the observed variables by marginalising over the latent variables. In particular, the GTM model is able to represent a broad class of densities, including multimodal distributions (although, as discussed in section 2.3.1, it is not a universal approximator for densities). The linear-normal models (factor analysis and PCA) are not interesting, for both the joint and conditional densities are normal, thus unimodal, and result in linear mappings.

We do not necessarily have to use a latent variable model. Other popular models include Gaussian mixtures (parametric) and kernel density estimation (nonparametric), both of which do have the property of universal

[11]However, it is recommended in the method of multiple imputation (reviewed in section 7.11.1.2), in which the aim is not reconstruction but inference about data sets with missing values where typically no (continuity) constraint is applicable and where the present variables do not strongly constrain the missing ones.

density approximation (Titterington et al., 1985; Scott, 1992). For all the specific latent variable models of chapter 2 (except ICA), as well as for mixtures of such latent variable models, the density of the observed variables has the form of a constrained Gaussian mixture: equations (2.16) for factor analysis, (2.24) for PCA, (2.37) for independent factor analysis and (2.43) for GTM. The constraint is imposed by the mapping from latent space onto observed space and the total number of parameters is relatively low because the noise model covariance is independent of the latent variables (section 2.3.1). Besides, owing to the axiom of local independence, the components are diagonal. We compared Gaussian mixtures and latent variable models in terms of the number of parameters in section 2.7.3.

Hereafter we will assume that the joint density of the observed variables has the form of a Gaussian mixture, whose parameters were estimated from a data sample in an unsupervised way. For a diagonal Gaussian mixture, computing conditional distributions requires little more than crossing out rows and columns; for a full-covariance Gaussian mixture, the conditional distribution requires inverting submatrices, but there is still an analytic solution (section 7.12.1). Chapter 8 gives algorithms to find all the modes of a Gaussian mixture, which will be our mechanism to obtain multiple candidate pointwise reconstructions.

7.5 Use of prior information to constrain multivalued mappings

So far we have exploited the redundancy between component variables of a given data point (via the conditional distribution) to constrain the possible values of the missing variables, but this can still result in multivalued mappings, as we have seen. In the absence of any additional information, the answer to the reconstruction problem would be those multivalued mappings. We now turn to the case of using extra information about the problem to constrain the possible values so that we obtain a unique global reconstruction of a data set.

First, we note that for some particular problems, knowledge of the measured values for a given data point can entail a further constraint apart from the one imposed by the conditional distribution. For example, in the problem of reconstruction of occluded speech (section 7.10.6), each point n corresponds to a frame of occluded speech and its energy is known. This energy can be taken as the energy of the unoccluded speech (our reconstruction goal) plus the energy of the corrupting signal (assuming additive corruption) and so it gives an upper bound for the energy of the unoccluded speech (which is also lower bounded by 0). In general, this type of constraint results in upper (A) and lower (B) bounds dependent on the present values for each missing component at each point in the data set: $A_{nd}(\mathbf{t}_{n\mathcal{P}}) \leq t_{nd} \leq B_{nd}(\mathbf{t}_{n\mathcal{P}})$ for $d \in \mathcal{M}$. A further type of constraint that depends only on the present values at a given point is given below for mapping inversion problems (see also section 7.11.1.1).

However, although for some points in the data set this kind of constraint may convert the multivariate mapping into a univariate one (if for each missing component all candidate values but one fall out of the bounds, or for mapping inversion problems only one candidate forward-maps well to the present values), in general the mapping will continue to be multivalued.

7.5.1 Continuity constraints

A more common (and powerful) source of additional information comes from the redundancy between data points and depends on the experimental conditions. The most usual such constraint is given by **continuity** of the observed variables as a function of the experimental conditions, $\mathbf{t} = \mathfrak{F}(\mathbf{z})$: nearby values of $\mathbf{z}$ produce nearby values of $\mathbf{t}$. In this case, typically $\mathbf{z}$ is a physical variable such as the time or space, and then $\mathbf{t}$ is a measured vector that depends continuously on it, resulting in a continuous trajectory or field, respectively. Consider the specific case of onedimensional $\mathbf{z} = z$ (e.g. the time): continuity then implies that consecutive points should be near each other according to a distance dependent on the problem, or "$d(\mathbf{t}_n, \mathbf{t}_{n+1})$ is small." To first order of approximation[12] we can reexpress a continuity constraint as a constraint on the derivative of the function $\mathfrak{F}$ (assuming such derivative exists): for points measured at consecutive instants z_n and z_{n+1}

$$\mathbf{t}_{n+1} - \mathbf{t}_n = \mathfrak{F}(z_{n+1}) - \mathfrak{F}(z_n) = \Delta z_n \mathfrak{F}'(z_n) + \mathcal{O}(\Delta z_n^2)$$

where $\Delta z_n \stackrel{\text{def}}{=} z_{n+1} - z_n$ is assumed small. Thus, the condition that consecutive points be near each other can be written as smallness of $\mathfrak{F}'(z)\Delta z$ and we can derive specific forms of the continuity constraint by numerically approximating the derivative by a finite difference scheme (Isaacson and Keller, 1966, chapter 6; Press et al.,

[12]In fact, by the mean-value theorem the relation is exact: $\mathfrak{F}(z_{n+1}) - \mathfrak{F}(z_n) = \Delta z_n \mathfrak{F}'(\xi)$ with $\xi \in [z_n, z_{n+1})$, but ξ is unknown. Any useful definition of constraint must rely only on the available variables, measured at $\{\mathbf{z}_n\}_{n=1}^N$.

1992, section 5.7) in terms of the available measures at $z_1, z_2, \ldots, z_N$ and then taking the (problem-dependent) norm of Δz times that scheme. For example, the forward difference scheme (accurate to first order)

$$\frac{\mathfrak{F}(z_{n+1}) - \mathfrak{F}(z_n)}{\Delta z_n}$$

gives $\|\mathbf{t}_{n+1} - \mathbf{t}_n\|$ and the central difference scheme (accurate to second order, for equispaced data: $z_{n+1} = z_n$ for all n)

$$\frac{\mathfrak{F}(z_{n+1}) - \mathfrak{F}(z_{n-1})}{2\Delta z}$$

gives $\frac{1}{2}\|\mathbf{t}_{n+1} - \mathbf{t}_{n-1}\|$, which translate as "$d(\mathbf{t}_n, \mathbf{t}_{n+1})$ be small" and "$\frac{1}{2}d(\mathbf{t}_{n-1}, \mathbf{t}_{n+1})$ be small," respectively.

A stronger constraint is **smoothness**: besides being continuous, the trajectory $\mathfrak{F}(z)$ cannot have high-curvature turns. Smoothness is usually defined through the second derivative and, again, by approximating it with finite differences, we can express it as a function of the measured points. For example, the forward difference scheme (again accurate to first order, for equispaced data)

$$\frac{\mathfrak{F}(z_{n+1}) - 2\mathfrak{F}(z_n) + \mathfrak{F}(z_{n-1})}{\Delta z^2}$$

gives $\|\mathbf{t}_{n+1} - 2\mathbf{t}_n + \mathbf{t}_{n-1}\|$, which translates as "$2d\left(\mathbf{t}_n, \frac{\mathbf{t}_{n-1}+\mathbf{t}_{n+1}}{2}\right)$ be small" (compare with eq. 6.4). And in general one can consider constraints of order p, of the form $\left\|\frac{d^p\mathfrak{F}}{dz^p}\right\|^2$ for some finite-difference approximation of the pth derivative.

This strategy of defining constraints based on the norm of finite difference approximations of derivatives can be readily generalised to multidimensional experimental conditions, although the resulting expressions are correspondingly more complex. For example, for continuity we obtain $\left\|\Delta\mathbf{z}^T\nabla\mathfrak{F}\right\| = \left\|\sum_{c=1}^C \Delta z_l \frac{\partial\mathfrak{F}}{\partial z_l}\right\|$ and we approximate $\nabla\mathfrak{F} = \left(\frac{\partial\mathfrak{F}}{\partial z_1}, \ldots, \frac{\partial\mathfrak{F}}{\partial z_C}\right)^T$. In two equispaced dimensions, a forward difference leads to the expression $\|\mathbf{t}_{n+1,m} - \mathbf{t}_{n,m} + \mathbf{t}_{n,m+1} - \mathbf{t}_{n,m}\|$ (where we write $\mathbf{t}_{n,m}$ instead of $\mathbf{t}_n$) and so on.

Another type of constraint that has been often found useful in inverse problems is a **quadratic** constraint $(\mathbf{t}_n - \mathbf{t}_0)^T\mathbf{Q}(\mathbf{t}_n - \mathbf{t}_0)$, which can be interpreted physically as an energy (potential, kinematic) or work in mechanical systems (e.g. in the acoustic-to-articulatory mapping, eq. (10.3)). In particular, $\|\mathbf{t}_n - \mathbf{t}_0\|^2$ corresponds to the potential energy of a Hooke oscillator with resting position at $\mathbf{t} = \mathbf{t}_0$ and restoring constant $k = 1$.

If instead of the L_2-norm one applies the squared L_2-norm to linear combinations of the data points, all the constraints described in this section result in quadratic forms on the variables $\{t_{nd}\}_{n,d=1}^{N,D}$.

7.5.2 Choice of distance

The constraints just discussed rely on a distance defined in the observed space. This distance depends necessarily on what the observed variables (physically) are, but typically will be the Euclidean or squared Euclidean distance. In this case, it is important to account for the effects of variable scaling (change of units): a separation of length 1 along one axis may mean a larger or shorter distance than a separation of length 1 along a different axis. This is particularly true when the observed variables have different units (length, mass, time, etc.) or different ranges. For example, in the acoustic-to-articulatory mapping application of chapter 10, some variables are a complex function of the speech acoustics while other variables are geometric parameters of the vocal tract. A convenient way to take into account the scale (or units) of each variable is to use a weighted distance, e.g. one based on the weighted squared Euclidean norm $\|\mathbf{t}\|_2^2 = \sum_{d=1}^D w_d t_d^2$, where $w_1, \ldots, w_D$ are positive weights. This is equivalent to rescaling each variable t_d by a weight $\sqrt{w_d}$. The weights depend on the (physical) nature of the observed variables and may or may not be related to the standard deviation or to the local noise of the observed variables (see fig. 2.8), so sphering the data may still require the use of a weighted distance.

Although the choice of distance should be dictated by the peculiarities of the reconstruction problem, some distances may have a computational advantage. For example, for distances satisfying the property that $d(\mathbf{t}_1, \mathbf{t}_2) = d(\mathbf{u}_1, \mathbf{u}_2) + d(\mathbf{v}_1, \mathbf{v}_2)$ if $\mathbf{t} = \binom{\mathbf{u}}{\mathbf{v}}$ then, for a constant missing data pattern ($\mathcal{M}_n = \mathcal{M}\ \forall n$) and using a global continuity constraint (see section 7.6.1)

$$\arg\min_{\left\{\mathbf{t}^{(n)}_{\mathcal{M}^{(n)}}\right\}_{n=1}^N} \sum_{n=1}^{N-1} d(\mathbf{t}^{(n)}, \mathbf{t}^{(n+1)}) = \arg\min_{\left\{\mathbf{t}^{(n)}_{\mathcal{M}^{(n)}}\right\}_{n=1}^N} \sum_{n=1}^{N-1} d\left(\mathbf{t}^{(n)}_{\mathcal{M}^{(n)}}, \mathbf{t}^{(n+1)}_{\mathcal{M}^{(n+1)}}\right)$$

since, for each n, $d\left(\mathbf{t}^{(n)}_{\mathcal{P}(n)}, \mathbf{t}^{(n+1)}_{\mathcal{P}(n+1)}\right)$ is constant, independent of $\mathcal{M}^{(n)}$. The L_2 norm does not satisfy this, but the square of the L_2 norm does.

Also, the global constraint optimisation is independent of additive terms or positive multiplicative factors in the distance because they preserve the order relation of the global constraint: if $\sum_{n=1}^{N-1} d(\mathbf{t}^{(n+1)}, \mathbf{t}^{(n)}) < \sum_{n=1}^{N-1} d(\hat{\mathbf{t}}^{(n+1)}, \hat{\mathbf{t}}^{(n)})$ then $\sum_{n=1}^{N-1} k_1 d(\mathbf{t}^{(n+1)}, \mathbf{t}^{(n)}) + k_2 < \sum_{n=1}^{N-1} k_1 d(\hat{\mathbf{t}}^{(n+1)}, \hat{\mathbf{t}}^{(n)}) + k_2$ if $k_1 > 0$.

7.5.3 Constraint by forward mapping

In the particular case where the reconstruction problem is a mapping inversion problem, we can use the known forward (direct) mapping as a further constraint. The forward mapping $\mathbf{g}$ maps the missing variables onto the present ones: $\mathbf{t}_{\mathcal{P}} = \mathbf{g}(\mathbf{t}_{\mathcal{M}})$, where $\mathcal{P}$ and $\mathcal{M}$ are independent of n (i.e., the same for all data points). Thus, given the values of $\mathbf{t}_{\mathcal{P}}$ and given a candidate reconstruction $(\mathbf{t}_{\mathcal{P}}\ \hat{\mathbf{t}}_{\mathcal{M}})$, such as for $\hat{\mathbf{t}}_{\mathcal{M}}$ being one of the modes of $p(\mathbf{t}_{\mathcal{M}}|\mathbf{t}_{\mathcal{P}})$, the distance (in observed space, as discussed earlier) between[13] $(\mathbf{t}_{\mathcal{P}}\ \hat{\mathbf{t}}_{\mathcal{M}})$ and $(\mathbf{g}(\hat{\mathbf{t}}_{\mathcal{M}})\ \hat{\mathbf{t}}_{\mathcal{M}})$ should be as small as possible.

In the ideal case where the procedure that provides candidate reconstructions (in this thesis, the modes of the conditional distribution) was perfect, this constraint would have no effect, since every $\hat{\mathbf{t}}_{\mathcal{M}}$ would exactly map onto $\mathbf{t}_{\mathcal{P}}$. In reality, correct reconstructions will give small, but nonzero, differences between $\mathbf{t}_{\mathcal{P}}$ and $\mathbf{g}(\hat{\mathbf{t}}_{\mathcal{M}})$, while incorrect reconstructions (such as spurious modes, see section 7.9.1) will generally give a much larger difference. Thus, the constraint by forward mapping can help to discard such incorrect reconstructions. Such constraint should be combined with the continuity constraint in the way explained in section 7.6.1.

7.5.4 Continuity of the modal mapping revisited

In section 2.9.2 and fig. 2.14 we showed that a mapping defined as the mean (of a conditional distribution) was always continuous, while a mapping defined as the global mode (or some other mode) was not necessarily continuous. This can also be seen in the discontinuous way in which the `gmode` method switches between branches in figures 9.8–9.9. We are now overcoming this problem by allowing a different mode to be picked at different points so that a continuity constraint is satisfied (e.g. in fig. 2.14 we would follow for all values of t either the mode near $x = 0$ or the mode near $x = 1$, avoiding the discontinuity at $t = \frac{1}{2}$).

7.6 Minimisation of constraints

The constraints introduced in the previous section are by definition local and generally take the form of the norm of a linear combination of a collection of neighbouring points around a given point. For example, $c_n = d(\mathbf{t}_n, \mathbf{t}_{n+1})$ as an approximation of $\left\|\Delta\mathbf{z}^T \nabla\mathfrak{F}\right\|_{\mathbf{z}_n}$.

7.6.1 Definition of global constraint

Here we derive from such local constraints a **global constraint** that involves the whole data set. The reason for doing this is to define an objective function depending on all missing variables $\{\mathbf{t}_{n,\mathcal{M}_n}\}_{n=1}^N$ and then find the combination of candidate pointwise reconstructions that minimises it. We will then have a single global reconstruction that should be a good approximation to the complete data set. The role of the global constraint in breaking the nonuniqueness of the reconstruction is analogous to that of Tikhonov regularising operators or stabilisers (Tikhonov and Arsenin, 1977) for ill-posed problems (such as inverse problems, chapter 6).

The obvious way to obtain the global constraint is to sum the local constraints for every point in the data set. As usual, we consider sequential data (one-dimensional experimental conditions). Then, if c_n is the local constraint at point n, we have[14] $\mathscr{C}\left(\{\mathbf{t}^{(n)}\}_{n=1}^N\right) \stackrel{\text{def}}{=} \sum_{n=1}^N c_n$. For example, if we use continuity constraints based on forward differences, $c_n \stackrel{\text{def}}{=} d(\mathbf{t}^{(n)}, \mathbf{t}^{(n+1)})$, and taking[15] $c_N \equiv 0$, then we have $\mathscr{C}\left(\{\mathbf{t}^{(n)}\}_{n=1}^N\right) = \sum_{n=1}^{N-1} d(\mathbf{t}^{(n)}, \mathbf{t}^{(n+1)})$. This expression is also the length of the polygonal trajectory passing

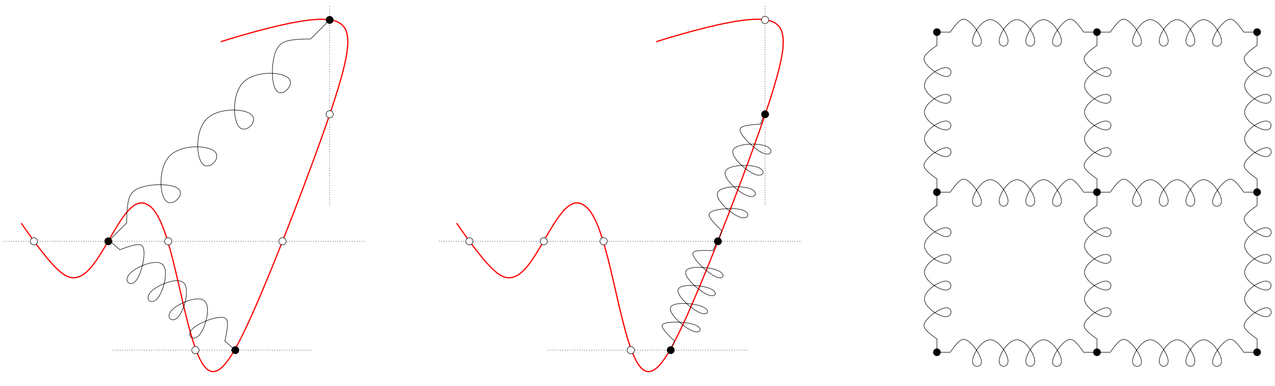
[13] We write "the distance between $(\mathbf{t}_{\mathcal{P}}\ \hat{\mathbf{t}}_{\mathcal{M}})$ and $(\mathbf{g}(\hat{\mathbf{t}}_{\mathcal{M}})\ \hat{\mathbf{t}}_{\mathcal{M}})$" rather than "the distance between $\mathbf{t}_{\mathcal{P}}$ and $\mathbf{g}(\hat{\mathbf{t}}_{\mathcal{M}})$" to emphasise that, strictly, the distance is defined in the observed space of variables $t_1, \dots, t_D$. However, in practice there is no difference.

[14] We write $\mathscr{C}$ as a function of $\{\mathbf{t}^{(n)}\}_{n=1}^N$ rather than of $\{\mathbf{t}^{(n)}_{\mathcal{M}(n)}\}_{n=1}^N$ for simplicity. The reader should bear in mind that the present variables $\{\mathbf{t}^{(n)}_{\mathcal{P}(n)}\}_{n=1}^N$ are fixed to the present values.

[15] Taking $c_N = d(\mathbf{t}^{(N)}, \mathbf{t}^{(1)})$ instead would account for a closed trajectory.

Constraint type	Local approximation	Global constraint $\mathscr{C}$ in distance form	Global constraint $\mathscr{C}$ in norm form
Continuity, $\mathscr{C}_1$	Forward difference	$\sum_{n=1}^{N-1} d(\mathbf{t}^{(n)}, \mathbf{t}^{(n+1)})$	$\sum_{n=1}^{N-1} \left\lVert \mathbf{t}^{(n)} - \mathbf{t}^{(n+1)} \right\rVert$
Continuity, $\mathscr{C}_2$	Central difference	$\frac{1}{2} \sum_{n=2}^{N-1} d(\mathbf{t}^{(n-1)}, \mathbf{t}^{(n+1)})$	$\frac{1}{2} \sum_{n=2}^{N-1} \left\lVert \mathbf{t}^{(n+1)} - \mathbf{t}^{(n-1)} \right\rVert$
Smoothness, $\mathscr{S}_1$	Forward difference	$2 \sum_{n=2}^{N-1} d\left(\mathbf{t}^{(n)}, \frac{\mathbf{t}^{(n+1)}+\mathbf{t}^{(n-1)}}{2}\right)$	$\sum_{n=2}^{N-1} \left\lVert \mathbf{t}^{(n+1)} - 2\mathbf{t}^{(n)} + \mathbf{t}^{(n-1)} \right\rVert$
Quadratic, $\mathscr{Q}_1$			$\sum_{n=1}^{N} (\mathbf{t}_n - \mathbf{t}_0)^T \mathbf{Q} (\mathbf{t}_n - \mathbf{t}_0)$

Table 7.2: Specific form of global continuity, smoothness and quadratic constraints $\mathscr{C}\left(\{\mathbf{t}^{(n)}\}_{n=1}^{N}\right)$. The continuity and smoothness constraints are obtained for different finite difference approximations of the first and second derivatives of the experimental conditions function $\mathfrak{F}$. d and $\lVert\cdot\rVert$ represent a distance and a norm in observed space, respectively; $\mathbf{Q}$ and $\mathbf{t}_0$ are constant.



NOT GOOD (high energy) GOOD (low energy)

Figure 7.7: Multiple pointwise reconstruction and continuity constraints from a physical point of view. The left and centre drawings show a one-dimensional case of high and low spring energy, respectively, for the given trajectory fragment. The intersections of the horizontal (abscissa missing) and vertical (ordinate missing) lines with the underlying trajectory mark the locations of the multiple pointwise reconstructions. The right drawing shows a possible setup for two-dimensional constraints.

through $\mathbf{t}^{(1)}, \dots, \mathbf{t}^{(N)}$, which is not surprising if we note that, in the limit $\Delta z \to 0$, $c_n \to c(z)$ is the elementary arc length traced by $\mathfrak{F}$ from z to $z + dz$ and so

$$\int_{\mathbf{t}^{(1)}}^{\mathbf{t}^{(N)}} c(z) = \int_{z^{(1)}}^{z^{(N)}} \lVert \nabla\mathfrak{F}(z) dz \rVert = \int_{z^{(1)}}^{z^{(N)}} \lVert \nabla\mathfrak{F}(z) \rVert \, dz$$

is exactly the arc length between the two end points. For a specific approximation of $\lVert \nabla\mathfrak{F}(z)\, dz \rVert$ applied at N points we will obtain a specific approximation of this arc length. Table 7.2 summarises some choices of global constraints. For the rest of this thesis we will mainly deal of the case of $\mathscr{C}_1$.

The combination of the multiple pointwise reconstruction and the continuity constraint can be seen from a physical point of view as placing a chain of springs so that every joint between two springs must be located on some specific points (the modes of the predictive distribution), as depicted in fig. 7.7. The tensions of the springs would achieve a state of minimal energy of the whole chain constrained to pass through the prescribed locations.

A probabilistic view of the constrained reconstruction problem is given in section 7.7.

For mapping inversion problems, we likewise derive a global forward mapping constraint by summing the terms as defined in section 7.5.3 for every point in the data set:

$$\mathscr{F}\left(\{\mathbf{t}^{(n)}\}_{n=1}^{N}\right) \stackrel{\text{def}}{=} \sum_{n=1}^{N} d\left(\begin{pmatrix} \mathbf{t}_{\mathcal{P}}^{(n)} \\ \mathbf{t}_{\mathcal{M}}^{(n)} \end{pmatrix}, \begin{pmatrix} \mathbf{g}(\mathbf{t}_{\mathcal{M}}^{(n)}) \\ \mathbf{t}_{\mathcal{M}}^{(n)} \end{pmatrix}\right) \tag{7.1}$$

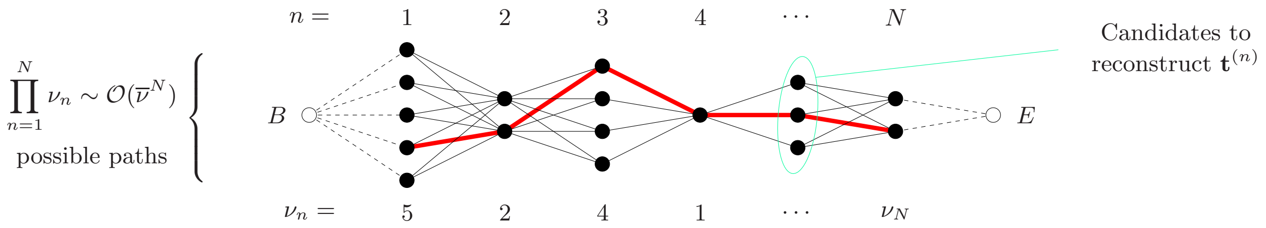


Figure 7.8: Constraint minimisation as a shortest path problem in a layered graph. The nodes in layer n in the graph are the candidate pointwise reconstructions of $\mathbf{t}^{(n)}_{\mathcal{M}^{(n)}}$ (the modes of the conditional distribution $p(\mathbf{t}_{n,\mathcal{M}^{(n)}}|\mathbf{t}_{n,\mathcal{P}_n})$). There are N layers and a single global reconstruction of the data set defines a left-right (or right-left) path passing through all layers exactly once (one such path is highlighted). By imagining fictitious end nodes B and E fully connected by zero-cost links to the end layers 1 and N, respectively, we can reformulate the problem as that of finding the shortest path between B and E. Each edge in the graph is undirected and has an associated cost given by the continuity or smoothness constraint; in the simplest case where we use the curve length as global constraint, the edge cost is the distance in observed space of the reconstructed points. There may also be a cost associated with each node for mapping inversion problems of the form explained in section 7.5.3.

and then combining it linearly with the continuity or smoothness constraint $\mathscr{C}$:

$$\lambda \mathscr{C}\left(\{\mathbf{t}^{(n)}\}_{n=1}^{N}\right) + \mathscr{F}\left(\{\mathbf{t}^{(n)}\}_{n=1}^{N}\right). \tag{7.2}$$

Potentially, both $\mathscr{C}$ and $\mathscr{F}$ may contradict each other at some points, so the weight λ may have to be chosen carefully (the same problem exists in codebook-based approaches to the acoustic-to-articulatory mapping problem, section 10.1.3). Note the analogy with a regularisation formula that combines model fitness with model complexity, such as eq. (2.46) or (6.5).

7.6.2 Constraint minimisation problem

Thus we arrive at the following minimisation problem:

$$\text{Reconstruct the data set as } \arg \min_{\{\mathbf{t}_{n,\mathcal{M}_n}\}_{n=1}^{N} \in \mathcal{S}} \mathscr{C}\left(\{\mathbf{t}_{n,\mathcal{M}_n}\}_{n=1}^{N} | \{\mathbf{t}_{n,\mathcal{P}_n}\}_{n=1}^{N}\right)$$

where the search space $\mathcal{S}$ is the Cartesian product of the N sets of candidate reconstructions for each $\mathbf{t}_{n,\mathcal{M}_n}$ (i.e., each set contains the modes of $p(\mathbf{t}_{n,\mathcal{M}_n}|\mathbf{t}_{n,\mathcal{P}_n})$). This is a combinatorial optimisation problem that can be expressed as finding the shortest path in a layered graph, as represented in figure 7.8. Calling $\nu_n \geq 1$ the number of candidate pointwise reconstructions at point n, the total number of paths is $\prod_{n=1}^{N} \nu_n$, which in an average case is of exponential order in N. Fortunately, there are efficient algorithms both for global (exact) and local (approximate) minimisation.

Figure 9.6 shows a demonstration of the use of the constraints with the multiple pointwise reconstruction given by the modes.

7.6.3 Global minimisation: dynamic programming

The problem of finding the shortest path in a layered graph is a particular case of that of finding the shortest path in an acyclic graph and can be conveniently solved by dynamic programming (Bellman, 1957; Bertsekas, 1987). We can apply dynamic programming because for this problem the following principle of optimality for a decision policy holds: *regardless of the policy adopted at previous stages, the remaining decisions must constitute an optimal policy*, where here a *stage* is a layer in the graph and a *policy* is a sequence of decisions (i.e., a sequence of chosen nodes). This leads us to an algorithm where the decision of what link to choose is taken locally (at each layer n), but ν_n paths must be kept. Figure 7.9 gives the forward recursion version of the dynamic programming algorithm (starting from layer 1) for a global continuity constraint $\mathscr{C}_1\left(\{\mathbf{t}^{(n)}\}_{n=1}^{N}\right) = \sum_{n=1}^{N-1} d(\mathbf{t}^{(n)}, \mathbf{t}^{(n+1)})$. Due to the symmetry of the problem, a backward algorithm (starting from layer N) is equivalent. We choose the forward version because it follows the direction of increasing time for sequence reconstruction. The algorithm requires the following definitions:

initialise
  for $i = 1, \dots, \nu_1$ — For each node of layer 1
    $\mathcal{A}_{1,i} \leftarrow [\mathbf{a}_{1,i}]$ — Path
    $l_{1,i} \leftarrow 0$ — Path length
  end
for $n = 2, \dots, N$ — For each layer
  for $i = 1, \dots, \nu_n$ — For each node of layer n
    $j^* \leftarrow \arg\min_{j=1,\dots,\nu_{n-1}} \{d(\mathbf{a}_{n-1,j}, \mathbf{a}_{n,i}) + l_{n-1,j}\}$
    $l_{n,i} \leftarrow d(\mathbf{a}_{n-1,j^*}, \mathbf{a}_{n,i}) + l_{n-1,j^*}$
    $\mathcal{A}_{n,i} \leftarrow [\mathcal{A}_{n-1,j^*}; [\mathbf{a}_{n,i}]]$
  end
end
$i^* \leftarrow \arg\min_{i=1,\dots,\nu_N} l_{N,i}$
return $\mathcal{A}_{N,i^*}$ — Shortest path

Figure 7.9: Dynamic programming algorithm for global constraint minimisation. Sequences of nodes are written in square brackets and $[\mathcal{A};\mathcal{B}]$ means the concatenation of sequences A and B.

$\{\mathbf{a}_{n,i}\}_{i=1}^{\nu_n}$	The set of candidate pointwise reconstructions for $\mathbf{t}^{(n)}$; thus $\mathbf{a}_{n,i}$ is node i of layer n in the graph.
$\mathcal{A}_{n,i}$	Minimal length path from layer 1 to node i of layer n, for $i = 1, \dots, \nu_n$ at layer n. Thus, $\mathcal{A}_{n,i} = [\mathbf{a}_{1,\bullet}; \mathbf{a}_{2,\bullet}; \dots; \mathbf{a}_{n,i}]$ where $\bullet$ indicates some node.
$l_{n,i}$	Total length of $\mathcal{A}_{n,i}$, i.e., $l_{n,i} \stackrel{\text{def}}{=} \sum_{m=1}^{n-1} d\left(\mathcal{A}_{n,i}(m), \mathcal{A}_{n,i}(m+1)\right)$, where $\mathcal{A}_{n,i}(m)$ is the mth element of the sequence $\mathcal{A}_{n,i}$.

There may be more than one minimal length path from layer 1 to node i of layer n, i.e., the $\arg\min_{j=1,\dots,\nu_{n-1}}$ operation may return several values of j, and so it may be necessary to keep more than one path per node. However, such ties are unlikely in practical cases involving real numbers, so we ignore this possibility in the practical implementation of the algorithm—all we lose is the remote possibility of having more than one optimal path of exactly the same length (but see section 7.9.2).

The dynamic programming algorithm examines each link in the graph (i.e., each pair of nodes in adjacent layers) only once, thus achieving its mission very efficiently.

7.6.4 Local minimisation: greedy algorithm

The dynamic programming algorithm is guaranteed to find a path of globally minimal cost, but it needs to keep ν_n paths at layer n, which can be costly if the average number of candidates per layer $\overline{\nu}$ is high (in the acoustic-to-articulatory mapping problem of chapter 10, ν_n can be several hundreds). For this reason, we give in fig. 7.10 a suboptimal algorithm that needs only keep one path at all times; it is also more intuitive than dynamic programming, so in fact one might think of it first to solve the constraint optimisation. The algorithm is a greedy version of dynamic programming: at layer n, it simply selects the minimal cost edge (i.e., the closest node). The starting point can be any node in any layer (the first layer, say); to improve the chances of getting a good path, it is better to start in a layer having very few nodes (hopefully just one). If such layer is an intermediate one ($1 < n_0 < N$), the algorithm greedily proceeds leftwards to 1 and rightwards to N, since all edges are undirected.

We anticipate that this algorithm is not an option for two reasons[16]. First and foremost it usually leads to poor solutions, not just in terms of a high value of the constraint, but also as yielding a high reconstruction error—which is our ultimate criterion. Such solutions are sensitive to the choice of starting layer n_0. Second,

[16] One might also try to use a *divide-and-conquer* algorithm along the lines of: find the layer n_0 with fewest nodes; for each node in it, solve the left ($n < n_0$) and right ($n > n_0$) subproblems; join the solutions; pick the best solution of all nodes in layer n_0. This does find a global minimal cost path, but with an average computational complexity of $\mathcal{O}\left(2^{\lfloor \log_2 N \rfloor} \overline{\nu}^{\lfloor \log_2 N \rfloor + 1}\right)$ it is no opponent for dynamic programming in any practical situation where $\overline{\nu}$ is significantly more than 1. Divide-and-conquer algorithms usually perform very well; the reason why this particular one does not is that some of the edges are examined more than once: in the divide-and-conquer cutoff point (say, the middle layer), every node is used only once; but the left and right subproblems "repeat themselves" (or part of themselves) for every node in that cutoff layer.

initialise

$n_0 \leftarrow$ one of $\begin{cases} 1 \\ \arg\min_{n=1,\dots,N} \nu_n \end{cases}$ Starting layer

$i \leftarrow$ any node in layer n_0 First node

$\mathcal{A}(n_0) \leftarrow [\mathbf{a}_{n_0,i}]$ Path

for $n = n_0 - 1, \dots, 1$ Left part of the path (only if $n_0 > 1$)

$j^* \leftarrow \arg\min_{j=1,\dots,\nu_n} \{d(\mathbf{a}_{n,j}, \mathcal{A}(n+1))\}$

$\mathcal{A}(n) \leftarrow \mathbf{a}_{n,j^*}$

end

for $n = n_0 + 1, \dots, N$ Right part of the path (only if $n_0 < N$)

$j^* \leftarrow \arg\min_{j=1,\dots,\nu_n} \{d(\mathcal{A}(n-1), \mathbf{a}_{n,j})\}$

$\mathcal{A}(n) \leftarrow \mathbf{a}_{n,j^*}$

end

return $\mathcal{A}$

Figure 7.10: Greedy algorithm for global constraint minimisation.

its computational complexity is hardly better than that of dynamic programming. More details are given in section 7.8 and in the experiments of chapters 9 and 10.

7.7 Probabilistic interpretation

In this section we adopt the shorthand notation $\mathbf{t}_m^n \stackrel{\text{def}}{=} \{\mathbf{t}^{(m)}, \mathbf{t}^{(m+1)}, \dots, \mathbf{t}^{(n)}\}$, commonly used in the hidden Markov model literature to write vector sequences.

The following theorem results from decomposing $p(\mathbf{t}_1^N)$ by conditional independence assumptions and by assuming that $p(\mathbf{t}^{(n)})$ is the same for all values of n and for all modes.

Theorem 7.7.1. *Under a bidirectional Markovian assumption, the following two optimisation problems are equivalent:*

$$\min_{\{\mathbf{t}^{(n)}\}_{n=1}^N} \sum_{n=1}^{N} \left\| \mathbf{t}^{(n+1)} - \mathbf{t}^{(n)} \right\|^2 \qquad \textit{and} \qquad \max_{\{\mathbf{t}^{(n)}\}_{n=1}^N} p(\mathbf{t}_1^N)$$

where:

- $\|\mathbf{u}\|^2 \stackrel{\text{def}}{=} \sum_{d=1}^D w_d u_d^2 = \mathbf{u}^T \mathbf{W} \mathbf{u}$ *is a weighted squared Euclidean distance of positive semidefinite matrix* $\mathbf{W} \stackrel{\text{def}}{=} \operatorname{diag}(w_1, \dots, w_D)$.
- $\mathbf{t}^{(n+1)} | \mathbf{t}^{(n)} = \boldsymbol{\tau} \sim \mathcal{N}\left(\boldsymbol{\tau}, (2\mathbf{W})^{-1}\right)$, *i.e., normal of mean* $\boldsymbol{\tau}$ *and covariance matrix* $(2\mathbf{W})^{-1}$.

Proof.

$$\begin{aligned}
\frac{p(\mathbf{t}_1^N)}{p(\mathbf{t}^{(1)})} &= \prod_{n=1}^{N-1} p(\mathbf{t}^{(n+1)} | \mathbf{t}_1^n) \stackrel{\text{(a)}}{=} \prod_{n=1}^{N-1} p(\mathbf{t}^{(n+1)} | \mathbf{t}^{(n)}) \\
&\stackrel{\text{(b)}}{=} \prod_{n=1}^{N-1} \left| 2\pi (2\mathbf{W})^{-1} \right|^{-\frac{1}{2}} e^{-\frac{1}{2} (\mathbf{t}^{(n+1)} - \mathbf{t}^{(n)})^T 2\mathbf{W} (\mathbf{t}^{(n+1)} - \mathbf{t}^{(n)})} \\
&= C e^{-\sum_{n=1}^{N-1} (\mathbf{t}^{(n+1)} - \mathbf{t}^{(n)})^T \mathbf{W} (\mathbf{t}^{(n+1)} - \mathbf{t}^{(n)})} = C e^{-\sum_{n=1}^{N-1} \left\| \mathbf{t}^{(n+1)} - \mathbf{t}^{(n)} \right\|^2} \\
&\Rightarrow \max \frac{p(\mathbf{t}_1^N)}{p(\mathbf{t}^{(1)})} = \min \sum_{n=1}^{N-1} \left\| \mathbf{t}^{(n+1)} - \mathbf{t}^{(n)} \right\|^2
\end{aligned}$$

where we have applied the Markov assumption in (a) and the normality assumption in (b). Since $\left\|\mathbf{t}^{(n+1)} - \mathbf{t}^{(n)}\right\|^2 = \left\|\mathbf{t}^{(n)} - \mathbf{t}^{(n+1)}\right\|^2$, the result also holds for the reversed sequence and so $p(\mathbf{t}_1^N)/p(\mathbf{t}^{(1)}) =$

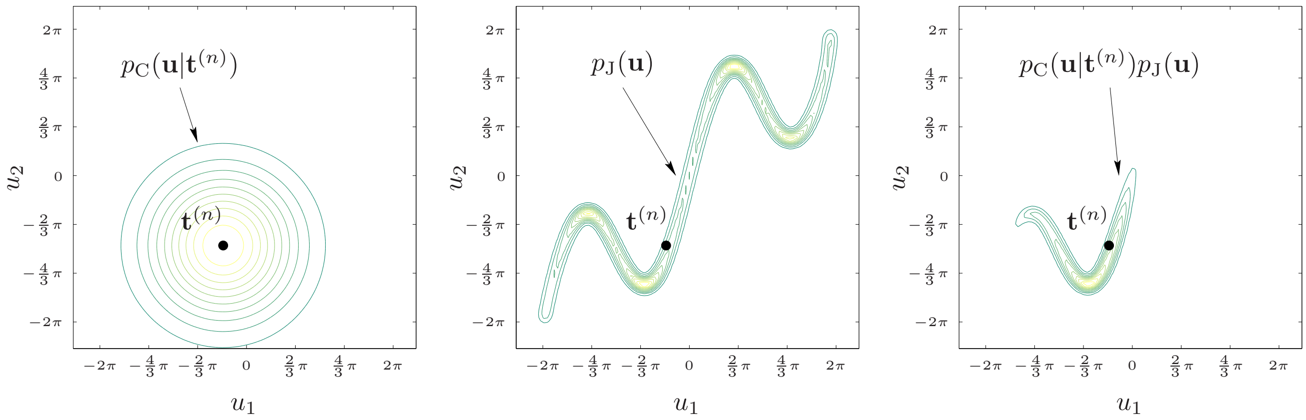


Figure 7.11: Combination of probabilistic constraint and joint pointwise density. Assuming that point $\mathbf{t}^{(n)}$ is known, the likely areas for the next point are given by the product (AND operation) of the constraint density $p_C(\mathbf{u}|\mathbf{t}^{(n)})$ and the joint density $p_J(\mathbf{u})$. In the left graph, the constraint density is assumed spherical normal centred in $\mathbf{t}^{(n)} = \binom{-1}{-3}$ and with standard deviation $\sigma = 2$. In the centre graph, the joint density is the same as the one used for the toy model of chapter 9. All graphs are contour plots of the corresponding density in the space of u_1 and u_2 and both variables u_1 and u_2 are assumed to be missing.

$p(\mathbf{t}_1^N)/p(\mathbf{t}^{(N)})$. Likewise, if we split the sequence at an arbitrary step $1 < k < N$ we have

$$\prod_{n=1}^{N-1} p(\mathbf{t}^{(n+1)}|\mathbf{t}^{(n)}) = \underbrace{\prod_{n=1}^{k-1} p(\mathbf{t}^{(n+1)}|\mathbf{t}^{(n)})}_{\frac{p(\mathbf{t}_1^k)}{p(\mathbf{t}^{(1)})} = \frac{p(\mathbf{t}_1^k)}{p(\mathbf{t}^{(k)})}} \underbrace{\prod_{n=k}^{N-1} p(\mathbf{t}^{(n+1)}|\mathbf{t}^{(n)})}_{\frac{p(\mathbf{t}_k^N)}{p(\mathbf{t}^{(k)})} = \frac{p(\mathbf{t}_k^N)}{p(\mathbf{t}^{(N)})}}$$

and so $p(\mathbf{t}^{(1)}) = p(\mathbf{t}^{(k)}) = p(\mathbf{t}^{(N)})$. Therefore $p(\mathbf{t}^{(n)}) =$ constant for $n = 1, \dots, N$ and $\max p(\mathbf{t}_1^N)/p(\mathbf{t}^{(1)}) = \max p(\mathbf{t}_1^N)$. $\square$

The set in which the variables over which the max- or minimisation takes place, $\{\mathbf{t}^{(n)}\}_{n=1}^N$, is irrelevant for the theorem, although in our case this is the set of multiple pointwise reconstructions. It is also independent of the method that was used to generate such set, thus being valid for vector quantisation, committees of neural networks, etc.

This result allows to generalise the continuity constraint in a probabilistic way by defining different distributions for $p(\mathbf{t}^{(n+1)}|\mathbf{t}^{(n)})$ other than diagonal normal or by defining different conditional independence assumptions other than the Markov assumption. In fact, a smoothness-based constraint will be equivalent to a second-order Markov assumption.

7.7.1 Distributions over trajectories

The previous derivation is trivial: weighted Euclidean distances are replaced with Gaussian distributions and constraints with Markov assumptions. It would be more useful to define a grand joint density function $p_G(\{\mathbf{t}^{(n)}\}_{n=1}^N)$ encompassing both the density model for a single point and the continuity constraints that link neighbouring points. Such a grand density would then be a distribution over trajectories (or data sets) rather than over points, and choosing a representative point of it (in the sense of section 7.3) would give us a likely reconstruction of the trajectory without having to explicitly minimise a constraint over the pointwise candidate reconstructions. Thus, it would attack the problem of global reconstruction directly, rather than splitting it into a set of pointwise reconstruction problems followed by a constraint optimisation. From a generative point of view, the grand distribution p_G would be a sequence generative model while the joint density model p_J would be a pointwise generative model.

In principle, it would be conceivable that this grand distribution be unimodal since there should be less ambiguity in the global reconstruction than in the local ones (except in pathological cases, as in fig. 7.12); in this case no optimisation would be required at all, since we could take the mean of the grand distribution as reconstructed trajectory.

The grand density should be constructed from the pointwise joint density for each point, $p(\mathbf{t}^{(n)})$, and from the constraints. That is, *the problem is to find a grand density p_G with marginals $p(\mathbf{t}^{(1)}), \dots, p(\mathbf{t}^{(N)})$ (with the form of the individual density model) and conditionals $p(\mathbf{t}^{(2)}|\mathbf{t}^{(1)}), \dots, p(\mathbf{t}^{(N)}|\mathbf{t}^{(N-1)})$ (as given by the continuity constraints)*. However, it does not seem possible to find a density verifying that; instead, we offer a heuristic combination in probabilistic form of both terms. Given a point $\mathbf{t}^{(n)}$, how likely is a point $\mathbf{u}$ to be the next point, $\mathbf{t}^{(n+1)}$? On the one hand, the further $\mathbf{u}$ is from $\mathbf{t}^{(n)}$, the less likely. We express this by a probabilistic continuity constraint as earlier: $p_{\mathrm{C}}(\mathbf{u}|\mathbf{t}^{(n)}) = \mathcal{N}(\mathbf{t}^{(n)}, \sigma^2\mathbf{I})$. The parameter σ would be related to the speed at which the curve $\mathbf{t} = \mathfrak{F}(\mathbf{z})$ is traversed, and more generally we could take an arbitrary covariance matrix. On the other hand, the point $\mathbf{u}$ itself must lie in likely areas of the observed space. We express this by the joint density model: $p_{\mathrm{J}}(\mathbf{u}) = p(\mathbf{u})$. Now we combine both conditions as an AND operation, i.e., we take their product. Figure 7.11 shows an example of $p_{\mathrm{C}}(\mathbf{u}|\mathbf{t}^{(n)})p_{\mathrm{J}}(\mathbf{u})$. Finally, we combine such products again as an AND for the whole sequence, taking into account that either the first or the last point is not constrained by its neighbour:

$$p_{\mathrm{J}}(\mathbf{t}^{(1)}) \prod_{n=1}^{N-1} p_{\mathrm{C}}(\mathbf{t}^{(n+1)}|\mathbf{t}^{(n)}) p_{\mathrm{J}}(\mathbf{t}^{(n+1)}) = \prod_{n=1}^{N} p_{\mathrm{J}}(\mathbf{t}^{(n)}) \prod_{n=1}^{N-1} p_{\mathrm{C}}(\mathbf{t}^{(n+1)}|\mathbf{t}^{(n)}). \tag{7.3}$$

We call this combination "heuristic" because the result is not a density function (although it could be normalised) and it does not satisfy our earlier desideratum. That is, its marginals are not the originally specified $p(\mathbf{t}^{(1)}), \dots, p(\mathbf{t}^{(N)})$ and its conditionals are not the originally specified $p(\mathbf{t}^{(2)}|\mathbf{t}^{(1)}), \dots, p(\mathbf{t}^{(N)}|\mathbf{t}^{(N-1)})$. Now, even if this function was unimodal, we would not be able to compute its mean analytically as speculated earlier, so numerical optimisation is necessary, e.g. by gradient ascent[17]. If equivalently, we maximise its logarithm, we find

$$\sum_{n=1}^{N} \ln p_{\mathrm{J}}(\mathbf{t}^{(n)}) - \frac{1}{2\sigma^2} \sum_{n=1}^{N-1} \left\| \mathbf{t}^{(n+1)} - \mathbf{t}^{(n)} \right\|^2 \tag{7.4}$$

which has the standard form[18] of a fitness term (on the left) and a constraint term (on the right) with weight $1/2\sigma^2$. We can obtain the method we have proposed in this chapter as a limit case of eq. (7.4): if $p_{\mathrm{J}}(\mathbf{t}^{(n)})$ is taken as a sum of deltas centred at the modes of $p_{\mathrm{J}}\left(\mathbf{t}^{(n)}_{\mathcal{M}^{(n)}}|\mathbf{t}^{(n)}_{\mathcal{P}^{(n)}}\right)$, then the search space is restricted to those modes and operates only on the right side term (our continuity constraint)[19].

If, as mentioned earlier, the function of eq. (7.4) was unimodal, we would be free from local-maxima problems; but the truth is that it may have many local maxima where point $\mathbf{t}^{(n)}$ tends to a mode of $p_{\mathrm{J}}\left(\mathbf{t}^{(n)}_{\mathcal{M}^{(n)}}|\mathbf{t}^{(n)}_{\mathcal{P}^{(n)}}\right)$ for all n; this will certainly be the case if such conditional distributions are very sharply peaked. In our method, the dynamic programming search is guaranteed to find the global optimum. On the other hand, an objective function over trajectories opens the door for multiple global reconstruction as defined in section 7.2.6.

Another possible disadvantage of this method is that the candidate pointwise reconstructions (the modes) are weighted by their respective density values, while in our method all modes are equally weighted. This may bias the reconstruction towards highly likely pointwise reconstructions at the expense of the continuity constraint. Finally, we are also left with the choice of the tradeoff parameter σ. An implementation and evaluation of this method is left for future research.

Using eq. (7.3) we can generate sequences as follows: generate a point $\mathbf{t}^{(1)}$ from $p_{\mathrm{J}}(\mathbf{t})$; then generate a point $\mathbf{t}^{(1)}$ from the normalised version of $p_{\mathrm{C}}(\mathbf{t}^{(2)}|\mathbf{t}^{(1)})p_{\mathrm{J}}(\mathbf{t}^{(2)})$; and so on. In fact, this results in a random walk (the term $p_{\mathrm{C}}(\mathbf{t}^{(n+1)}|\mathbf{t}^{(n)})$) constrained to the data manifold (the term $p_{\mathrm{J}}(\mathbf{t}^{(n+1)})$). The weighted distance (i.e., the covariance matrix) of the continuity constraint determines the "sampling period" of the sequence.

7.8 Computational complexity

Reconstructing a data set $\{\mathbf{t}^{(n)}\}_{n=1}^{N}$ of N vectors requires two separate computations: (1) implicitly constructing the layered graph of fig. 7.8, i.e., computing the multiple pointwise reconstructions of each point; (2) finding

[17]Remember that the optimisation takes place only over the missing variables $\{\mathbf{t}^{(n)}_{\mathcal{M}^{(n)}}\}_{n=1}^{N}$, since the present ones are frozen. This also precludes the local optimum where $\mathbf{t}^{(1)} = \cdots = \mathbf{t}^{(N)}$ (a zero-length trajectory) that would occur in the (practically uninteresting) situation where there are no present variables at all.

[18]Compare it with the elastic net objective function of eq. (4.3): our $\mathbf{t}^{(n)}$ would be a net knot $\boldsymbol{\mu}_m$, but there are no "cities".

[19]Strictly, this requires to find the global maximum of eq. (7.4), since for the delta limit any trajectory where every point $\mathbf{t}^{(n)}$ is a mode represents a local maximum.

the shortest path on the graph, i.e., minimising the constraint on the multiple pointwise reconstructions search space. We analyse them separately[20].

There is also the cost of estimating the joint density model in observed space, but since this is done offline with a training set and the resulting density can be reused for many different data sets, we do not consider it here.

7.8.1 Computing the multiple pointwise reconstructions

If R_n is defined as the cost of computing all pointwise candidate reconstructions of point n (all the modes of the conditional distribution) and $\overline{R} \stackrel{\text{def}}{=} \frac{1}{N}\sum_{n=1}^{N} R_n$ as the average such cost, a simple average estimate would be $\mathcal{O}(N\overline{R})$. Estimating R_n is difficult, though. For the algorithms of chapter 8 (where the conditional distribution is a Gaussian mixture of M components) finding all the modes at point n requires $\hat{M}_n$ iterative searches in D-dimensional space. $\hat{M}_n$ is the *effective number of components* of the mixture (the number of components whose mixing proportion is higher than a user-defined threshold used to discard spurious modes): it will usually be a small fraction of M, depending in a complex way on the nature of the data manifold, the amount of missing data and the exact missing data pattern, as well as on the threshold. It is thus unknown. The cost of a search depends on the (also unknown) number of iterations required, each of which is proportional to $\hat{M}_n$ and to D, D^2 or D^3 (depending on whether spherical, diagonal or full covariance matrices are used and whether gradient or quadratic optimisation is used). The accuracy of the modes obtained can also be regulated by further user parameters. Consequently, all we can say is that the order of the average cost of computing all multiple pointwise reconstructions is proportional to $ND^\alpha \hat{M}^2$, where $1 \leq \alpha \leq 3$ and $\hat{M} \stackrel{\text{def}}{=} \sqrt{\frac{1}{N}\sum_{n=1}^{N} \hat{M}_n^2}$ is a "small fraction" of M.

7.8.2 Minimising the constraint

Define the average number of candidate pointwise reconstructions as $\overline{\nu} \stackrel{\text{def}}{=} \frac{1}{N}\sum_{n=1}^{N} \nu_n$ (for unbounded horizon problems, described in section 7.9.3, N can be taken as some large enough number).

Dynamic programming From the algorithm of fig. 7.9 we see that every link between layer $n-1$ and layer n is used once (loop on $i = 1, \dots, \nu_n$ and minimisation on $j = 1, \dots, \nu_{n-1}$). That gives $\nu_{n-1}\nu_n$ links which, summed for all layers (loop on $n = 2, \dots, N$) gives $\sum_{n=1}^{N} \nu_{n-1}\nu_n$, indeed the total number of links in the graph. For an average case where every layer has $\overline{\nu} \geq 1$ nodes we get a complexity of $\mathcal{O}(N\overline{\nu}^2)$.

Greedy algorithm From the algorithm of fig. 7.10 we see that not every link between layer $n-1$ and layer n is used, only ν_n (or ν_{n-1} if going backwards) of them are. So the number of links used is $\sum_{n=1}^{n_0-1} \nu_n + \sum_{n=n_0+1}^{N} \nu_n = \sum_{n=1}^{N} \nu_n - \nu_{n_0}$ if the starting layer n_0 is chosen at random. If we select as starting layer the one with fewest nodes, an additional $\mathcal{O}(N)$ term must be added. For an average case where every layer has $\overline{\nu} \geq 1$ nodes we get a complexity of $\mathcal{O}(N\overline{\nu})$. This is $\overline{\nu}$ times faster than the dynamic programming algorithm, which might be considerable for problems where many pointwise reconstructions are possible—perhaps a data set with lots of missing data.

7.8.3 Conclusion

Since $\alpha \geq 1$ and $\overline{\nu} \leq \hat{M}$, and usually $\overline{\nu}$ is a fraction of $\hat{M}$ (since Gaussian mixture components coalesce), the dominant computational complexity term is that of computing the multiple pointwise reconstructions, as we confirmed experimentally. We also found that a crucial factor is the amount of missing data (as happens with EM algorithms for maximum likelihood): the higher it is, the higher $\overline{\nu}$ becomes and the longer the algorithm takes.

Thus, reconstructing a large data set with lots of missing data can take long due to the exhaustive mode search, which may be a problem for real-time reconstruction. As as indication for a hard, real-world problem, for the mappings between electropalatographic and acoustic speech features of chapter 10, reconstructing an utterance a few seconds long (for which typically $N \approx 400$ and $D \approx 75$) takes from some seconds (EPG to acoustic, 16% data missing) to 10 minutes (acoustic to EPG, 84% data missing) in our Matlab implementation. Section 10.2.4 gives additional execution times. It is possible to accelerate the mode search by:

[20]Some familiarity with the algorithms of chapter 8 is required, especially in section 7.8.1.

- Raising the threshold parameter θ of section 8.2.3 to discard low-probability mixture components in the conditional distribution. Doing this, we obtained up to $10\times$ speedup with up to 17% increase in reconstruction error.
- Reducing the number of mixture components (either by simply using fewer components or by using full-covariance mixtures[21]) at the cost of a less accurate density model.

Considerable speedups should also be gained by coding the method more efficiently (or by waiting for a couple of years for a faster CPU generation!).

7.9 Discussion

In this chapter, we defined the problem of missing data reconstruction, which includes multivariate regression, mapping approximation and mapping inversion as particular cases. We defined single/multiple pointwise/global reconstruction and constraints based on "experimental conditions" variables. We then proposed an algorithm for single global reconstruction based on multiple pointwise reconstruction and constraint minimisation: multiple pointwise reconstruction was achieved as the modes of the conditional distribution of the missing variables given the present ones for the point being reconstructed; constraint minimisation was achieved by dynamic programming. For the joint probability model of the observed variables, we chose to use a latent variable model (GTM) which adopted the form of a Gaussian mixture; exhaustive mode finding in Gaussian mixtures was achieved with the algorithms described in chapter 8; and the constraint was defined as (temporal) continuity via the trajectory length in the (weighted) Euclidean distance sense.

From a probabilistic point of view, the joint density model can be used as a pointwise generative model, while its product with the continuity constraint (viewed as a conditional probability of the new point given the current one), can be used as a sequence generative model, where the sequences are random walks along the data manifold.

Computationally, the bottleneck at reconstruction time is the exhaustive mode finding in the conditional distribution (the dynamic programming search being very fast in comparison), particularly for high amounts of missing data. This may limit the method in some real-time applications.

This section discusses additional issues. Chapters 9 and 10 report experimental results.

7.9.1 Choice of density model: robustness and smoothness

Robustness is intuitively defined as insensitivity to small deviations from the assumptions (Huber, 1981). An important aspect of robustness concerns the effect of outliers in the training set on the estimated density model. Classical statistical procedures, including our latent variable models, are typically based on assumptions of normality and thus are not robust. One possibility of difficult implementation is the use of long-tailed distributions, as mentioned in section 2.3.1. Peel and McLachlan (2000) (also in McLachlan and Peel, 2000, chapter 7) have proposed the use of mixtures of Student-t distributions (trained with an ECM algorithm). This would require a new mode-finding algorithm, since that of chapter 8 is specific for Gaussian mixtures.

The modes are a key aspect of our approach. Unfortunately, the mode is not a robust statistic of a distribution since small variations in the distribution shape can have large effects on the location and number of modes. This is related to the **smoothness** of the density model and is demonstrated in section 9.2.3: with finite mixtures of localised functions, spurious modes can appear as ripple superimposed on a smoothly-varying function. These spurious modes have the following characteristics:

- They can happen in regions where the mixture components are sparsely distributed and have little interaction, as in the boundaries of the density model, but not only there.
- They can happen whether the components have independent or shared covariance parameters, and whether the covariance parameters are isotropic, diagonal or full.
- Their probability value can be as high as that of true modes, which rules out the use of a rejection threshold to filter the spurious ones out (see section 8.2.3).
- They are not due to local maxima or singularities of the log-likelihood surface.

[21]Which, having more parameters, have to keep the number of components down to avoid overfitting.

Such spurious modes can lead the dynamic programming search to a wrong trajectory and a large reconstruction error, although we observed this only in regression problems, not in general missing data patterns. For regression problems, specially mapping inversion, it should be possible to prevent spurious modes from forming part of the reconstructed sequence by applying a forward mapping constraint (section 7.5.3 and equations (7.1)–(7.2)), where the forward mapping may be known exactly or derived by a mapping approximator. The reason is that spurious modes, by definition, will give a high value for constraint $\mathscr{F}$ in eq. (7.1).

To prevent spurious modes from entering the constraint minimisation at all, possible solutions are:

- To globally smooth the density model at estimation time. This may be done in different ways:
  - By regularisation, adding a term to the log-likelihood to penalise low variance parameters, thus penalising nonsmooth models. For example, a zero-mean Gaussian prior on σ^{-1} for GTM. GTM includes a regularisation parameter for the mapping $\mathbf{f}$ (section 2.6.5), but this does not guarantee a smooth density model. Perhaps a regularisation to force $\mathbf{f}$ to be unit-speed (section 2.8.3) would work better, by uniformly separating the mixture components in observed space. The extension of GTM to a manifold-aligned noise model (section 2.6.5.1) may also result in a smoother (full-covariance) Gaussian mixture.

    However, the cost of smoother density models is to increase the width of the error bars of each mode location (if used) and to incur higher errors where the data manifold bends itself (see section 9.2.5).

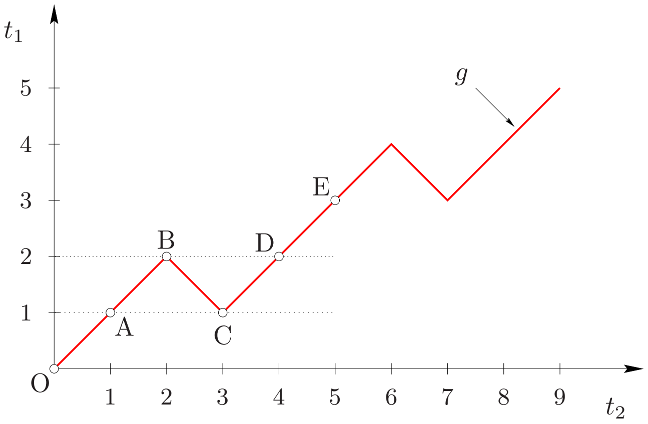
    Several methods have been proposed to obtain a sparse Gaussian mixture, i.e., one that represents well the data with a small number of components[22]. For example, Roberts et al. (1998) and Ormoneit and Tresp (1998) use Bayesian regularisation to penalise overcomplex models and so obtain an optimal number of components. Weston et al. (1999) and Vapnik and Mukherjee (2000) use support vector machines with a Gaussian kernel for density estimation, which results in a sparse Gaussian mixture due to the property of support vector machines of selecting only a few support vectors. Ueda et al. (2000) propose an algorithm that splits and merges components according to a criterion embedded in the EM algorithm. The algorithm of Brand (1999) uses a maximum-a-posteriori (MAP) estimation where the likelihood is maximised and the entropy of the mixing proportions is minimised, and at the same time that it obtains the optimal parameters it removes components. However, it is likely that, in their efforts to reduce the number of components, these methods will result in nonsmooth models. Ormoneit and Tresp (1998) also propose an approach based on a ensemble of Gaussian mixtures, each trained differently to disagree from the others, with the final density model being the average of the ensemble members (see section 7.11.2.1); this might produce smoother models.

    Finally, if the density model is obtained by kernel estimation (Silverman, 1986), it is possible to either oversmooth the model (at the cost of a worse fit) or to use different values for the smoothing parameter in different areas of the space (adaptive kernel estimation). In both cases, the disadvantage is that the number of kernels is very large, equal to the number of training points.
  - By increasing the number of components. In latent variable models that sample the latent space prior distribution (e.g. GTM), the mixture centroids in data space (associated with the latent space samples) are not trainable parameters. We can then improve the density model at a higher computational cost with no generalisation loss by increasing the number of mixture components. The number of components required will depend exponentially on the intrinsic dimensionality of the data (ideally coincident with that of the latent space, L) and not on the observed one, D. However, good coverage in latent space does not guarantee good coverage in data space because the amount by which the latent space is stretched when mapped onto the data space (the local magnification factor, defined in section 2.8.3) may separate the component centres.
- To locally smooth the density model at mode-finding time, i.e., to smooth the conditional distribution.
- To look for *bumps* (regions of high probability mass) instead of modes. This is discussed in section 7.9.9.

If smoothing the density model, either globally or locally, the difficulty is just how much to smooth so that important modes are not removed.

Outliers at reconstruction time can affect the dynamic programming search when they result in violations of the constraint, particularly discontinuities. This is discussed in section 7.9.6.

[22]The word "sparse" here is used purely to mean "few components" and is not related to the concept of distribution sparseness of section 7.3.1. Besides, the number of components of a Gaussian mixture does not determine the sparseness of the distribution; for example, a Gaussian mixture with many narrow, separated components will have a higher distribution sparseness than a single, broad Gaussian.



Reconstructed trajectory	$\mathscr{C}_1$	$\mathscr{S}_1$
$\overline{\mathrm{ABCD}}$	$3\sqrt{2}$	4
$\overline{\mathrm{ABCB}}$	$3\sqrt{2}$	$2+2\sqrt{2}$
$\overline{\mathrm{ABAB}}$	$3\sqrt{2}$	$4\sqrt{2}$
$\overline{\mathrm{CBAB}}$	$3\sqrt{2}$	$2+2\sqrt{2}$
etc.		

Figure 7.12: Several global reconstructions with the same length. The reconstruction problem shown is the inversion of the forward mapping $t_2 = g(t_1)$ at several given values of t_2 in the interval $[1, 2]$ in the following sequence: $1, 2, 1, 2$ (for simplicity, but more realistically there would be intermediate values between 1 and 2). A possible reconstruction is the trajectory $\overline{\mathrm{ABCD}} = [\binom{1}{1}, \binom{2}{2}, \binom{3}{1}, \binom{4}{2}]$, but not the only one. The table shows several others (and there are more), all of which have the same length (value of $\mathscr{C}_1$) but different smoothness (value of $\mathscr{S}_1$). If the missing values were $0, 1, 2, 1, 2, 3$ then the minimal-length reconstructed trajectory would be unique: $\overline{\mathrm{OABCDE}}$.

7.9.2 Choice of constraints

In principle, constraints based on other criteria (e.g. smoothness instead of continuity) may result in significantly different data set reconstructions. Figure 7.12 shows an example where a segment of a trajectory $\overline{\mathrm{ABC}}$ admits several reconstructions that have the same length. Those reconstructions are indistinguishable when using continuity constraint $\mathscr{C}_1$, but the smoothness constraint does distinguish between some of them, penalising every sharp turn. However, the situation where several global reconstructions tie in the constraint value is unlikely in practice because the following factors would break the ambiguity:

- An imperfect density model derived from a noisy sample.
- An arbitrary pattern of missing data or the presence of one-to-one relationships.
- A longer trajectory segment, e.g. $\overline{\mathrm{OABCDE}}$.
- The use of finite-precision computer arithmetic.

Therefore, it is not clear how different the effect would be without actually performing simulations, since the degree of discretisation (given by the sampling of the experimental condition variables) is likely to have some influence on the result.

Another important point is that applying a given type of constraint to different coordinate systems may lead to qualitatively different kinds of behaviour. Jordan (1990)[23] gives the example of a pianist playing a lengthy ascending scale: crossing the fingers leads to faster transitions between piano keys. That is, a nonsmooth trajectory in actuator[24] space entails a high cost (the difficulty of crossing the fingers) but it is offset by the gain in the speed of transitions in the work space (from note to note). Section 10.2.3 shows some results with the use of weighted Euclidean distances.

It would be interesting to test the method with multidimensional constraints (when the experimental conditions are multidimensional). Section 7.10 discusses a possible example of this. An important problem with constraints of dimension D higher than one is the curse of the dimensionality: the complexity of the multidimensional dynamic programming algorithm grows exponentially with D (specifically: $\mathcal{O}(N_1 N_2 \dots N_D)$ times a term for $\overline{\nu}$); e.g. see Durbin et al. (1998, pp. 141–143), who consider it for multiple sequence alignment methods in biological sequence analysis. Thus, global minimisation will not be feasible except for very small dimensions D. Further research is necessary to develop efficient heuristic approximations to multidimensional dynamic programming.

[23] I am grateful to Michael Jordan for sending me a hardcopy of this paper.
[24] These terms are explained in the robot arm problem of section 9.3.1.

7.9.3 Unbounded horizon problems

The dynamic programming problem considered in section 7.6.3 is finite (N is finite), deterministic (once a decision is taken, the transition is deterministic) and discrete (the number of nodes ν_n at each layer n is finite). Here we consider the case where N is infinite. There are practical reconstruction problems where the data set to be reconstructed is infinite or long enough that the user periodically demands partial reconstruction; for example, in continuous speech with missing data, the user should receive reconstructed speech in real time, which requires that past speech be frozen once reconstructed, passed to the user and discarded for reconstruction of future speech. In operations research problems such as inventory control this is called an unbounded horizon problem and approaches to it usually assume stationarity (Wagner, 1975, chapters 11–12 and 17–18)—which we have assumed throughout by using a density model independent of n.

The greedy algorithm, if started from point $n_0 = 1$, requires no modification for unbounded horizon problems, but we do not recommend it for the reasons of section 7.9.4. As for the dynamic programming algorithm, there are two simple approaches:

- Upon arrival of point $\mathbf{t}^{(n)}$, which has some missing values, compute all ν_n possible reconstructed trajectories—each one having as end node a different candidate reconstruction for $\mathbf{t}^{(n)}$. This can be done incrementally from the ν_{n-1} trajectories of the previous step in $\mathcal{O}(\overline{\nu}^2)$ average time and the current best trajectory can be provided if desired. This method remains exact, i.e., it still finds the global optimum up to point n. Its drawback is the storage cost of, on the average, $\overline{\nu}$ D-dimensional trajectories of length N that grows indefinitely.

- Split the data stream into chunks (perhaps coinciding with user requests) and reconstruct them separately by dynamic programming. This has the risk of (1) getting discontinuities at the splitting points and (2) getting a suboptimal reconstruction of the whole stream by concatenating the reconstructed chunks. It has the advantage of keeping on the average $\overline{\nu}$ D-dimensional trajectories of length less than or equal to the chunk length, rather than equal to N. It should be possible to improve the average performance of this suboptimal approach by using heuristic rules. For data-rich applications with limited storage capability this is the way to go.

Both approaches may be combined if at the splitting points there is a unique pointwise reconstruction ($\nu_n = 1$), since effectively this splits the layered graph into separate subgraphs (e.g. at node $n = 4$ in fig 7.8). That is, whenever $\nu_n = 1$ the reconstructed trajectory for points earlier than n can be frozen (to its optimal value) and the dynamic programming algorithm "restarted" from scratch, saving computer time and storage. Depending on the particular application and on the amount of missing data such zero-uncertainty points may be common; in speech, one likely example are silent frames, which are easily detected by thresholding the frame energy.

7.9.4 Dynamic programming algorithm versus greedy algorithm

The greedy algorithm has a tendency to obtain reconstructed trajectories that retrace themselves at turning points, as the "movie" in fig. 7.13 shows using the example of fig. 7.12. This behaviour is due to two reasons: at the turning point (B), one of its two neighbouring points must be the closest[25]: either the reconstructed point to the left (L) or the one to the right (R); and the greedy algorithm cannot undo a decision. While in the examples of figures 7.12 and 7.13 there is ambiguity in short segments of trajectory with respect to $\mathscr{C}_1$, for longer segments the greedy algorithm results in suboptimal reconstructions with retracing and abrupt jumps, as in figure 9.10 (see also the reconstruction error in table 9.1). For multidimensional experimental conditions, it is not clear what kind of retracing behaviour can be expected. The reconstruction found by the greedy algorithm is also sensitive to the starting layer (n_0 in fig. 7.10)—an issue that does not arise with dynamic programming.

The greedy algorithm has the advantage over dynamic programming of a slightly lower computational complexity, depending on the average number of candidate pointwise reconstructions $\overline{\nu}$ (section 7.8). It might also be argued that it is more suited to unbounded horizon problems on the grounds that it looks like an "online" reconstruction algorithm: upon the arrival of a new point $\mathbf{t}^{(n)}$ with missing values, a new global reconstruction of the whole trajectory is immediately obtained incrementally, at the small cost of computing the multiple pointwise reconstructions of $\mathbf{t}^{(n)}$ and choosing the closest one to $\mathbf{t}^{(n-1)}$. However, the dynamic programming algorithm is also able to incrementally update its current reconstruction—but rather than one, it keeps ν_n current reconstructions, of which one is the current global optimum. Again, this requires on the average $\overline{\nu}$ times more operations and storage at step n.

[25] The unlikely case where L and R are equidistant from T does not alter the point we are making.

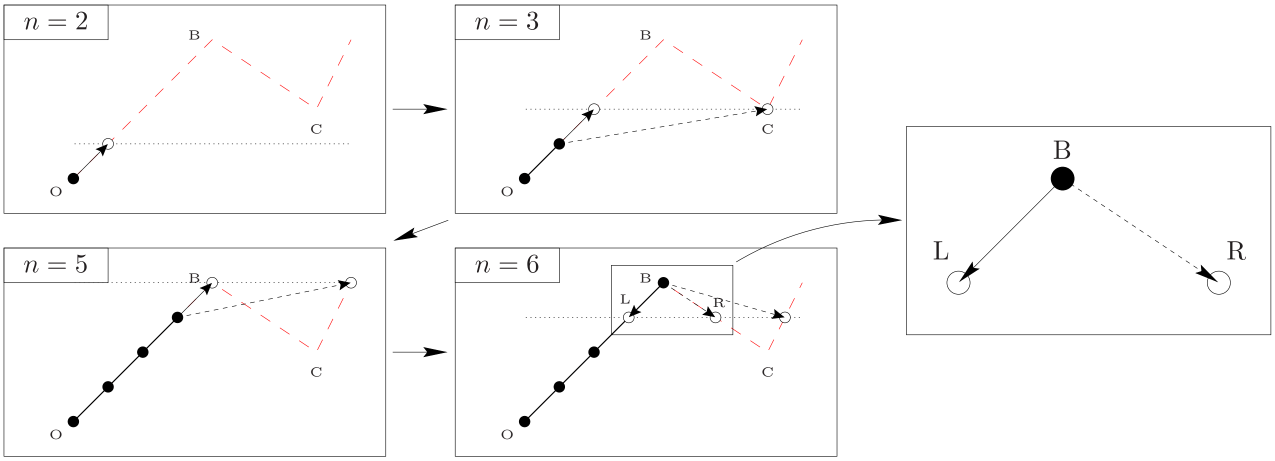


Figure 7.13: The greedy algorithm leads to reconstructed trajectories that retrace themselves. The problem is to find t_1 given t_2, as in fig. 7.12, with a slightly modified mapping. Starting at point O, the greedy algorithm successfully reconstructs the trajectory till point B but then starts tracing back instead of proceeding towards C. The inset shows that at B it cannot select R because it is farther than L from B. Starting the greedy algorithm from a point to the right of B would avoid this problem in this particular case, but in general this problem is likely to occur. *Key to the graph:* at each step n of the greedy algorithm, the horizontal dotted line indicates the known value of t_2 (t_1 runs in abscissas and t_2 in ordinates; the axes are omitted for clarity). The crossings of this line with the underlying mapping (long-dashed line) give the (ideal) candidate pointwise reconstructions, marked as white circles. The candidate which is chosen is marked with a solid arrow and the ones not chosen are marked with a dashed arrow. The so-far reconstructed trajectory is marked with a solid line and black circles. Not all steps n are shown.

Whatever its potential advantages, the greedy algorithm has a great risk of obtaining poor reconstructions. The experiments of chapters 9–10 show that this risk is likely to occur and that its effect on the quality of the reconstruction is strong. Thus we do not recommend the use of the greedy algorithm.

7.9.5 Many missing variables

If at some point n there are few variables present, then the missing variables are likely to be strongly underdetermined (as in section 7.3.5): they can take values compatible with those of the present variables in a continuous manifold, rather than a finite set. Practical implementations using Gaussian mixtures of our method do not break down in this situation: they will still produce the modes of the conditional distribution as candidate reconstructions. Effectively, this is a finite sample of such manifold, and a quantisation error appears. This error can be reduced by using more mixture components, but at a cost that grows exponentially with the observed space dimension.

In the extreme case where all variables are missing at point n, the modes are now the modes of the joint density function and can be computed once and stored for subsequent points where all variables are missing, to save computer time. If the density is a Gaussian mixture, another possibility with nil computational cost is simply to use all the component centroids, since in principle they should all lie in high-density areas of the observed space. This will also produce a finer discretisation of the observed data manifold, since there will be fewer modes than centroids (for the mixtures that verify conjecture 8.1).

7.9.6 Discontinuities

Discontinuities may occur in isolated instances in data sets that are otherwise continuous (as a function of the experimental conditions):

- Due to **undersampling**: if the frequency at which the observed variables are sampled is not high enough compared to the rate at which they are changing we may get samples that are too widely separated (see fig. 7.14(left, top)). This typically occurs when a signal changes slowly most of the time but has occasional short-duration changes; for economy, the signal is sampled at the slow rate and therefore isolated discontinuities occur. Since generally one cannot predict when a fast change is going to occur,

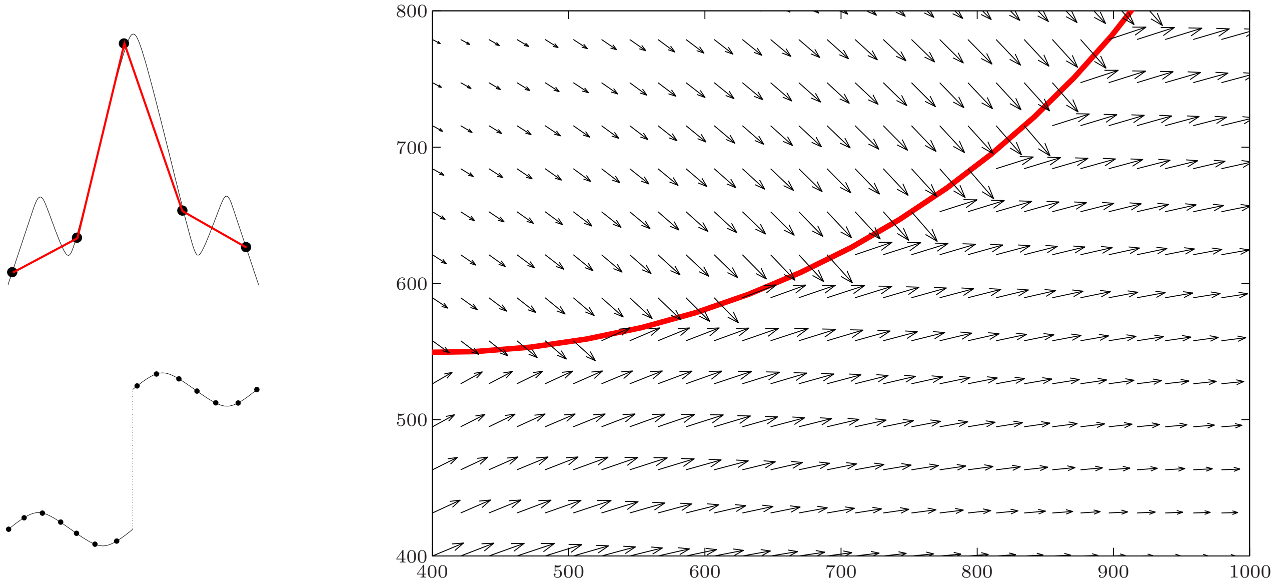


Figure 7.14: Isolated discontinuities in continuous data sets. *Left, top*: undersampled signal. *Left, bottom*: intrinsic discontinuity. *Right*: a front in a wind field (figures are in kilometres).

the only foolproof solution is to sample it at the high rate. An example from EPG signals was described in section 5.5 (fig. 5.20).

- Due to **intrinsic discontinuities** in the data as in fig. 7.14(left, bottom). An example with two-dimensional experimental conditions is a *front* in a wind field. Ordinarily, the wind vector varies continuously as a function of the location on an area (considered flat) of the Earth. A front is the interface between two air masses of different density and temperature and is generated by complex atmospheric dynamics. A front is frequently accompanied by marked changes in wind direction and relative humidity, as well as by low barometric pressure (a pressure trough) and considerable cloudiness and precipitation. Therefore, at a typical observation scale (e.g. that of a scatterometer) they appear as a discontinuity along a curve (not necessarily straight) in that area (although the front extends in the vertical direction too), where the wind vector changes its direction (but not its modulus) abruptly. Fig. 7.14(right) shows schematically a front.

  Another example of intrinsic discontinuities happens in geophysical inverse problems. For example, the Earth properties generally change abruptly between soil and rock units. Traditional regularisation methods apply smoothness constraints and so cannot recover discontinuities in the structure.

Discontinuities in the data set to be reconstructed may potentially bring about a wrong global reconstruction in our algorithm, since effectively they violate the definition of continuity constraint and can confuse the dynamic programming search. It would be interesting to carry out controlled experiments where isolated discontinuities occur in a continuous data set to determine the sensitivity of the dynamic programming search to such discontinuities.

While undersampling discontinuities can be blamed to the data collection procedure, intrinsic discontinuities in otherwise continuous data pose challenging modelling difficulties. In a Bayesian setting for wind retrieval from scatterometer data (see section 7.10.2), Cornford et al. (1999b) define a prior distribution for wind fields via Gaussian processes that allows the inclusion of constrained singularities.

7.9.7 When is the method not applicable?

Our method is not applicable in the following situations:

Discrete observed variables In common with the rest of this thesis, we have assumed that the observed variables are continuous. Discretised variables are also acceptable (e.g. the EPG data of chapters 5 and 10) although there will be a quantisation error. Intrinsically discrete, or categorical, variables, are not acceptable because, even though probability models can be constructed, the definition of local mode makes no sense; only the global mode makes sense. Specific applications may exist where it is possible to extract several candidates from the predictive distribution, but in general the multiple pointwise reconstruction is not applicable.

Independent data If every data point $\mathbf{t}^{(n)}$ is independent of its neighbours $\mathbf{t}^{(n-1)}$, $\mathbf{t}^{(n+1)}$, etc. then no constraint across data points exists and consequently only multiple pointwise reconstruction is possible, not global reconstruction. Examples are i.i.d. data or shuffled data (where the original ordering of the data has been irreversible altered). Such data sets are fine for training the joint pointwise density model, though.

The continuity or smoothness constraints are very general in form and should apply to a variety of physical and engineering reconstruction problems. Seeking other forms of constraints or interpoint dependences that may be applicable to other problems seems a worthwhile endeavour.

7.9.8 Reconstruction as a preprocessing step

If the missing data reconstruction is a preprocessing step for some other processing that ordinarily operates on the complete data, then the whole procedure may be suboptimal but faster than marginalising over the missing variables. For example, in a classification task such as speech recognition, one wants to compute $p(C_i^{(n)}|\mathbf{t}^{(n)})$ where $C_i^{(n)}$ is a phoneme class and $\mathbf{t}^{(n)}$ an acoustic feature vector (Rabiner and Juang, 1993). Using a hidden Markov model, such probabilities can be computed for every point n in an utterance and an optimal transcription $C^{(1)}, \dots, C^{(N)}$ obtained with the Viterbi algorithm. However, if some features are deemed to be missing (due to the presence of noise, for example; see section 7.10.6), then the correct thing to do is to use $p(C_i^{(n)}|\mathbf{t}^{(n)}_{\mathcal{P}^{(n)}})$, i.e., to marginalise over the missing variables the joint distribution $p(C_i^{(n)}, \mathbf{t}^{(n)}_{\mathcal{M}^{(n)}}|\mathbf{t}^{(n)}_{\mathcal{P}^{(n)}})$ of what is unknown given what is known. If we reconstruct the data as $\hat{\mathbf{t}}^{(n)} \stackrel{\text{def}}{=} (\mathbf{t}^{(n)}_{\mathcal{P}^{(n)}}\ \hat{\mathbf{t}}^{(n)}_{\mathcal{M}^{(n)}})$ with $\hat{\mathbf{t}}^{(n)}_{\mathcal{M}^{(n)}}$ given by $\mathbf{t}^{(n)}_{\mathcal{P}^{(n)}}$ and then use $p(C_i^{(n)}|\hat{\mathbf{t}}^{(n)})$ instead, we are implicitly wasting all the information contained in the distribution $p(\mathbf{t}^{(n)}_{\mathcal{M}^{(n)}}|\mathbf{t}^{(n)}_{\mathcal{P}^{(n)}})$:

- Marginalisation:

$$p(C_i^{(n)}|\mathbf{t}^{(n)}_{\mathcal{P}^{(n)}}) = \int p(C_i^{(n)}, \mathbf{t}^{(n)}_{\mathcal{M}^{(n)}}|\mathbf{t}^{(n)}_{\mathcal{P}^{(n)}})\, d\mathbf{t}^{(n)}_{\mathcal{M}^{(n)}} = \int p(C_i^{(n)}|\mathbf{t}^{(n)}_{\mathcal{M}^{(n)}}, \mathbf{t}^{(n)}_{\mathcal{P}^{(n)}}) p(\mathbf{t}^{(n)}_{\mathcal{M}^{(n)}}|\mathbf{t}^{(n)}_{\mathcal{P}^{(n)}})\, d\mathbf{t}^{(n)}_{\mathcal{M}^{(n)}}.$$

- Reconstruction: $p(C_i^{(n)}|\hat{\mathbf{t}}^{(n)}_{\mathcal{M}^{(n)}}, \mathbf{t}^{(n)}_{\mathcal{P}^{(n)}})$ where $\hat{\mathbf{t}}^{(n)}_{\mathcal{M}^{(n)}}$ is the mean, mode or some other convenient statistic of the distribution $\mathbf{t}^{(n)}_{\mathcal{M}^{(n)}}|\mathbf{t}^{(n)}_{\mathcal{P}^{(n)}}$.

Effectively, the reconstruction method is equivalent to the marginalisation where the distribution of missing features given present features has been replaced by a delta function at $\hat{\mathbf{t}}^{(n)}_{\mathcal{M}^{(n)}}$, thus throwing away all the distribution $\mathbf{t}^{(n)}_{\mathcal{M}^{(n)}}|\mathbf{t}^{(n)}_{\mathcal{P}^{(n)}}$. Cooke et al. (2001) have demonstrated empirically the superiority of the marginalisation approach for classification in the context of recognition of occluded speech.

However, strictly what we have shown is that reconstructing and then classifying is worse only when the reconstruction is done on a point-by-point basis, i.e., considering the speech frames independent from each other—which they are not. Thus, there may indeed be a benefit in using a global, utterance-wide constraint to reconstruct the whole speech segment—ideally recovering the original speech—and then classifying it; in other words, reconstructing $\mathbf{t}^{(n)}_{\mathcal{M}^{(n)}}$ from $\{\mathbf{t}^{(n)}_{\mathcal{P}^{(n)}}\}_{n=1}^N$, not just from $\mathbf{t}^{(n)}_{\mathcal{P}^{(n)}}$.

7.9.9 Bump-finding rather than mode-finding

If we want to pick representative points of an arbitrary density $p(\mathbf{t})$, using a mode as a reconstructed point is not appropriate in general because *locally* the optimal value (in the L_2 sense) is the mean, as shown in section 7.3.3. That is, if a function is multivalued it will have several branches; in a neighbourhood around a branch the function becomes univalued and so the mean of that branch would be L_2-optimal. Besides, the modes are very sensitive to the idiosyncrasies of the training data and in particular to outliers—they are not robust statistics. This suggests that, when the conditional distribution is multimodal, we should look for *bumps*[26] associated with the correct values and take the means of these bumps as reconstructed values instead

[26] Bumps of a density function $p(\mathbf{t})$ are usually defined as continuous regions where $p''(\mathbf{t}) < 0$, while modes are points where $p'(\mathbf{t}) = 0$ and $p''(\mathbf{t}) < 0$ (Scott, 1992). However, there does not seem to be agreement on the definition of "bumps," "modes" or even "peaks" in the literature. Titterington, in his comments to Friedman and Fisher (1999), claims that "bump-hunting" has tended to be used specifically for identifying and even counting the number of modes in a density function, rather than for finding fairly concentrated regions where the density is comparatively high.

of the modes. If these bumps are symmetrical then the result will coincide with picking the modes, but if they are skewed, they will be different.

How to select the bumps and their associated probability distribution is a difficult problem not considered here. A possible approach would be to decompose the distribution $p(\mathbf{t})$ as a mixture:

$$p(\mathbf{t}) = \sum_{k=1}^{K} p(k)p(\mathbf{t}|k) \tag{7.5}$$

where $p(\mathbf{t}|k)$ is the density associated with the kth bump. This density should be localised in the space of $\mathbf{t}$ but can be asymmetrical. If $p(\mathbf{t})$ is modelled by a mixture of Gaussians (as is the case in chapter 8) then the decomposition (7.5) could be attained by regrouping Gaussian components. What components to group together is the problem; it could be achieved by a clustering algorithm—but this is dangerous if one does not know the number of clusters (or bumps) to be found. Computing then the mean of each bump would be simple, since each bump is a Gaussian mixture itself.

This approach would avoid the exhaustive mode finding procedure of chapter 8, replacing it by a grouping and averaging procedure, which would probably be much faster (and it could be accelerated by discarding low-probability components from the Gaussian mixture, as explained in section 8.2.3).

These ideas operate exclusively with the functional form of a Gaussian mixture as starting point, as do our mode-finding algorithms of chapter 8. Bump-finding methods that work directly with a data sample exist, such as algorithms that partition the space of the $\mathbf{t}$ variables into boxes where $p(\mathbf{t})$ (or some arbitrary function of $\mathbf{t}$) takes a large value on the average compared to the average value over the entire space, e.g. PRIM (Friedman and Fisher, 1999); or some nonparametric and parametric clustering methods, e.g. scale-space clustering (Wilson and Spann, 1990; Roberts, 1997) or methods based on morphological transformations (Zhang and Postaire, 1994).

7.9.10 Reconstruction via dimensionality reduction

If a latent variable model is used as density model (with latent variables $\mathbf{x}$), one could think of performing missing data reconstruction via dimensionality reduction: $\mathbf{t}_{\mathcal{P}} \to \mathbf{x} \to \mathbf{t}_{\mathcal{M}}$. That is, if $\mathbf{t}_{\mathcal{M}}$ are missing and $\mathbf{t}_{\mathcal{P}}$ are present:

1. Reduce dimensionality by picking a representative point of $p(\mathbf{x}|\mathbf{t}_{\mathcal{P}}) = \int p(\mathbf{x}, \mathbf{t}_{\mathcal{M}}|\mathbf{t}_{\mathcal{P}})\, d\mathbf{t}_{\mathcal{M}} = \int p(\mathbf{x}|\mathbf{t})p(\mathbf{t}_{\mathcal{M}}|\mathbf{t}_{\mathcal{P}})\, d\mathbf{t}_{\mathcal{M}}$.
2. Map that point onto $\mathcal{T}$ using the mapping $\mathbf{f}$ from latent onto observed space ($p(\mathbf{t}|\mathbf{x})$ is unimodal centred in $\mathbf{f}(\mathbf{x})$, so it has no effect).

This will not work well except when $p(\mathbf{x}|\mathbf{t}_{\mathcal{P}})$ is sharply unimodal, that is, $\mathbf{t}_{\mathcal{P}}$ strongly constrains $\mathbf{t}_{\mathcal{M}}$ to lie in a small region. Usually $\mathbf{x}|\mathbf{t}_{\mathcal{P}}$ will be multimodal and therefore step (1) is translating the problem of finding a multivalued relationship $\mathbf{t}_{\mathcal{P}} \to \mathbf{t}_{\mathcal{M}}$ to that of a multivalued dimensionality reduction $\mathbf{t}_{\mathcal{P}} \to \mathbf{x}$! Besides, step (2) will produce a value not just for $\mathbf{t}_{\mathcal{M}}$ but also for $\mathbf{t}_{\mathcal{P}}$, which may differ from the actual value of $\mathbf{t}_{\mathcal{P}}$.

In fact,

$$p(\mathbf{t}_{\mathcal{M}}|\mathbf{t}_{\mathcal{P}}) = \int_{\mathcal{X}} p(\mathbf{t}_{\mathcal{M}}, \mathbf{x}|\mathbf{t}_{\mathcal{P}})\, d\mathbf{x} = \int_{\mathcal{X}} p(\mathbf{t}_{\mathcal{M}}|\mathbf{x}, \mathbf{t}_{\mathcal{P}})p(\mathbf{x}|\mathbf{t}_{\mathcal{P}})\, d\mathbf{x} = \int_{\mathcal{X}} p(\mathbf{t}_{\mathcal{M}}|\mathbf{x})p(\mathbf{x}|\mathbf{t}_{\mathcal{P}})\, d\mathbf{x}$$

where we have used the axiom of local independence in the last equality. Therefore, the procedure is equivalent to collapsing $\mathbf{x}|\mathbf{t}_{\mathcal{P}}$ onto a delta function centred on $\hat{\mathbf{x}}$, thus throwing away all the probability mass not in $\hat{\mathbf{x}}$. For this same reason, in general it is not convenient to apply the constraints to the latent variables rather than to the observed ones (besides, for GTM the modes of $\mathbf{x}|\mathbf{t}_{\mathcal{P}}$ are not well defined since its latent space is discretised; see section 7.9.7).

What does makes sense for the purposes of dimensionality reduction exclusively is to use the conditional distribution

$$p(\mathbf{x}|\mathbf{t}_{\mathcal{P}}) = \frac{p(\mathbf{x}, \mathbf{t}_{\mathcal{P}})}{p(\mathbf{t}_{\mathcal{P}})} = \frac{\int p(\mathbf{x}, \mathbf{t})\, d\mathbf{t}_{\mathcal{M}}}{\int p(\mathbf{x}, \mathbf{t})\, d\mathbf{t}_{\mathcal{M}}\, d\mathbf{x}}$$

if some data are missing. We can then define a corresponding dimensionality reduction mapping $\mathbf{x} \stackrel{\text{def}}{=} \mathbf{F}(\mathbf{t}_{\mathcal{P}})$, as in section 2.9.1. This mapping will often be multivalued (for latent variable models that allow multimodal distributions), since $\mathbf{t}_{\mathcal{P}}$ cannot in general fully determine $\mathbf{x}$, and the same ideas developed in this chapter (multiple pointwise dimensionality reduction, continuity constraints, etc.) are applicable with little modification.

One such modification would concern the definition of what a mode of $p(\mathbf{x}|\mathbf{t}_{\mathcal{P}})$ is for models that sample the latent space, such as GTM (since effectively the latent space is then discrete), but we do not pursue this issue here.

7.9.11 Summary and further work

In summary, we have showed that the modes of the conditional distribution of the missing variables given the present ones are potentially plausible reconstructions of the missing values, and that the application of local continuity constraints—when they hold—can help to recover the actually plausible ones. We could call this approach *modal regression* or *modal reconstruction* (with constraints), in analogy with the standard definition of regression in terms of the mean of the conditional distribution of missing given present data. Our method has the following advantages:

- It is applicable to varying patterns of missing data: by using a joint probability model, the observed variables are treated symmetrically, unlike methods based on function approximators or conditional distribution approximators, which treat each variable either always as a predictor or always as a response. Predictive distributions for the missing data can be flexibly constructed as the conditional distribution.

  However, if the pattern of missing data is constant (but the corresponding mapping is multivalued), one can simply model the conditional distribution of inputs given outputs rather than the joint distribution of inputs and outputs—the latter being a harder problem. The conditional distribution is then used to provide the modes, as usual. Section 7.11.3 describes some conditional distribution models, such as mixture density networks.

- It deals by design with multivalued mappings, representing all solution branches and choosing the right branch only at reconstruction time. This is unlike standard function approximators, which transform the multivalued mapping into a univalued one by either selecting always a given branch (irreversibly losing the others) or by averaging branches (which often results in a non-solution mapping).

- For mapping inversion problems, the inverse mapping can be constructed from measured input-output data, without knowledge of the functional form of the forward system—which can sometimes be difficult to obtain (e.g. the acoustic-to-articulatory mapping of chapter 10). And since the pattern of missing data is constant, one can simply model the conditional distribution of inputs given outputs rather than their joint distribution.

  If the forward mapping is not known analytically, it is worth to also approximate it from measured input-output data for two reasons: (1) to use it as a double-check that the candidate inverse points produced by another method (in our case, the conditional distribution of missing variables given present ones) are indeed inverse points (i.e., map correctly to the present values); and (2) to include it in the constraint for dynamic programming minimisation.

- It is insensitive to time warping, i.e., to reparametrisations of the trajectory, because the continuity constraint is the arc length—a geometric invariant.

- It requires no assumption about the reason why or how the data are missing, it just needs to know what data are missing. Although such information, if available, might be used to further constrain the candidate reconstructions.

- It is modular (fig. 7.15): joint density model, mode finding in conditional distributions, constraint minimisation by dynamic programming; and different algorithms, models or definitions may be used for each module.

- It can also give confidence regions for each reconstructed value, derived from the Hessian at the corresponding mode that was selected.

- Multiple pointwise reconstruction by the conditional distribution is robust: no matter what values the present variables may have, one can compute a conditional distribution on them (at least for the Gaussian mixture density model) and locate its modes. This is very helpful in situations for which the forward mapping is nearly singular (in which case numerical inversion is unstable) or when the present values are just out of their domain due to noise (in which case a reconstruction strictly does not exist).

And the following disadvantages:

- For general patterns of missing data, the method performs extremely robustly, but for constant patterns of missing data (as in regression problems), it is sensitive to the smoothness of the density model: the conditional distribution can contain spurious modes that result in suboptimal reconstructed trajectories with low constraint value. In mapping inversion problems, the effect of spurious modes may be eliminated by using a forward constraint.
- Density estimation in high dimensions is difficult due to the curse of the dimensionality.
- When many variables are missing, it can have a high computational cost at reconstruction time due to the mode-finding step.

The method should not be used:

- As a replacement for universal mapping approximators (e.g. neural networks) in univalued mapping approximation problems (e.g. in forward mappings), since universal mapping approximators are faster and more reliable at both training and reconstruction time.
- When the constraint assumptions do not hold, e.g. continuity does not hold in a shuffled sequence. Care must be taken too when continuity does hold but there exist isolated discontinuities (either intrinsic or due to undersampling).

Several directions for future theoretical work appear specially fruitful to us: the evaluation of constraints other than continuity (implemented as polygonal curve length); the extension to multidimensional constraints—e.g. reflecting spatial structure (one- to three-dimensional) as well as temporal (one-dimensional)—and the development of an efficient multidimensional constraint minimisation algorithm; the implementation of continuous reconstruction in real time (unbounded horizon); the use of bump means rather than modes for multiple pointwise reconstruction; and the efficient estimation of smooth density models in high dimensions.

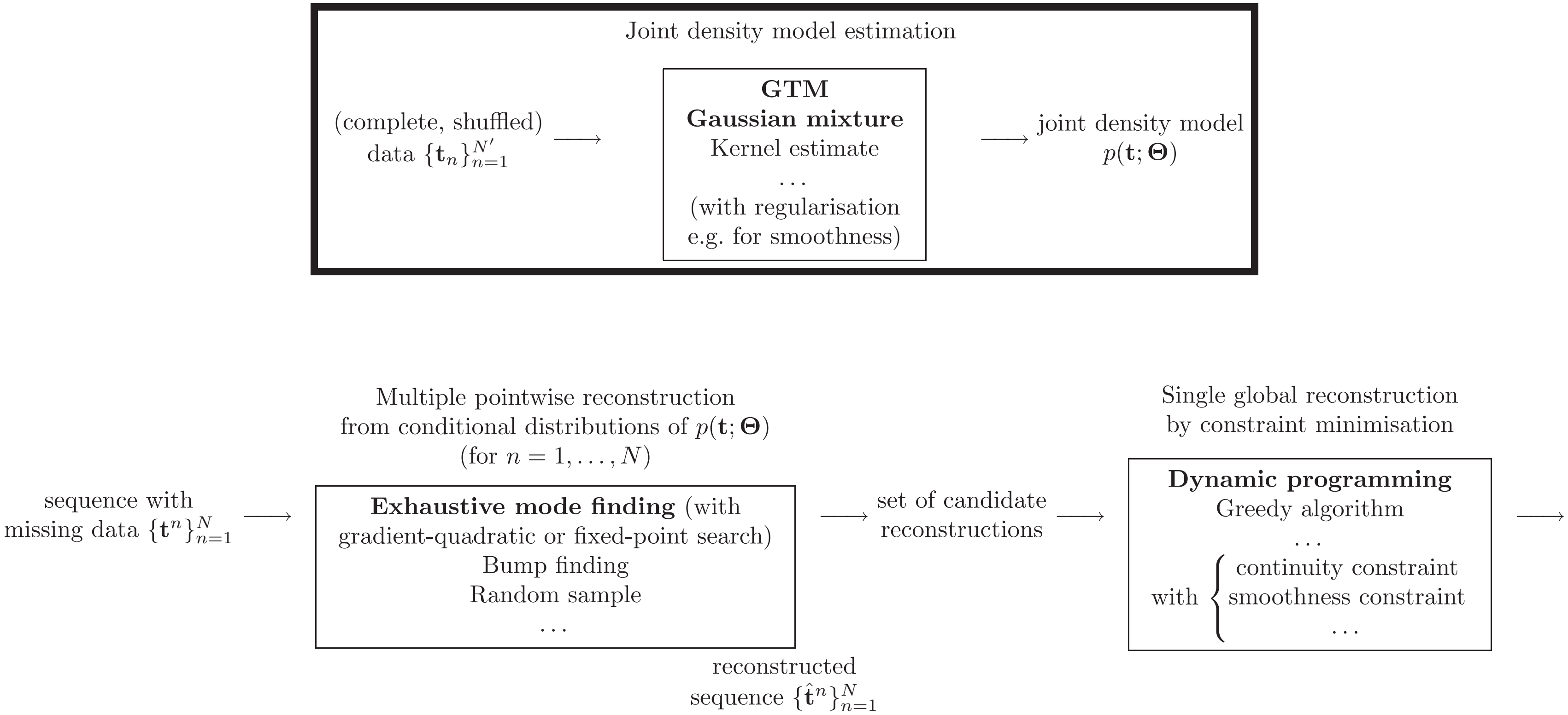


Figure 7.15: Modular structure of the missing data reconstruction approach. The boxes represent modules that admit different implementations, such as the ones given; the ones recommended are in boldface. The density estimation stage, enclosed in a thick box, takes place offline.

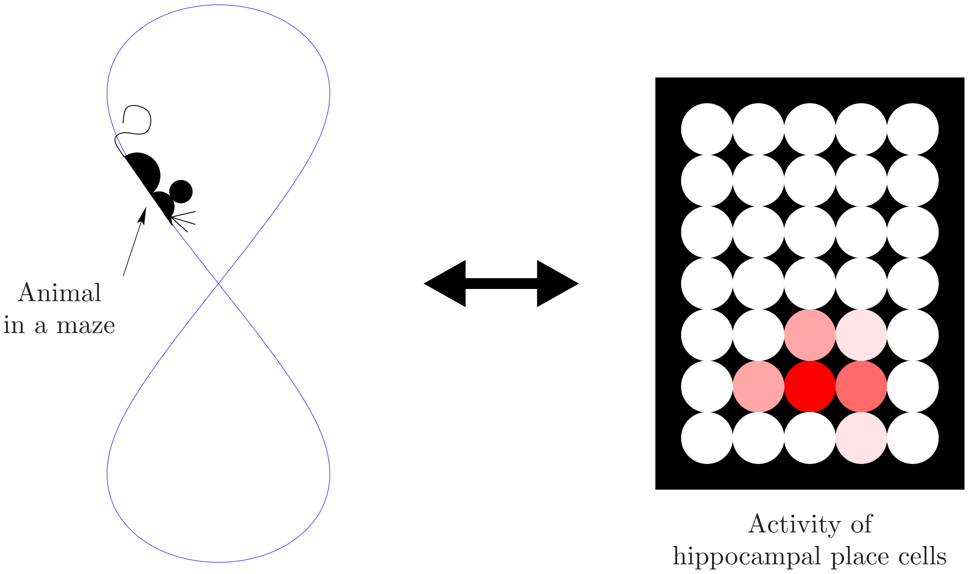


Figure 7.16: Reconstruction from hippocampal place cells. After the rat has explored the maze (represented by the 8-shaped line) for some time, a population of hippocampal CA1 place cells contains a representation of it: a given place cell tends to fire strongly only when the rat is at or near a certain location in the maze.

7.10 Possible applications

We believe the method has potential practical utility and it would be a personal pleasure to see it implemented in real situations. We have identified several reconstruction problems in the literature that are described below. Mapping inversion problems are clear candidate applications and abound. However, problems with arbitrary missing data patterns where constraints exist do not seem so abundant, so we have the case of a method in search of a problem!

7.10.1 Decoding neural population activity: reconstruction from hippocampal place cells

The brain contains populations of neurons that code as activity levels (spike trains) certain physical variables, such as the orientation of a line in the visual field or the location of the body in space. We can record the simultaneous activity of many neurons with an electrode array and try to reconstruct or decode from it the values of the physical variables. This poses a mapping inversion problem (often referred to as an inverse problem). Reconstruction can elucidate how much information about the physical variables is actually present in the population and how the brain might use it to solve computational problems such as object recognition and navigation. Practically, it can be applied to generate control signals based from the neural representation (e.g. to control a robot arm; Nicolelis, 2001).

Here we consider the particular problem of reconstructing the trajectory of a rat freely moving in a maze from the activity of hippocampal CA1 place cells (see Zhang et al., 1998; Brown et al., 1998; and references therein). If a rat is left in a new environment, it will explore it for a while, after which any given place cell tends to fire strongly when the rat is at or near a certain physical location but not when it is far from it (see fig. 7.16). Thus, such cells build a map of the environment. While the rat location is the main variable that affects the cell's firing rate, other variables can modulate it, notably the rat speed and direction of movement and the phase of its theta rhythm; in particular, place cells fire above background only when the rat is moving.

The reconstruction problem can be formalised as follows (typical experimental values are given in parentheses). At a given instant, the present variables are the instantaneous firing rates $\mathbf{f} = (f_1, \dots, f_C)^T$ of the C cells being measured (around 100) and the missing variables are the coordinates[27] $\mathbf{x}$ (two- or three-dimensional) of the rat's position; a further missing variable can be the (vectorial) velocity $\mathbf{v}$ of the rat. The position and velocity can be measured with a video camera (sampling at around 20 Hz) and the firing rates are obtained as spike counts over a time window of fixed width τ (around 1 s). Ideally, the intrinsic dimensionality of the

[27] Not to be confused with the notation of chapter 2, where $\mathbf{x}$ meant latent variable and $\mathbf{f}$ mapping latent $\to$ observed.

joint variables should be that of the maze (e.g. 1 for a linear maze). The mapping $\mathbf{x} \to \mathbf{f}$ is multivalued for self-intersecting mazes (e.g. for an 8-shaped maze); the mapping $\mathbf{f} \to \mathbf{x}$ may not be univalued since sometimes a cell responds to different locations; the spike trains are noisy; and continuity constraints hold, at least for the position and velocity as a function of time (the experimental condition), assuming a high enough sampling rate. Thus, our method should be applicable, given a training set of pairs $(\mathbf{x}_n, \mathbf{f}_n)$.

Several decoding methods have been used, e.g. based on radial basis functions. The most successful ones are based on probabilistic models and are often referred to as Bayesian reconstruction methods[28] (Sanger, 1996; Zhang et al., 1998; Oram et al., 1998; Brown et al., 1998). Here we analyse one proposed by Sanger (1996) and extended by Zhang et al. (1998), because it has some similarities with ours and because the analysis demonstrates the generality of our method, which only requires mild modelling assumptions about the problem. Sanger (1996) reconstructs the rat location $\mathbf{x}_n$ given the firing rate $\mathbf{f}_n$ at time n as the (global) mode of the conditional distribution

$$p(\mathbf{x}_n|\mathbf{f}_n) = \frac{p(\mathbf{f}_n|\mathbf{x}_n)p(\mathbf{x}_n)}{p(\mathbf{f}_n)}$$

where:

- Place cells are assumed independent given $\mathbf{x}$: $p(\mathbf{f}|\mathbf{x}) = \prod_{c=1}^{C} p(f_c|\mathbf{x})\ \forall \mathbf{x}$.
- $f_c|\mathbf{x}$ is modelled as an inhomogeneous Poisson process, that is, a Poisson distribution whose rate parameter λ_c is not constant (a common model for neuronal spike statistics). Specifically, $\lambda_c = \tau f_c(\mathbf{x})$, where τ is the time window width and $f_c(\mathbf{x})$ the firing rate map of cell c, i.e., the average firing rate of cell c when the animal is at position $\mathbf{x}$.
- $p(\mathbf{x})$ is set to the (normalised) number of times that the rat was found at position $\mathbf{x}$ during training; this requires quantising $\mathbf{x}$ (e.g. 256×256 grid).
- $p(\mathbf{f}_n)$ is a normalisation factor.

This is then the same strategy as that of section 6.2.3.8 (Bayesian inversion with locally independent inverse problems): a forward mapping is modelled with a factorised conditional distribution $\mathbf{f}|\mathbf{x}$ which is "inverted" by Bayes' theorem.

Using $p(\mathbf{x}_n|\mathbf{f}_n)$ alone leads to high reconstruction errors due to erratic jumps often caused by low instantaneous firing rates, especially when the animal stops running ($v = 0$): if all cells stop firing, there is not enough information for accurate reconstruction[29]. Zhang et al. (1998) introduce a continuity constraint by a Markov assumption on $\mathbf{x}$: reconstruct $\mathbf{x}_n$ as the mode of $p(\mathbf{x}_n|\mathbf{f}_n, \mathbf{x}_{n-1})$, approximated as

$$p(\mathbf{x}_n|\mathbf{f}_n, \mathbf{x}_{n-1}) = \frac{p(\mathbf{x}_{n-1}|\mathbf{x}_n, \mathbf{f}_n)p(\mathbf{x}_n, \mathbf{f}_n)}{p(\mathbf{x}_{n-1}, \mathbf{f}_n)} \stackrel{\text{(a)}}{=} \frac{p(\mathbf{x}_{n-1}|\mathbf{x}_n)p(\mathbf{x}_n|\mathbf{f}_n)}{p(\mathbf{x}_{n-1}|\mathbf{f}_n)} \propto p(\mathbf{x}_n|\mathbf{f}_n)p(\mathbf{x}_{n-1}|\mathbf{x}_n)$$

by assuming in (a) that $p(\mathbf{x}_{n-1}|\mathbf{x}_n, \mathbf{f}_n) = p(\mathbf{x}_{n-1}|\mathbf{x}_n)$, i.e., that the activity $\mathbf{f}_n$ at the current time step cannot directly affect the position $\mathbf{x}_{n-1}$ at the preceding time step. The only new factor in this conditional probability is $p(\mathbf{x}_{n-1}|\mathbf{x}_n)$, which forces the current position, $\mathbf{x}_n$, not to be far away from the previous one, $\mathbf{x}_{n-1}$. They model $\mathbf{x}_{n-1}|\mathbf{x}_n \sim \mathcal{N}(\mathbf{x}_n, \sigma_n^2\mathbf{I})$ where $\sigma_n \propto v_n^\alpha$, $v_n \stackrel{\text{def}}{=} \|\mathbf{v}_n\|$ is the speed at time n (estimated in one of two ways: either as linearly dependent on $\mathbf{f}_n$, or as the average speed at each position $\mathbf{x}$ computed from previous data) and $\alpha = \frac{1}{2}$ for random walks and 1 for linear movements. If σ_n is too large the continuity constraint has little effect, but if it is too small it becomes too restrictive and the reconstructed position might get stuck and not change; an intermediate value can be found that reduces the reconstruction error. The same kind of continuity constraint also worked well for Brown et al. (1998), who used a nonlinear Kalman filter approach, where the rat's position is the state variable (which follows a random walk) and the spike train is the (nonlinear and discrete) observed process. However, they found experimentally that the path distribution had longer tails than a normal distribution.

We now compare the extended method of Zhang et al. (1998) with ours. Firstly, the fact that the extended method outperforms the one based on $\mathbf{x}_n|\mathbf{f}_n$ alone reinforces the beneficial role of continuity constraints. Then,

[28] Because they use Bayes' theorem to "invert" a conditional distribution that represents a forward mapping. But the adjective "Bayesian" does not seem appropriate since, strictly, Bayesian methods are associated with the use of probability distributions over parameters, which is not the case here. And, although some of the models have no free parameters (estimated from data), it would be arguable whether τ, $p(\mathbf{x})$, $f_c(\mathbf{x})$ and λ_c are parameters estimated from data or fixed parameters.

[29] This fact makes it important to model the modulation of the firing rate by the rat speed. Zhang et al. (1998) took $\mathbf{f}(\mathbf{x}, \mathbf{v})$ either independent of $\mathbf{v}$ or as $\mathbf{f}(\mathbf{x})(av + b)$ for constants a and b. Brown et al. (1998) took instead $\mathbf{f}(\mathbf{x}, \phi) = \mathbf{f}(\mathbf{x})e^{\beta \cos(\phi - \phi_0)}$ for constants β and ϕ_0, where ϕ is the theta rythm, but experimentally found a higher reconstruction error than with $\mathbf{f}(\mathbf{x})$ alone.

the extended method can be seen as a greedy reconstruction, where only one mode is computed and only the local constraint between $\mathbf{x}_n$ and $\mathbf{x}_{n-1}$ is considered (which can lead to wrong trajectories, as discussed in section 7.9.4). Finally, their density model $p(\mathbf{f}|\mathbf{x})p(\mathbf{x})$ forces some simplistic assumptions about $p(\mathbf{f}|\mathbf{x})$ and $p(\mathbf{x})$ (in common with conditional modelling methods, section 7.11.3), e.g.: discretising $\mathbf{x}$; assuming conditional independence of the firing rates given $\mathbf{x}$; modelling the firing rate distribution as Poisson (Treves et al., 1999 discuss other distributions, e.g. based on truncated normals); the prior distribution on $\mathbf{x}$, as if the rat preferred some locations to others, seems arbitrary; particularly risky are the assumptions about the speed of the rat (and about σ_n) in the continuity constraint, which our global constraint does not require, and about the modulation of the firing rate by the rat speed. These assumptions are relaxed in our method by using a universal density approximator trained with joint data for $\mathbf{x}$, $\mathbf{v}$ and $\mathbf{f}$. Any variable suspected to play a role can be included without forcing an assumption on its distribution or on its relation with other variables, other than that dictated by the density estimate.

7.10.2 Wind vector retrieval from scatterometer data

In section 6.2.4.2 we described the inverse problem of obtaining the wind field $\mathbf{U} = (\mathbf{u}_i)$ on a region of the Earth, $\mathbf{u}_i$ being the two-dimensional wind vector at location i, given a backscatter field $\mathbf{\Sigma}^0 = (\boldsymbol{\sigma}_i^0)$ measured by a satellite, $\boldsymbol{\sigma}_i^0$ being the three-dimensional backscatter vector. We showed that this inverse problem—which is subject to two-dimensional continuity constraints—can be factorised into independent mapping inversion problems. Therefore our method should be applicable.

The forward mapping $\mathbf{u} \to \boldsymbol{\sigma}^0$ is univalued and can be solved analytically or learned from data; the inverse mapping $\boldsymbol{\sigma}^0 \to \mathbf{u}$ can be multivalued. Training data consisting of pairs (wind vector, backscatter vector) is available from weather forecast organisations, such as those used in the NEUROSAT project mentioned in section 6.2.4.2. Two new problems arise here concerning the constraints: (1) the implementation of a two-dimensional continuity constraint (using finite differences) and efficient extension of the dynamic programming search; and (2) the possible presence of discontinuities along curves caused by fronts (described in section 7.9.6), that probably would disrupt the dynamic programming search.

7.10.3 Inverse kinematics and dynamics of a redundant manipulator

In section 9.3 we show that our method outperforms methods based on the conditional mean or in universal mapping approximators for the inverse kinematics problem of a very simple robot arm, having only two degrees of freedom and being restricted to planar movement. The method should be tested with a more realistic manipulator, having redundant degrees of freedom and three-dimensional movement. This will require a GTM model with a latent space of dimension $L = 3$. Another extension is to the case of inverse dynamics, where one considers not just the instantaneous manipulator position but also its velocity and acceleration.

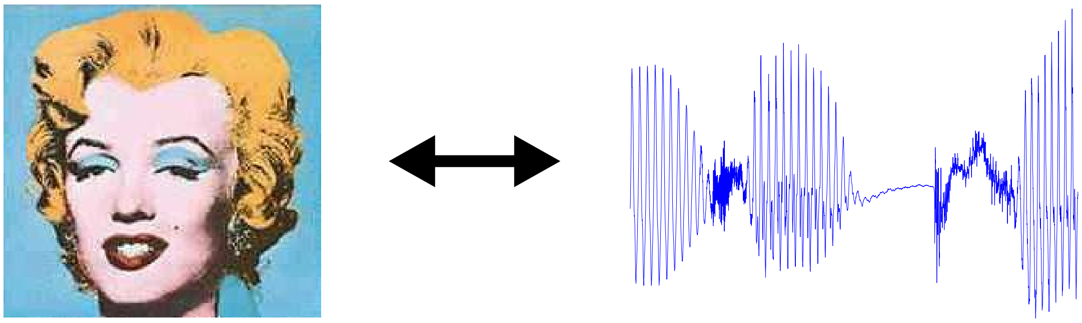
In this problem, the forward mapping is analytically known from the geometrical and inertial properties of the manipulator and so it can be used to generate abundant training data and to augment the global constraint for greater accuracy.

7.10.4 Audiovisual mappings for speech recognition

Human speech is bimodal: although we primarily rely on hearing to process speech, much information is received visually as well from the movements of the lips and other facial features. Evidence supporting this includes the fact that human recognition of speech in noisy environments[30] is enhanced by visual information because some sounds can be easily confused acoustically but not visually and vice versa. Another, classical demonstration of the bimodality of speech is the McGurk effect: when humans receive conflicting acoustic and visual stimuli, the perceived sound may not exist in either modality (e.g. a person that listens to /ba/ but sees the speaker saying /ga/ perceives something like /da/ instead). Therefore, it is better to process acoustic and visual speech jointly rather than separately. Audiovisual integration has applications in automatic robust speech recognition, human and automated lip reading, facial animation (human-computer interfaces, computer-aided instruction, cartoon animation, video games and multimedia telephony for the hearing impaired), lip synchronisation, joint audio-video coding, bimodal speaker verification and multimodal speech perception modelling. For a review see Chen and Rao (1998).

It is then important to study the mappings between acoustic and visual features of speech, i.e., to estimate the acoustics from the mouth shape (lip reading) and vice versa (facial animation); see fig. 7.17. In a statistical

[30] During spontaneous speech, occlusion can occur in the acoustic domain (e.g. noise), in the visual domain (e.g. speaker turns head) or in both.



facial features   acoustic signal

Figure 7.17: Audiovisual mappings for multimodal speech processing: to recover facial features such as the lips from the acoustics and vice versa.

learning approach the mappings are implemented between two sets of feature vectors: one for the speech, such as cepstral or filter-bank coefficients extracted from the waveform, and another one for the face, such as locations and dimensions of various landmarks (lip opening, cheek stretch, etc.) extracted from the visual stream. A separate problem that we do not deal with here but important for practical applications is the tracking and segmentation of facial features from a video sequence. To study the audiovisual mappings, there are databases of facial data (synchronised with speech) collected with tracking aids such as special makeup or metallic pellets attached to specific points in the face (whose location can be tracked by EMA, as in section 7.10.5). Some extra information that can be used by a human is difficult to capture in the measurements. For example, whether the teeth touch the lips or whether the tongue is visible.

The usual methods for construction of (temporal) mappings have been applied. Yehia et al. (1998) showed that a considerable part of the acoustic variability can be linearly predicted from the three-dimensional distortions of the face (and also that facial distortions are highly predictable from the movement of the vocal tract articulators, but not vice versa). But, surely, the mapping cannot be linear, and in fact Barker and Berthommier (1999) and Yehia et al. (1999) have showed improvements using neural networks. Lavagetto (1997) used time-delay neural networks to implement the mapping acoustic $\rightarrow$ lip shape in an audio-video synchronisation task. Chen and Rao (1998) jointly model the acoustic and visual features with a Gaussian mixture and use the conditional mean to define the appropriate mapping. Yamamoto et al. (1998) use hidden Markov models to implement the mapping acoustic $\rightarrow$ lip shape: the HMM is trained on the acoustic data as usual and specific lip shapes are associated with each HMM state (e.g. the average of the lip shapes for all frames associated with the same state).

The mapping between acoustic and facial features is many-to-many. Multiple phonemes can be associated with a single viseme[31]: the phonemes /p/, /b/ and /m/ are all produced by a closed mouth shape and may be visually indistinguishable. Conversely, a given phoneme may be associated with different visemes, because the vocal tract (which is not readily visible, except for occasionally the tongue, but is primarily responsible for the acoustics) can be controlled relatively independently of the lips and jaw. The reasons for the existing correlation of the acoustic and visual modes during speech production have been debated. Yehia et al. (1998) claim that configuring the vocal tract to produce the acoustic signal simultaneously deforms the face through positioning of the jaw, shaping of the lips and motion of the cheeks, and thus not just the lip movements but also those of a much larger region of the face are caused by the articulator motion. Barker and Berthommier (1999) argue that the correlation may be due to all parts of the speech apparatus (vocal tract and face) being driven in a coordinated manner to achieve a common goal rather than one causing the other. In any case, the fact is that acoustic and facial features are correlated and this allows (partial) reconstruction of one set given the other. Also, considerations of critical articulators and "don't care" values should apply here (section 10.1.1.6).

The facial features are subject to continuity constraints as a function of time (although not necessarily the acoustic ones, see section 10.1.1.3), and besides should have a low dimensionality since the face is a semirigid solid (if it was a rigid solid the dimensionality would be 3 at most, since all the facial features could be derived from the position of, say, the face centre-of-mass). This and the multivalued character of the audiovisual mappings suggest that our method could be a flexible way of representing both mappings for a

[31]Phonemes and visemes are the basic units of acoustic speech and mouth configurations, respectively. Thus, a viseme is the smallest visibly distinguishable unit of speech.

speech utterance.

Possible future work could be done with data[32] from the Institut de la Communication Parlée (ICP), Grenoble, France (Benoît et al., 1992). Their corpus consists of simultaneous measurements of the acoustic waveform (10 LSP coefficients) and of 15 features describing the configuration of face, lips and jaw, obtained using Chroma-Key video processing for several fully-voiced French VCVCV sequences, totalling some 13 000 frames. This data also provides an opportunity to study the effects on our algorithm of undersampling discontinuities, which we have observed in the lip position during bilabial plosives (e.g. /b/), in which lip closure occurs very fast. Performance should be reported in terms not just of reconstruction error but also of Pearson's product-moment correlation coefficient (eq. (10.1)) between the original and reconstructed data, as is customary in audiovisual mapping modelling. As a result, and using purely experimental measurements and probabilistic learning techniques, we would expect to characterise the extent to which the mappings are nonlinear and many-to-many and to determine to what extent it is possible to recover visual features from acoustic ones and vice versa.

7.10.5 Acoustic-to-articulatory mapping

The speech processing problem of the acoustic-to-articulatory mapping, to recover the vocal articulators' positions given the observed speech waveform, is explained in detail in chapter 10. There we apply our method to a version of this problem using EPG data, which is an incomplete articulatory representation of the vocal tract. More thorough experiments are desirable using measured articulatory data[33] that more completely characterise the vocal tract during an utterance, such as the 2D (midsaggittal) or 3D positions of several articulators sampled in time in a number of phonetic contexts. The instrumental techniques of X-ray microbeam and electromagnetic articulography (EMA or EMMA) (Hardcastle and Hewlett, 1999, chapter 12) are both able to provide this kind of data. The X-ray microbeam device solves the difficulties associated with conventional X-ray imaging (Hardcastle and Hewlett, 1999, chapter 11) of the whole vocal tract (e.g. as used in Fant, 1970), namely high radiation dose (which prevents extended imaging) and difficult segmentation of the articulator landmarks of interest from the image (automatic detection is unreliable). Two public databases exist that contain articulators' trajectories: the Wisconsin X-ray microbeam database and the still under construction MOCHA database.

The X-ray Microbeam Speech Production Database (Westbury, 1994) is a publicly available speech resource developed at the University of Wisconsin. It incorporates representations of lingual, labial, and mandibular movements, recorded in association with the sound pressure wave, for 57 normal, young adult speakers of American English, for a rich set of speech tasks: prose passages, counting and digit sequences, oral motor tasks, citation words, near-words, sounds and sound sequences, and sentences. Kinematic data recorded from each speaker represent the time-varying, mid-sagittally projected positions of a set of small pellets (gold beads, roughly 3 mm in diameter) glued to the articulators.

State-of-the-art ASR systems require more continuous speech data for training than is available in the Wisconsin database. The MOCHA (Multi-CHannel Articulatory) database (Wrench, 2000) is being prepared to provide a resource for training speaker-independent continuous ASR systems and for general coarticulatory studies. The planned dataset includes 40 speakers of English, each reading up to 460 TIMIT sentences (British version). The articulatory channels currently include EMA sensors directly attached to the vermillion border of the upper and lower lips, lower incisor (jaw), tongue tip (5–10 mm from the tip), tongue blade (2–3 cm posterior to the tongue tip sensor), tongue dorsum (2–3 cm posterior to the tongue blade sensor) and soft palate (10–20 mm from the edge of the hard palate). Additionally, a laryngograph provides voicing information and an electropalatograph provides tongue-palate contact data at 62 normalised positions on the hard palate defined by landmarks on the upper maxilla. EPG includes lateral tongue contact information which is missing from the EMA data. The sampling rates used are 200 Hz for EPG, 500 Hz for EMA and 16 000 Hz for laryngography.

7.10.6 Reconstruction of occluded speech[34]

In the acoustic-to-articulatory mapping problem, the speech waveform was assumed to be observed. A reconstruction problem of considerable practical interest occurs when the speech itself is partly occluded or

[32] I am grateful to Frédéric Berthommier and Jon Barker for their help in obtaining the data.

[33] It is possible to use articulatory data generated by articulatory models, as in fact many studies of the acoustic-to-articulatory mapping have (see chapter 10). However, articulatory models are always approximate. We can avoid any assumptions by using measured articulatory data.

[34] In this section, a point n in the sequence (the utterance) is called a *frame* and the observed variables are called *acoustic features*, as is common in the literature of automatic speech recognition.

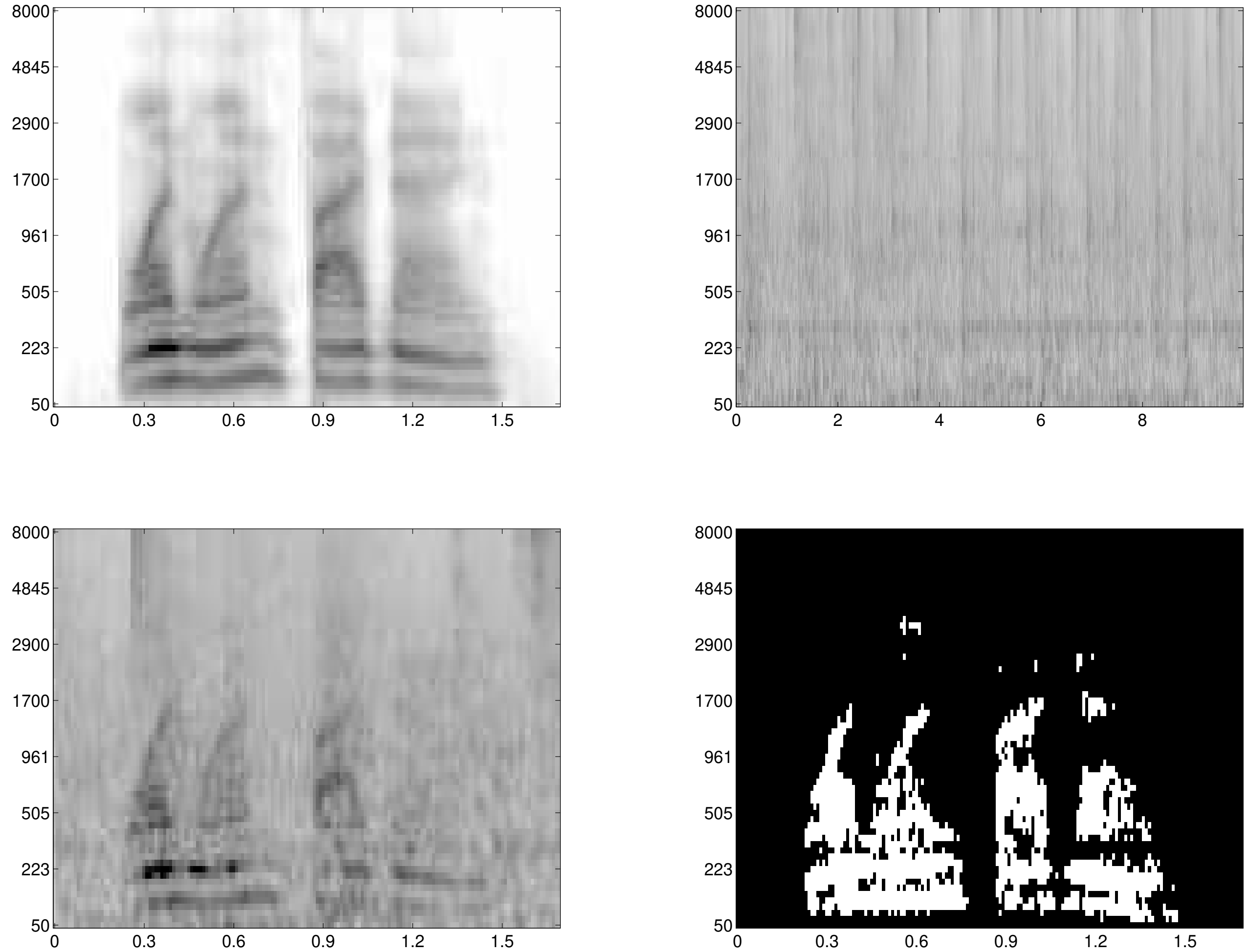


Figure 7.18: Speech occluded by noise (data kindly provided by Ljubomir Josifovski). In all graphs, the horizontal axis is the time in seconds and the vertical axis the subband centre frequency in hertz. In the three spectrograms, the grey shade is proportional to the log-energy in dB at the corresponding frequency and time. *Left, top*: spectrogram of the clean, non-occluded speech for the utterance "one one five nine". *Right, top*: spectrogram of the occluding signal (factory noise). *Left, bottom*: spectrogram of the occluded speech. *Right, bottom*: ideal pattern of missing data (mask) derived from the spectrogram of the occluded speech by comparing it with the nonoccluded speech; black (white) areas represent missing (present) data.

corrupted by a different acoustic signal, such as background noise. Note that here all observed variables are measured: it is just that some of them are severely corrupted and thus effectively missing (how to know which ones are missing is explained below). In this case, and assuming that the acoustic features (observed variables) are related to a collection of speech frequencies (obtained for a time window of fixed width), we will have a reconstruction problem with a genuinely varying missing data pattern—since different features may be missing at different times, depending on the time course of the occluding signal. A naive signal processing approach is to apply a filter to the occluded speech to remove the noise, but this only works well when the noise is stationary and its characteristics known, so that it can be modelled and thus a specific filter designed (e.g. a bandpass filter); this is not the case in, say, spontaneous speech in a noisy environment. There are several subproblems with occluded speech:

Recognition of occluded speech (see Cooke et al., 2001 for an excellent review) This is a classification problem, not a reconstruction one. For non-occluded, clean speech the state-of-the-art methods are (still!) hidden Markov models in any of their flavours with Viterbi decoding (Rabiner and Juang, 1993); however, the performance of HMMs deteriorates drastically as the speech signal-to-noise ratio decreases. In the recognition of occluded speech, or robust speech recognition, the missing variables are the occluded features and the frame classes (the phoneme that each frame represents) but one is interested in the latter only. As was discussed in section 7.9.8, reconstructing the speech and the classifying it as usual is worse[35] than attempting directly to classify conditionally on the present variables, as Cooke et al. (2001) have demonstrated empirically.

Reconstruction of occluded speech: interframe constraints Estimating a joint density model for the acoustic features is straightforward and there are many public databases for different languages and styles (telephone speech, news broadcasts, conversational speech, etc.). However, as mentioned in section 10.1.1.3, the acoustic features are in general not continuous, and so the key to allow the application of our method consists of finding (and formalising in terms of the acoustic features) a different interframe constraint. Even a cursory examination of a spectrogram shows that the frames are not independent; e.g. observe the formant trajectories in fig. 7.18, both in the clean and in the occluded speech (although interrupted by various discontinuities: silence, unvoiced sounds, etc.). Perhaps perceptual grouping based on Gestalt principles, as used in computational auditory scene analysis (Brown and Cooke, 1994; Cooke and Ellis, 2000), could be helpful here.

If the speech is detected by several sensors, the problem of reconstructing occluded speech can also be formulated as blind signal separation. In fact, independent component analysis has been applied to it with some success (sections 2.6.3 and 2.9.1.2), but only under the following conditions: that the number of signals is known, constant and not larger than the number of sensors; that the mixing is linear and constant. These stringent assumptions rarely hold for spontaneous speech. Our method does not require such assumptions and needs only one sensor.

The determination of the missing data pattern This is here a difficult problem, since the observed variables are not really missing but corrupted in various amounts by the unknown noise. Strategies used to identify regions of missing data are usually based on the assumption of energy additivity and on an estimate of the noise energy (such as the average of a few frames containing only noise). An example is *spectral subtraction*, where features whose energy is smaller than that of the noise estimate are deemed missing.

For the problem of recognition, although not for reconstruction (as noted in section 7.2.2), the speech features may be considered as uncertain rather than either present or missing, since the amount of occlusion varies in a continuum[36], from no occlusion (feature present) to full occlusion (feature missing). In this case, it makes sense to define the pattern of missing data in a soft, continuous way, rather than in a binary one (Barker et al., 2000). The element m_{nd} of the mask matrix $\mathbf{M} = (m_{md})$ of section 7.2.2 is then redefined as the probability that t_{nd} be present. It is possible to incorporate such uncertainty into a probabilistic treatment of the recognition problem but the problem with this approach is, naturally, to determine the values of $\mathbf{M}$.

7.11 Related work

The key aspects of our approach are the use of a joint density model (learnt in an unsupervised way); the use of the modes of the conditional distribution as multiple pointwise candidate reconstructions; the exhaustive mode search; the definition of a geometric trajectory measure derived from continuity constraints and its minimisation by dynamic programming. Several of these ideas have been applied earlier in the literature.

The conditional distribution obtained from a joint density model has been used in other work to derive mappings. While such work usually acknowledges the fact that the conditional distribution may be multimodal and that the conditional mean will not be a good summary of that distribution, in most cases a single point is still selected (typically the mean, the global mode or some approximation to it). Only the multiple imputation method of statistics has used multiple values from the conditional distribution, although not intended as candidate reconstructions but to inject randomness. Our use of all the modes as candidate reconstructions is new, as is its implementation with a Gaussian mixture derived from a latent variable model.

The use of constraints either at estimation time or minimised by dynamic programming at runtime is not new. It has been used in inverse problem theory and in mapping inversion problems, particularly the acoustic-to-articulatory mapping and the inverse kinematics of a robot arm.

Given the generality of the missing data reconstruction problem, related work spans many fields. We review some of it without claiming to be comprehensive.

[35] At least, on a frame-by-frame basis, i.e., reconstructing $\mathbf{t}^{(n)}_{\mathcal{M}^{(n)}}$ with no global constraints, using information in $\mathbf{t}^{(n)}_{\mathcal{P}^{(n)}}$ only.

[36] Although experimentally it is often found that at any frame most features are dominated by either the speech or the noise.

7.11.1 Statistical approaches to missing data and imputation methods

In this chapter we have dealt with the problem

> *given a data set with missing data, reconstruct it*

and we have assumed that a model for the data was available (perhaps obtained from a complete training set). The problem

> *given a data set with missing data, estimate parameters (and standard errors, p-values, tests, etc.) of a model of the data; or more generally, make inferences about the population from which the data come from*

has been the main concern of the statistical literature on missing data (for reviews see Little and Rubin, 1987; Little, 1992; Schafer, 1997). Such inferences must be done incorporating the missing data uncertainty; otherwise one will obtain too small standard errors, too low p-values or too high rates of type-I error. We discuss here two aspects of this literature in relation with our work: the mechanism of missing data and the statistical methods for missing data.

7.11.1.1 Missing data mechanisms

Statistical methods for missing data are based on assumptions about the mechanism whereby the data become missing. The pattern of missing data, given by the matrix $\mathbf{M}$ of section 7.2.2, is considered a random variable. The present data are[37] then $(\mathbf{T}_\mathcal{P}, \mathbf{M})$ and the complete data $\mathbf{T} = (\mathbf{T}_\mathcal{P}, \mathbf{T}_\mathcal{M})$. If a joint distribution of $(\mathbf{T}, \mathbf{M})$ with parameters $\boldsymbol{\Theta}$, $\boldsymbol{\Psi}$ is assumed:

$$p(\mathbf{T}, \mathbf{M}|\boldsymbol{\Theta}, \boldsymbol{\Psi}) = p(\mathbf{T}|\boldsymbol{\Theta})p(\mathbf{M}|\mathbf{T}, \boldsymbol{\Psi})$$

the distribution of the present data is obtained by marginalisation:

$$p(\mathbf{T}_\mathcal{P}, \mathbf{M}|\boldsymbol{\Theta}, \boldsymbol{\Psi}) = \int p(\mathbf{T}, \mathbf{M}|\boldsymbol{\Theta}, \boldsymbol{\Psi})\, d\mathbf{T}_\mathcal{M} = \int p(\mathbf{T}_\mathcal{P}, \mathbf{T}_\mathcal{M}|\boldsymbol{\Theta})p(\mathbf{M}|\mathbf{T}_\mathcal{P}, \mathbf{T}_\mathcal{M}, \boldsymbol{\Psi})\, d\mathbf{T}_\mathcal{M}$$

and we are interested in inferences about $\boldsymbol{\Theta}$ (parameter estimates, confidence intervals and tests) from $p(\boldsymbol{\Theta}, \boldsymbol{\Psi}|\mathbf{T}_\mathcal{P}, \mathbf{M})$. The mechanisms of missing data are classified as follows:

- Missing data are **missing completely at random (MCAR)** if $p(\mathbf{M}|\mathbf{T}_\mathcal{P}, \mathbf{T}_\mathcal{M}, \boldsymbol{\Psi}) = p(\mathbf{M}|\boldsymbol{\Psi})$ for all $\mathbf{T}$: the pattern of missing data does not depend on the actual data. Examples: a study ran out of funds before some subjects could come in for followup visits; a dropped test tube.
- Missing data are **missing at random (MAR)** if $p(\mathbf{M}|\mathbf{T}_\mathcal{P}, \mathbf{T}_\mathcal{M}, \boldsymbol{\Psi}) = p(\mathbf{M}|\mathbf{T}_\mathcal{P}, \boldsymbol{\Psi})$ for all $\mathbf{T}_\mathcal{M}$: the probability that a variable is missing does not depend on the value of that variable when it is missing. Example: females are less likely to give their personal income than males (but giving it or not is independent of their actual income) and the sex of the subject is known. Therefore $p(\mathbf{T}_\mathcal{P}, \mathbf{M}|\boldsymbol{\Theta}, \boldsymbol{\Psi}) = p(\mathbf{T}_\mathcal{P}|\boldsymbol{\Theta})p(\mathbf{M}|\mathbf{T}_\mathcal{P}, \boldsymbol{\Psi})$ and if $\boldsymbol{\Theta}$ and $\boldsymbol{\Psi}$ do not contain common parameters, likelihood-based inferences for $\boldsymbol{\Theta}$ from $p(\boldsymbol{\Theta}, \boldsymbol{\Psi}|\mathbf{T}_\mathcal{P}, \mathbf{M})$ will be the same as likelihood-based inferences for $\boldsymbol{\Theta}$ from $p(\boldsymbol{\Theta}|\mathbf{T}_\mathcal{P})$, i.e., the missing data mechanism is **ignorable**. Most statistical analyses assume MAR.
- Missing data are **not ignorable** or **informative missing** if the probability that a variable is missing depends on the values of the missing variables. Example: subjects with lower income are less likely to provide their personal income in a survey.

Comparison with our approach In section 7.2.2 we ignored any dependence between the probability that a variable be missing and the values that it or other variables may take. If information about such dependence was available, we could use it to further constrain the predictive distribution resulting in fewer candidate reconstructions, either pointwise (given $\mathbf{t}_{n,\mathcal{P}_n}$, constrain $\mathbf{t}_{n,\mathcal{M}_n}$) or globally (given $\{\mathbf{t}_{n,\mathcal{P}_n}\}_{n=1}^N$, constrain $\mathbf{t}_{n,\mathcal{M}_n}$ for each n). This would only be useful for varying missing data patterns, since we do not gain any information if it is always the same variables that are missing.

[37]In this section we use the notation $\mathbf{T} = \{\mathbf{t}_n\}_{n=1}^N$, $\mathbf{T}_\mathcal{M} = \{\mathbf{t}_{n,\mathcal{M}_n}\}_{n=1}^N$ and $\mathbf{T}_\mathcal{P} = \{\mathbf{t}_{n,\mathcal{P}_n}\}_{n=1}^N$ for short. The word "case" refers to a particular point $\mathbf{t}_n$.

7.11.1.2 Statistical methods for missing data

We briefly review the statistical methods for missing data following the taxonomy given by Little (1992) for the problem of regression with missing data, i.e., inferences about a model $p(y|\mathbf{x})$ where the training data for $\mathbf{x}$ have missing values:

Complete-case analysis Cases with any missing data are discarded. While trivial to implement, this has two serious disadvantages: inefficiency, due to the possibly very large reduction in sample size, and therefore larger standard errors, wider confidence intervals and lower power of tests; and bias, if the data are not MCAR. For example, consider $\mathbf{x}$ = (age, sex, blood pressure) and y = (death) with blood pressure sometimes missing, typically when the subject is about to die.

Available-case analysis The largest sets of available cases are used for estimating individual parameters, e.g. to compute the covariance σ^2_{de} between variables x_d and x_e, all cases are used where both x_d and x_e are present. However, the estimated covariance matrix may not be positive definite, especially when the data are highly correlated.

Least squares on imputed data The missing data are filled in or *imputed*[38]. Then, regression of y on $\mathbf{x}$ is computed by least squares (possibly downweighting incomplete data). Imputation methods include:

- *Unconditional mean imputation*: missing data are imputed by their unconditional sample means. The inferences are seriously distorted by bias, so it cannot be generally recommended.
- *Conditional mean imputation based on* $\mathbf{x}$*'s*: information in the present $\mathbf{x}$'s in a case is used to impute the missing $\mathbf{x}$'s, e.g. by linear regression (with parameters estimated from the complete cases).
- *Conditional mean imputation based on* $\mathbf{x}$*'s and* y: using y to fill in missing $\mathbf{x}$'s results in biased estimates.
- *Hot-deck imputation*: missing values are imputed with randomly selected values present in a pool of similar complete cases. This results in less tendency toward the mean because of the variation introduced by the pool. However, depending on the data set, it may be difficult to obtain pools that are large enough to provide with reasonable variance.

Maximum likelihood ML estimates for a model (typically normal) of the joint distribution of y and $\mathbf{x}$. Under model assumptions, this method remains valid when the data are MAR, and simulations have shown is superiority to least-squares methods and complete-case or available-case analyses. Although for certain patterns of missing data closed-form solutions exist, in general iterative methods are necessary, such as scoring, Newton's algorithm or an EM algorithm. The appeal of EM is that, for many problems, the M step is a complete-data problem with a direct solution. Ghahramani and Jordan (1994) and McLachlan and Krishnan (1997) give examples of the use of EM to train a model with missing training data.

Bayesian methods For small samples, maximum likelihood is limited. Bayesian methods approach this by adding a prior to the likelihood and basing inference on the posterior distribution. The complexity of the likelihood function does not allow explicit expressions for marginal posterior distributions of parameters, which have to be approximated by numerical integration or simulation, such as data augmentation, the Gibbs sampler or importance sampling (Schafer, 1997).

Multiple imputation In single-imputation methods the confidence intervals for the complete data are too small because errors in the imputations are not taken into account. In the method of multiple imputation (Rubin, 1987), instead of imputing a single mean for each missing value, $M > 1$ values are drawn from the predictive distribution and then complete-data analyses repeated M times, once with each imputation substituted. That is, inferences from $p(\mathbf{\Theta}|\mathbf{T}_\mathcal{P})$ are done by approximating

$$p(\mathbf{\Theta}|\mathbf{T}_\mathcal{P}) = \int p(\mathbf{\Theta}|\mathbf{T}_\mathcal{P}, \mathbf{T}_\mathcal{M}) p(\mathbf{T}_\mathcal{M}|\mathbf{T}_\mathcal{P})\, d\mathbf{T}_\mathcal{M} \approx \frac{1}{M} \sum_{m=1}^{M} p(\mathbf{\Theta}|\mathbf{T}_\mathcal{P}, \mathbf{T}_{m,\mathcal{M}}) \tag{7.6}$$

from M samples of the predictive distribution

$$p(\mathbf{T}_\mathcal{M}|\mathbf{T}_\mathcal{P}) = \int p(\mathbf{T}_\mathcal{M}|\mathbf{T}_\mathcal{P}, \mathbf{\Theta}) p(\mathbf{\Theta}|\mathbf{T}_\mathcal{P})\, d\mathbf{\Theta}$$

[38] The terms *to impute*, *imputation*, etc. have been traditionally used in statistical analysis with missing data to mean *to fill in* some missing value not to reconstruct it but as a means for inference. They have also been recently adopted in the literature of robust speech recognition (e.g. Cooke et al., 2001).

which, being in general intractable, requires methods such as Markov chain Monte Carlo (Schafer, 1997). Here $\mathbf{T}_{\mathcal{P}}$ ($\mathbf{T}_{\mathcal{M}}$) contains the present (missing) variables of both $\mathbf{x}$'s and y's, since when draws are imputed, conditioning on the $\mathbf{x}$'s alone gives biased estimates while conditioning on the $\mathbf{x}$'s and y does not (the opposite as when means are imputed).

If $\hat{\theta}_m$ and $\hat{v}_m$ are the estimated values for a parameter θ and its variance by the mth analysis, then the final estimate is the average $\hat{\theta} = \frac{1}{M}\sum_{m=1}^{M} \hat{\theta}_m$, with estimated variance $\hat{v}^2 = S_{\mathrm{W}}^2 + \left(1 + \frac{1}{M}\right) S_{\mathrm{B}}^2$, where $S_{\mathrm{W}}^2 \stackrel{\text{def}}{=} \frac{1}{M}\sum_{m=1}^{M} \hat{v}_m$ is the average variance within imputed data sets and $S_{\mathrm{B}}^2 \stackrel{\text{def}}{=} \frac{1}{M-1}\sum_{m=1}^{M} (\hat{\theta}_m - \hat{\theta})^2$ is the between-imputation variance, which reflects uncertainty in the imputation process. Large-sample inference for θ is based on treating $\frac{\hat{\theta}-\theta}{\hat{v}}$ as a Student-t with $\nu = (M-1)\left(1 + \frac{M}{M+1}\frac{S_{\mathrm{W}}^2}{S_{\mathrm{B}}^2}\right)$ degrees of freedom.

If imputations are predictions based on an explicit model, multiple imputation is closely related to ML inference. For example, as the sample size and M increase, $\hat{\theta}$ converges to the ML estimate of θ and $\hat{v}$ converges to the variance of the ML estimate based on the information matrix.

Comparison with our approach There are parallels between imputation methods and our reconstruction algorithm: single imputation with reconstruction by the mean (see our criticism in section 7.3.4) and multiple imputation with multiple pointwise reconstruction by a sample from the conditional distribution (section 7.3.7). However, in multiple imputation each imputation is done on the whole data set, not on each point separately; the latter would imply a number of imputations exponential (on the sample size). We avoided such an exponential complexity by minimising a constraint by dynamic programming. No constraint-based approaches seem to have been used in the statistical literature of missing data—but, again, its aim is model inference rather than data set reconstruction.

Also, the basic approach of the best statistical analysis methods for missing data consists of averaging over the missing data. The averaging is done deterministically by EM and stochastically (by simulation, eq. (7.6)) by multiple imputation; other methods use different types of averaging. This results in averaging branches of a multivalued mapping and contrasts with our method, which is based on mode finding and thus on branch identification.

7.11.2 Universal function approximators

In section 7.3.6 we showed that universal mapping approximators (UMAs), e.g. MLPs, cannot deal with multivalued mappings (because they are by definition functions in their mathematical sense) and are, roughly speaking, equivalent to the conditional mean mapping. In this section we critically review some extensions of mapping approximators that have been proposed for mapping inversion or sequence reconstruction. We consider the problem of approximating a multivalued mapping $\mathbf{y} \rightarrow \mathbf{x}$ and will assume that it is the inverse of a univalued forward mapping $\mathbf{x} \xrightarrow{\mathbf{g}} \mathbf{y}$. None of the methods described here can deal with varying patterns of missing data.

7.11.2.1 Ensembles

Rather than solving a mapping approximation problem by using a single mapping approximator, one can use a finite collection of them, called an **ensemble**[39], whose outputs are combined in some way (Bishop, 1995, section 9.6; Dietterich, 1997). The usual combination strategy is to take the (weighted) average, which is known to improve over the ensemble members in approximation error and generalisation to unseen data if the members are independent from each other and disagree with each other. Ways of obtaining independent, disagreeing predictors include:

- Using different architectures for each predictor.
- Using different training algorithms or different regularisation.
- Training each predictor with different data, e.g. by subsampling methods. A simple and good one is bagging (Breiman, 1996): given a training database of N examples, construct a training set for one predictor by sampling N examples randomly with replacement, and repeat for each predictor.

[39]The terms *ensemble*, *assembly* and *committee* are used interchangeably in the pattern recognition literature. The ensemble approach has been applied to problems of classification too (typically with a voting strategy), but we concentrate on mapping approximation (regression), since we are dealing with reconstruction of continuous data.

- Training to a different local minimum, e.g. by using different initialisations. Backpropagation, for example, is sensitive to initial conditions (Kolen and Pollack, 1990).

The improvement of the ensemble is intuitively due to the cancellation of the individual errors and goes back to the L_2-optimality of the mean for approximation of univalued mappings (section 7.3.3). But for multivalued mappings we have shown that this is wrong (section 7.3.4). For ensembles to succeed with multivalued mappings we need to represent every branch of the mapping with a different ensemble member: this achieves multiple pointwise reconstruction. The ensemble members do not necessarily overlap and a selection, rather than averaging, strategy can then be applied to attain a single reconstruction; constraint minimisation (section 7.6) is an example. We describe here several methods based on the following tasks:

1. *Branch determination* or *resolution*: this is the key part and requires to partition the space of the $\mathbf{x}$-variables into subsets over which the forward mapping is invertible. One can try to do this analytically or by clustering a training set (unsupervised learning). Obviously, random partitions are inappropriate.

2. *Branch inversion*: the forward mapping restricted to a branch is by definition one-to-one, so a separate UMA (such as a polynomial, RBF network or MLP) can now be fit to each branch to define a local inverse (supervised learning). The collection of all UMAs, restricted to their respective subsets of $\mathbf{y}$-space, defines the ensemble and the global inverse. We could learn local forward mappings in the same way, but this is not necessary since a single UMA will correctly learn the global forward mapping, being unimodal (besides, the forward mapping may be known, as for a robot arm).

First we describe two methods based on branch determination by clustering, each proposed to solve a well-known mapping inversion problem:

- Rahim et al. (1993) consider the acoustic-to-articulatory mapping problem, described in section 10.1, where the mapping from articulator configurations $\mathbf{x}$ to acoustic features $\mathbf{y}$ is many-to-one. The algorithm works as follows: first the training data is clustered using a modified k-means algorithm into a selected number $N_{\mathbf{y}}$ of acoustic clusters using an acoustic distance and each cluster is partitioned into another selected number $N_{\mathbf{x}}$ of articulatory subclusters using the articulatory distance. Additional heuristics are applied: clusters are forced to overlap by sharing border vectors to ensure a smooth transition from cluster to cluster; very small clusters are absorbed into neighbouring ones. At this point, it is assumed that within each of the $N_{\mathbf{x}}N_{\mathbf{y}}$ clusters, the $\mathbf{x}$-vectors are close to each other and the $\mathbf{y}$-vectors are close to each other; and that the acoustic space is split into regions such that each region maps one-to-one to a corresponding region in the articulatory domain, each region being associated with a cluster. Now, a different MLP is trained in each cluster; the procedure for this is rather complex and based on heuristics, because one does not know which MLP to adjust for a given speech frame. The actual system uses $N_{\mathbf{y}} = 32$, $N_{\mathbf{x}} = 4$ and an MLP with 26 hidden units.

- DeMers and Kreutz-Delgado (1992, 1998) consider the problem of the inverse kinematics of a robot arm, described in section 9.3.1. The mapping[40] from joint angles $\mathbf{x}$ to location of the end effector in Cartesian coordinates $\mathbf{y}$ is many-to-one. Assume we have a training set $\{(\mathbf{x}_n, \mathbf{y}_n)\}_{n=1}^N$, obtained by finely discretising the $\mathbf{x}$-space and computing $\mathbf{y} = \mathbf{g}(\mathbf{x})$. The algorithm works as follows: select a point $\mathbf{y}_0$ in $\mathbf{y}$-space, define an ϵ-ball around it and search through the training set for all the values of $\mathbf{x}$ whose image lies inside that ball. These $\mathbf{x}$-values will consist of several distinct neighbourhoods in the $\mathbf{x}$-space. If the sampling of the training set is adequate, there should be one such neighbourhood for each of the inverse mapping branches. Thus, each point can be labelled with the corresponding branch. If $\mathbf{g}$ is continuous, perturbing slightly $\mathbf{y}_0$ will perturb the neighbourhoods in $\mathbf{x}$-space and the new data can be labelled in a way consistent with the previous labels. Sweeping the $\mathbf{y}$-space in this way could ideally label, or cluster, a significant proportion of the training set, resulting in tuples $(\mathbf{x}_n, \mathbf{y}_n, b(\mathbf{x}_n))$ where $b(\mathbf{x}_n) \in \{1, \dots, B\}$ indicates which of the B inverse branches the point $\mathbf{x}_n$ (and its unique image $\mathbf{y}_n = \mathbf{g}(\mathbf{x}_n)$) is in. Using these labelled data, one can construct a classifier with supervised learning to compute $b(\mathbf{x})$ for any $\mathbf{x}$ and use it to partition the training set into B subsets, one for each of the inverse branches, and fit a UMA to each separately.

Instead of by clustering, determination of the branches can be done by analysis for some simple robot arms. DeMers and Kreutz-Delgado (1996) consider a synthetic, planar, three-link robot arm problem. They use a combination of self-organising maps to represent the redundant manifolds associated with each solution branch

[40] For uniformity of notation thoughout this section, we use $\mathbf{x}$ and $\mathbf{y}$ rather than $\boldsymbol{\theta}$ and $\mathbf{x}$, respectively (the latter are the conventional symbols in the inverse kinematics problem, section 9.3.1).

of a one-to-many mapping. Each manifold is parameterised by a different self-organising map of dimension equal to the number of degrees of freedom and topology equal to that of the manifold (a circle in their case). The solution branches are determined analytically, so it is necessary to know the topology of the global manifold (i.e., how many branches and of what dimension). This knowledge is also necessary to select some ad-hoc parameters of the learning algorithm. For systems with more than one redundant degree of freedom, which have a complex topological structure (a fiber bundle), this is very difficult. Thus, analytic methods may succeed in some simple problems, but they are inapplicable in general.

The underlying strategy of these methods is divide-and-conquer, with branch determination as "divide" and branch inversion as "conquer." The divide-and-conquer strategy was also the basis of the mixture-of-experts architecture (Jacobs et al., 1991; Jordan and Jacobs, 1994), which is a set of function approximators (*expert networks*) combined by a classifier (*gating network*). The gating net, a multinomial logit model, softly splits the **x**-space into regions where particular experts specialise but allowing data to be processed by multiple experts. The output of each expert is weighted by the gating network's estimate of the probability that that expert is the appropriate one to use (Jordan and Jacobs, 1994), or a particular expert may be chosen according to the gating network's estimates (Jacobs et al., 1991), e.g. the one with the largest estimate. The expert nets are MLPs (Jacobs et al., 1991) or linear mappings (Jordan and Jacobs, 1994). In the latter case, an EM algorithm trains all the parameters of the mixture simultaneously and in both cases the mixture of experts is a UMA. However, it is still restricted to learning univalued mappings (in fact, Jordan and Jacobs, 1994 applied it to the forward dynamics, not the inverse dynamics, of a robot arm). The mixtures of experts can be also extended to do conditional density approximation (section 7.11.3) in a similar way to mixture density networks (Bishop, 1995, pp. 369–371).

A divide-and-conquer architecture, similar to the mixture of experts but contemplating inverse mappings, has been proposed by Wolpert and Kawato (1998) for motor learning. Rather than having a single controller which uses all the contextual information to produce appropriate control signals (which would demand enormous complexity to allow for all possible scenarios), they propose a modular approach with several controllers (inverse models), each suitable for one or a few contexts. This needs a context identification process that will select the appropriate controller given a context; it also needs a learning algorithm so that the set of controllers can cover all the relevant contexts. It remains to be seen whether this generic architecture can be applicable to multivalued mappings.

Critique These methods can in an ideal case represent all the branches of a one-to-many mapping. While the learning stage (clustering and fitting the branches) is computationally costly, they have the advantage over our method of being fast at run-time: inversion requires evaluating the local UMAs rather than mode finding. But they have the disadvantage that getting the right clustering is very difficult, particularly in high dimensions:

- Creating the training set and clustering it requires heuristic parameters, such as the sampling period of the **x**-variables or the size of the ϵ-balls for the method of DeMers and Kreutz-Delgado (1998) or the numbers of clusters $N_{\mathbf{x}}$ and $N_{\mathbf{y}}$ for the method of Rahim et al. (1993). Without a priori knowledge of the number of branches, it is difficult to detect when two neighbourhoods or clusters are really different. These parameters depend on the topologic and the geometric structure of the mapping manifold (e.g. its curvature) which is unknown in general. The clustering is probably quite sensitive to these parameters and a wrong clustering can seriously deteriorate the global inverse obtained.

- The method of DeMers and Kreutz-Delgado (1998) is badly affected by the curse of the dimensionality, since each dimension of the **x**-variables must be thoroughly and finely sampled to ensure that enough points fall within each ϵ-ball.

In short, with clustering there is no guarantee that the local mappings are one-to-one inside every region and determining the regions is computationally costly in high dimensions. And the fact that high-dimensional forward mappings with multivalued inverses have a complex topological structure makes difficult to analytically determine branches where the mapping is one-to-one.

The power of density estimation is that it implicitly represents all the branches, i.e., implicitly determines the topology of the manifold—notwithstanding the difficulty of density estimation in high dimensions and the smoothness problem of section 7.9.1. In our method, branch determination is achieved at reconstruction time by mode-finding in the corresponding conditional distribution[41]; this is so to preserve the flexibility to

[41] The dynamic programming search acts as a voting scheme for an ensemble where a vote happens at each point n in the sequence and the ensemble members are the modes (in varying number at every n).

cope with varying missing data patterns. However, for mapping inversion problems, it is perfectly feasible to determine the branches at training time just as in the methods above by finely sampling the $\mathbf{y}$-variables, using mode-finding in the conditional distribution $\mathbf{x}|\mathbf{y}$ and clustering modes in the same branch (for varying missing data patterns this is not possible because of the combinatorial explosion of mappings).

Density estimation, especially with latent variable models, also naturally solves the problem of redundant degrees of freedom considered by DeMers and Kreutz-Delgado (1996), since only the low-dimensional manifold of the observed space will receive nonnegligible probability mass.

7.11.2.2 Irreversible branch selection at training time

Direct application of a UMA to a multivalued inverse mapping $\mathbf{g}^{-1}$ results in a univalued mapping $\mathbf{h}$ equivalent to the conditional mean that may not be a solution for nonconvex problems (section 7.3.4), i.e., $\mathbf{g}(\mathbf{h}(\mathbf{y})) \neq \mathbf{y}$. Several methods convert the multivalued mapping into a univalued one that is a valid inverse, i.e., that satisfies $\mathbf{g}(\mathbf{h}(\mathbf{y})) = \mathbf{y}\ \forall \mathbf{y}$. For example:

- Jordan and Rumelhart (1992) propose a distal learning procedure applicable to mapping inversion, in particular to the problem of inverting the many-to-one forward kinematics of a robot arm[42] $\mathbf{x} \xrightarrow{\mathbf{g}} \mathbf{y}$. They first train a neural network to model the forward kinematics $\mathbf{g}$; then they prepend to it another network and retrain the resulting, cascaded network to learn the identity, $\mathbf{y} \rightarrow \mathbf{y}$, but keeping unchanged the weights of the forward model. This results in the prepended portion of the network learning one of the possible inverses; in general, it is not possible to know which one, although an inverse satisfying certain desired properties may be found by including appropriate constraint terms in the error function.
- Rohwer and van der Rest (1996) introduce a cost function with a description length interpretation whose minimum is approximated by the densest mode of a distribution. A neural network trained with this cost function can learn one branch of a multivariate mapping.

Therefore, these methods regularise the multivalued inverse mapping by adding some kind of constraints at training time so that the mapping becomes univalued: a single, particular branch is selected and the other inverses can never be recovered. The methods are relatively straightforward and learn well a particular inverse $\mathbf{y} \xrightarrow{\mathbf{h}} \mathbf{x}$, so that trajectories that are contained in that branch exclusively will be accurately reconstructed. But they have important disadvantages:

- The extra inverse branches may be necessary if new constraints must be satisfied at run-time. In other words, trajectories that are contained in other branches will be incorrectly reconstructed.
- The learned inverse mapping must contain discontinuous jumps between branches, similar to those of the global mode method (e.g. see fig. 9.8).

7.11.2.3 Recurrent nets

Feedforward nets, such as the MLP, are memoryless function approximators in that the predicted value depends only on the current input of a sequence. To represent information from the past, recurrent nets (Hertz et al., 1991; Robinson, 1994; Tsoi, 1998) extend this architecture via feedback loops, e.g. to additional hidden units called context units or to a tapped delay line (time-delay neural networks). For observed data $\mathbf{t}^{(1)}, \mathbf{t}^{(2)}, \dots, \mathbf{t}^{(N)}$ they then estimate $\mathbf{t}_n|\mathbf{t}_1, \dots, \mathbf{t}_{n-1}$, which makes them attractive for time series modelling. There exist different architectures and several training algorithms, e.g. gradient-based (Pearlmutter, 1995). Bengio and Gingras (1996) have used recurrent nets with feedback into the input units for handling missing input variables. Schuster (1999) has extended recurrent nets to estimate the outputs given future as well as past inputs.

Recurrent nets have a higher representational power than feedforward nets and they may conceivably be able to learn a constraint (given by the neighbouring sequence points) and an inverse mapping from data so that the right mapping branch is tracked at reconstruction time. Thus, it would be worth to investigate their performance on a mapping inversion problem. However, they have two important practical disadvantages:

- Compared with feedforward nets, they are more difficult to train (it requires large training sets, takes longer and there may be convergence problems) and do not generalise as reliably.

[42] As earlier, we use $\mathbf{x}$ and $\mathbf{y}$ rather than the conventional $\boldsymbol{\theta}$ and $\mathbf{x}$, respectively.

- It may be difficult to find the right architecture for a given problem, particularly the number of context units or the time lag. This also applies to other time series models, for example to an *autoregressive model* of lag p, which takes $(\mathbf{t}_n|\mathbf{t}_1,\dots,\mathbf{t}_{n-1},\mathbf{\Theta}) \sim \mathcal{N}(\boldsymbol{\mu}+\beta_1\mathbf{t}_{n-1}+\cdots+\beta_p\mathbf{t}_{n-p},\mathbf{\Sigma})$, where the parameters $\mathbf{\Theta}$ are the constant $\boldsymbol{\mu}$, the regression coefficients $\beta_1,\dots,\beta_p$ and the error variance $\mathbf{\Sigma}$. Least-squares estimates of $\mathbf{\Theta}$ are found by regressing $\mathbf{t}_n$ on $(\mathbf{t}_{n-1},\dots,\mathbf{t}_{n-p})$ for observations $n = p, p+1,\dots,N$, but the appropriate value for p is unknown.

7.11.3 Conditional vs density modelling

To approximate a multivalued mapping $\mathbf{x} \rightarrow \mathbf{y}$, only the conditional distribution $p(\mathbf{y}|\mathbf{x})$ is necessary, not the joint distribution $p(\mathbf{x},\mathbf{y})$. Several approaches for estimating conditional distributions from paired data $\{(\mathbf{x}_n,\mathbf{y}_n)\}_{n=1}^N$ have been proposed (see Husmeier, 1999 for further references):

- One approach is to estimate a density model for $\mathbf{y}$ with parameters $\mathbf{\Theta}$ (means, variances, etc.) depending on the inputs $\mathbf{x}$ through a mapping approximator, e.g. an MLP with weights $\mathbf{w}$, inputs $\mathbf{x}$ and outputs $\mathbf{\Theta}$ that computes $\mathbf{\Theta}(\mathbf{x};\mathbf{w})$. If the density model for $\mathbf{y}$ and the mapping approximator are flexible enough then any reasonable conditional distribution can be represented. The only parameters to be learned are the MLP weights $\mathbf{w}$, which can be done by minimising the log-likelihood with gradient descent in a backpropagation-style way. Care is needed to ensure that the density parameters $\mathbf{\Theta}$ computed by the MLP satisfy certain constraints, which can be done by using a suitable transformation. For example, variances must be positive, which can be done by applying the exponential function to the relevant MLP outputs (this also helps to avoid zero variances); mixture proportions must be positive and add to one, which can be done by applying the softmax function to the relevant MLP outputs; and covariance matrices must be positive definite, which can be done by representing the covariance matrix as its Cholesky factorisation $\mathbf{\Sigma} = \mathbf{L}\mathbf{L}^T$ where $\mathbf{L}$ is lower triangular and has positive diagonal elements. Location parameters can usually take any real value and so their corresponding MLP outputs need no special transformation. Two examples of these models using MLPs are the *mixture density networks* of Bishop (1994) (see also Bishop, 1995, pp. 211–222), who uses a Gaussian mixture of spherical components as density model for $\mathbf{y}$, and the model of Williams (1996), who uses a single full-covariance Gaussian instead.
- Another approach is to implement $p(\mathbf{y}|\mathbf{x})$ directly with a function approximator. Husmeier (1999) uses a neural network architecture to approximate the conditional cdf (for univariate y) which can then be differentiated to obtain the conditional pdf. He shows that at least two hidden layers of sigmoidal nonlinearities are necessary for that architecture to be a universal approximator of conditional distributions.

Bishop (1994) uses the centroid of one of the mixture components as an approximation to the global mode of the conditional distribution $\mathbf{y}|\mathbf{x}$ (estimated by the mixture density network): either the component with highest mixture proportion $\pi_m(\mathbf{x})$ or the component with highest density value $\frac{\pi_m(\mathbf{x})}{\sigma_m(\mathbf{x})}$. This results then in the same branch of the mapping being selected for a given value of $\mathbf{x}$ (just as in the irreversible branch selection methods of section 7.11.2.2), with the rest of the information contained in the conditional distribution being ignored.

The conditional distribution obtained for a value of $\mathbf{x}$ could be used to provide multiple pointwise reconstruction by finding its modes, as we propose in this chapter. And estimating only the conditional distribution (D_2 variables, $y_1,\dots,y_{D_2}$) is more efficient than estimating the joint density model ($D_1 + D_2$ variables, $x_1,\dots,x_{D_1},y_1,\dots,y_{D_2}$), especially in view of the curse of the dimensionality. But the disadvantage is that, like function approximators, it treats the variables in an asymmetric way: $\mathbf{y}$ missing and $\mathbf{x}$ present. To reconstruct missing $\mathbf{x}$ from present $\mathbf{y}$ (for example) one would need the conditional distribution $p(\mathbf{x}|\mathbf{y}) \propto p(\mathbf{y}|\mathbf{x})p(\mathbf{x})$ which requires estimating the density of the $\mathbf{x}$ variables or equivalently the joint density $p(\mathbf{x},\mathbf{y})$.

7.11.4 Vector quantisation, codebooks and dynamic programming

Consider again a known forward mapping $\mathbf{g}: \mathcal{X} \rightarrow \mathcal{Y}$ to be inverted. In vector quantisation, one constructs a set of pairs $\{(\mathbf{x}_m,\mathbf{y}_m)\}_{m=1}^M \subset \mathcal{X}\times\mathcal{Y}$ called *codebook* where $\mathbf{g}(\mathbf{x}_m) = \mathbf{y}_m$ and the codebook thoroughly and finely spans the low-dimensional manifold of the mapping $\mathbf{g}$. Then, given a point $\mathbf{y} \in \mathcal{Y}$, we can look up or scan the codebook for candidate inverses that approximately map onto $\mathbf{y}$. There are different options for doing this:

1. Return all points $\mathbf{x}_m$ such that $d(\mathbf{g}(\mathbf{x}_m) - \mathbf{y}) < \epsilon$ where d is a distance in the space $\mathcal{Y}$ and ϵ is large enough to return at least one $\mathbf{x}$ but not too large that many wrong $\mathbf{x}$s are returned.

2. Return the M' best inverses, i.e., order the codebook in increasing distance $d(\mathbf{g}(\mathbf{x}_m) - \mathbf{y})$ and pick the first M', with $M' \ll N$.

3. Return the whole codebook, i.e., $M' = N$.

To invert a sequence $\{\mathbf{y}^{(n)}\}_{n=1}^N$, the pointwise candidate reconstructions provided by the codebook can be used with dynamic programming to minimise a global constraint or cost function, such as continuity. Since the forward mapping is known, a constraint of the form of eq. (10.2) (see sections 7.5.3 and 7.6.1) can be used too.

Dynamic programming search of codebooks has been used for articulatory inversion (section 10.1.3) and in fact it is the leading method for that problem. There, option 3 is used at each point n, i.e., the whole codebook is used as candidate reconstructions; option 2 is used only when the codebook size N is very large, to limit computation time; option 1 seems not to have been used. It is surprising that no effort has been devoted to refine the set of candidate codebook vectors in view of the problems that spurious candidates can bring about (see section 7.9.1 about the smoothness of density models). The likely reason is that using the forward mapping in the constraint partly filters out such wrong candidate reconstructions.

Codebooks have the following disadvantages:

- Huge size: finely sampling in several dimensions implies very high codebook storage and search time (the curse of the dimensionality).

- Constructing the codebook is difficult for several reasons (see section 10.1.3 for details). Among others, simple clustering algorithms like k-means cannot be used because the data manifold usually is not convex and so interpolated values may be illegal; and it is difficult to obtain a good sampling because the forward mapping $\mathbf{g}$ can stretch or compress the distance between neighbouring samples in the space $\mathcal{X}$.

- Even assuming a good codebook, the look-up returns fewer or, more likely, more inverse values than really exist (e.g. several per branch), which should result in the same problems as the spurious modes or the heuristic sampling of the conditional distribution (section 7.3.7).

- The reconstructed values are quantised, that is, only a finite number of different values is available to fill in the missing variables, even though their range is continuous. For reconstruction methods based on the mean, modes (as ours) or random samples from a Gaussian mixture, a continuous range is preserved for every variable. The reason is that different values of the present variables result in different forms for the conditional distribution.

Dynamic programming search of codebooks is a particular case of our method. In effect, the codebook is a zero-variance limit version of a mixture density model. The latter has the advantage of (1) being more parsimonious (the size of the parameters is much smaller than the size of the codebook, although this depends on how fine the latter is and how accurate and smooth the former is) and (2) providing with the correct inverse values (assuming a good density model), not requiring a neighbourhood parameter ϵ or M'. In terms of computational complexity at reconstruction time, which method is faster depends on the choice of M' and the implementation of the mode finding algorithm.

Finally, the codebook approach should be applicable to general missing data patterns, although noone seems to have done this.

7.11.5 Dynamical, sequential and time series modelling

Many sequence models are based on the Markov assumption, which correlates consecutive points and so provides a local constraint (as well as simplifying the algebra). For example:

- For continuous time: a *Kalman filter* (Harvey, 1991) takes $(\mathbf{t}^{(n)}|\mathbf{M}^{(n)}\mathbf{x}^{(n)}, \boldsymbol{\Theta}) \sim \mathcal{N}(\mathbf{M}^{(n)}\mathbf{x}^{(n)}, \mathbf{B})$, $(\mathbf{x}^{(1)}|\boldsymbol{\Theta}) \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ and $(\mathbf{x}^{(n)}|\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(n-1)}, \boldsymbol{\Theta}) \sim \mathcal{N}(\boldsymbol{\Phi}\mathbf{x}^{(n-1)}, \mathbf{Q})$ where $\{\mathbf{x}^{(n)}\}_{n=1}^N$ is a random unobserved series and the parameters $\boldsymbol{\Theta}$ are $\mathbf{B}$, $\boldsymbol{\mu}$, $\boldsymbol{\Sigma}$, $\boldsymbol{\Phi}$ and $\mathbf{Q}$. If $\boldsymbol{\Theta}$ was known, the optimal estimates for $\{\mathbf{x}^{(n)}\}_{n=1}^N$ would be their means conditioned on $\boldsymbol{\Theta}$ and the observed data $\{\mathbf{t}^{(n)}\}_{n=1}^N$. There are recursive formulas and an EM algorithm to obtain $\boldsymbol{\Theta}$.

  Like function approximators, Kalman filters learn to average paths when the underlying path distribution is multimodal. *Diffusion networks* (Movellan et al., 1999) have been recently proposed to represent multimodal path distributions. A diffusion network is a stochastic extension of continuous-time, continuous-state recurrent neural networks in which the dynamics are described by stochastic differential equations. A diffusion network results in a parametric model for distributions of time-varying signals that, once trained, can be used for tasks such as sequence recognition, sequence generation, stochastic filtering, prediction, smoothing or decoding.

- For discrete time: *hidden Markov models* (HMMs) (Rabiner and Juang, 1993) are the most successful models in speech recognition and recently in biological sequence analysis (Durbin et al., 1998). An HMM is a probabilistic finite state machine with a probabilistic output process attached to each state (or transition): if at time n the machine is at state $q^{(n)}$, it will output a value $\mathbf{t}^{(n)}$ according to an emission probability distribution $p(\mathbf{t}^{(n)}|q^{(n)})$ and move to a state $q^{(n+1)}$ according to a state transition probability distribution $p(q^{(n+1)}|q^{(n)})$. The sequence of states $\{q^{(n)}\}_{n=1}^N$ models the temporal structure of the series and is unobserved; the sequence of state output processes models the locally stationary character of the series and is observed. The parameters are the transition probability matrix $\mathbf{P} = (p_{ij})$ (where p_{ij} is the transition probability from state i to state j) and the parameters of the emission distribution. Given observed data, maximum likelihood estimates of the parameters can be computed exactly with the Baum-Welch algorithm, which is an EM algorithm, or approximately but faster with Viterbi training. Given an observed sequence $\{\mathbf{t}^{(n)}\}_{n=1}^N$, the state sequence maximising $p(\{\mathbf{t}^{(n)}\}_{n=1}^N|\{q^{(n)}\}_{n=1}^N)$, or $p(\{q^{(n)}\}_{n=1}^N|\{\mathbf{t}^{(n)}\}_{n=1}^N)$ for MAP, can be obtained by Viterbi decoding, which is a dynamic programming algorithm.

  However, in standard HMMs the state space has no topology, just as it happens in a Gaussian mixture: the state (or mixture component) index is just a label and is not attached to a Euclidean space. There have been several extensions of HMMs that embed the states in a low-dimensional Euclidean space and constrain the transitions to occur between neighbouring states—in an effort to implement the mechanical constraints associated with the speech production organs, the slow-moving articulators of the vocal tract. For example, the *GTM through time* model of Bishop et al. (1997a) reuses the latent space points of GTM (section 2.6.5) as states of an HMM. The output probability at each state of the HMM is naturally given by $p(\mathbf{t}|\mathbf{x}_k) = p(\mathbf{t}|\mathbf{f}(\mathbf{x}_k))$. The transition probabilities are tied to groups of states. For example, in 2D one can consider 10 groups (and thus 10 parameters) from a given state at position (i, j): the 9 neighbouring states at $(i \pm d_1, j \pm d_2)$ with $d_1, d_2 \in \{0, 1\}$ (9 parameters) plus all the remaining states (1 parameter). This constrained transition matrix $\mathbf{P}$ allows for continuity (neighbouring states) and for transitions between different regimes (remaining states). The free parameters are the matrix $\mathbf{P}$ (tied) and $\mathbf{W}$ and σ from the standard GTM (the same for all states). An EM algorithm for maximum likelihood estimation is given, which updates $\mathbf{W}$ and σ like the standard GTM and $\mathbf{P}$ like the Baum-Welch algorithm of HMMs. As another example, Roweis (2000) constrains HMM transitions to occur between neighbouring knots of a grid and also gives an EM algorithm. In general, these constraints result in a banded, tied structure for the transition matrix $\mathbf{P}$. A more abstract way of imposing such constraints would be to do so on the transition matrix directly, for example using Toeplitz matrices, as has been done for the HMM covariance matrices (see Roberts and Ephraim, 2000 and references therein). An additional advantage of these constrained HMM models is that they reduce the number of free parameters and so require smaller training sets, which is particularly welcome in automatic speech recognition.

  Nix and Hogden (1999) have presented the *maximum-likelihood continuity mapping* (MALCOM) as a variation of HMMs to allow for a continuous path in hidden state space. The model formulation is basically that of HMMs with integrals over state-space paths instead of sums. However, to make it tractable, they quantise the acoustic space into an acoustic codebook $\{\mathbf{y}_m\}_{m=1}^M$ and take the conditional distribution of state given acoustics as $\mathbf{x}|\mathbf{y}_m \sim \mathcal{N}(\boldsymbol{\mu}_m, \sigma^2\mathbf{I})$, thus unsurprisingly converting the hidden state distribution $p(\mathbf{x})$ in a Gaussian mixture. By forcing the distribution of hidden states given the acoustics to be unimodal (Gaussian), the model becomes unable to account for the nonuniqueness of the acoustic-to-articulatory mapping (since the hidden space is assumed to be an articulatory space). The smoothness constraint enters the training algorithm as a low-pass filtering of the state path $\mathbf{x}_1, \dots, \mathbf{x}_N$ for a given observed acoustic sequence $\mathbf{y}_1, \dots, \mathbf{y}_N$. The training algorithm itself is iterative, with each iteration consisting of a gradient maximisation of the log-likelihood over the means $\{\boldsymbol{\mu}_m\}_{m=1}^M$ for fixed path $\{\mathbf{y}_n\}_{n=1}^N$ followed by a gradient maximisation and low-pass filtering over the path (and a reestimation of the variance σ) for fixed means. It remains to be proven whether MALCOM has any advantage over simply constraining the transitions of a standard HMM as mentioned before.

  For articulatory speech modelling, production information can be incorporated in the state space more literally by simply quantising a few articulatory variables and forbidding some transitions. In the *articulatory-feature model* of Erler and Freeman (1996), an articulatory feature space is described by 7 features taken from the theory of articulatory phonology of Browman and Goldstein (1992) (section 10.1.2.1) such as the degree of rounding of the lips, the position of the tongue tip constriction or the presence or absence of voicing. Each articulatory feature is quantised into 2 to 6 values (e.g. the lip rounding into 5 possible values, ranging from 'closed' to 'spread'). This results in 7200 states, which define the state space of an HMM. State transitions are subjected to several types of constraints, including (1) static rules, which discard physically unrealisable articulatory configurations (e.g. 'tongue tip cannot be behind tongue body'); and (2) dynamic rules, which dictate what values a given articulatory variable can take at a particular time (e.g. depending on the intrinsic inertia of its associated articulator, that is, continuity and velocity constraints). The model is trained using words labelled with their ortographic description. This description is mapped to a phonemic description using a lookup dictionary and each phonemic segment is mapped to an articulatory target which, after applying the rules, results in an articulatory state sequence. Each state outputs an acoustic vector according to its own emission distribution. Initialising the model is difficult, and the trained model is sensitive to the particular initialisation chosen. Recognition results using a carefully crafted model with a large-vocabulary, speaker-dependent, isolated word task approach the performance of standard acoustic HMMs.

Other variations of Markovian sequence models are *segment models* (Ostendorf et al., 1996), a complex extension of HMMs where the states generate sequences themselves rather than a single output vector and allow the inclusion of trajectory and correlation constraints; and *Markov processes on curves* (Saul and Rahim, 2000a), which combine the Markov assumption with a continuous trajectory parameterised by arc length to achieve invariance to time warpings (trajectory reparametrisations).

Reinhard and Niranjan (1999) propose a method to capture segmental transition information for diphone classification. Each diphone is sampled as a sequence of acoustic feature vectors. They enforce the sequence order by augmenting each vector with its index (weighted by a scale parameter) in the sequence. The augmented sequence is then reduced to a two-dimensional sequence with PCA. In this subspace, each diphone is represented by a trajectory model (a principal curve, GTM or a simple framewise average) estimated from several sequences. A given test sequence is first smoothed with a cubic spline and then classified as the diphone whose trajectory model is closest to it. The Euclidean distance normalised by the trajectory length is used in an attempt to attain duration invariance.

Comparison with our approach Our approach does not attempt to model the temporal evolution of the system. The joint probability model is estimated statically. The temporal aspect of the data appears indirectly and a posteriori through the application of the continuity constraints to select a trajectory[43]. In this respect, our approach differs from that of dynamical systems or from models based on Markovian assumptions, such as those above.

The fact that the duration or speed of the trajectory plays no role in our algorithm makes it invariant to time warping. That is, the dynamic programming algorithm depends only on the values of the observed variables but not on the experimental conditions and so is independent of the speed at which the trajectory is traversed. It is also independent of direction, since it can be run forwards (from point 1 to N) or backwards with the same result. Therefore, our reconstruction algorithm does not depend on the particular parametrisation of the trajectory, but just on its geometric shape. This is important in the case of speech: it is well known that hidden Markov models are very sensitive to time warpings, i.e., fast or slow speech styles, where the trajectory in speech feature space is the same but is traversed fast or slowly, respectively. Our reconstruction method should then be robust to time warpings.

Formally, a time series prediction is a reconstruction problem where the data set is $\{\mathbf{t}^{(1)}, \mathbf{t}^{(2)}, \dots, \mathbf{t}^{(N)}, \mathbf{t}^{(N+1)}, \dots, \mathbf{t}^{(N+N')}\}$, $\{\mathbf{t}^{(n)}\}_{n=1}^{N}$ are present and $\{\mathbf{t}^{(n)}\}_{n=N+1}^{N+N'}$ are missing. But, even if continuity constraints and stationarity assumptions ($p(\mathbf{t})$ independent of n) hold, the values for the future, missing variables $\{\mathbf{t}^{(n)}\}_{n=N+1}^{N+N'}$ are too unconstrained: many reconstructed trajectories would be equally plausible[44]. Thus, our method is not useful for time series prediction.

Finally, there exists work in the (statistical) signal processing literature concerning the restoration of audio and images corrupted by impulsive noise. A typical application is the removal of click and crackle noise from old gramophone recordings or of dirt and sparkle from pictures and films, due to surface irregularities, scratches, dust particles, etc. (Vaseghi, 2000; Godsill and Rayner, 1998a). In the problem of audio reconstruction, the objective is to recover an uncorrupted audio sequence $\{s^{(n)}\}_{n=1}^{N}$ (where n is the time) from an observed sequence $\{s_{\text{obs}}^{(n)}\}_{n=1}^{N}$. Several methods have been developed for this, including median filtering, least-squares interpolation

[43]The constraints, though, may be seen probabilistically as a normal, Markovian dependence between adjacent frames (section 7.7).

[44]In fact, the trivial application of a continuity or smoothness constraint would lead to $\hat{\mathbf{t}}^{(n)} = \mathbf{t}^{(N)}\ \forall n > N$, i.e., a constant sequence.

(Vaseghi and Rayner, 1990) and Bayesian reconstruction using a likelihood model $p(\{s_{\text{obs}}^{(n)}\}_{n=1}^{N}|\{s^{(n)}\}_{n=1}^{N}, \mathbf{\Theta})$ with parameters $\mathbf{\Theta}$ and a prior on the uncorrupted data (Godsill and Rayner, 1998b). In these methods, redundancy in the temporal domain is exploited via a one-dimensional *autoregressive process* (section 7.11.2.3). The same methods are applicable to image reconstruction, where the signal s depends then on the spatial coordinates x and y as well as the time n, and so a three-dimensional autoregressive process is necessary (Kokaram et al., 1995). Thus, these methods apply to reconstruction of one-dimensional signals that are (semi)continuous or smooth over time and/or space; however, the method we propose in this chapter applies when the signal itself is multidimensional and exploits the redundancy that arises due to its lower intrinsic dimensionality—in addition to the temporal or spatial redundancy.

7.12 Mathematical appendix

In this appendix, selection of variables follows the notation of section 7.2.5. Let $\mathcal{I}, \mathcal{J} \in \{1, \dots, D\}$ be disjoint, nonempty sets of indices with $\mathcal{I} \cup \mathcal{J} \subseteq \{1, \dots, D\}$ not necessarily equal to $\{1, \dots, D\}$.

7.12.1 Marginal and conditional distributions of Gaussian mixtures

Let us write $\mathbf{t} \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{\Sigma})$ meaning that the D-dimensional vector $\mathbf{t}$ has a D-variate normal distribution of mean vector $\boldsymbol{\mu}$ and covariance matrix $\mathbf{\Sigma}$. Up to an index permutation, D-dimensional vectors and $D \times D$ matrices are partitioned in the usual way according to index sets $\mathcal{I}$, $\mathcal{J}$:

$$\boldsymbol{\mu} = \begin{pmatrix} \boldsymbol{\mu}_{\mathcal{I}} \\ \boldsymbol{\mu}_{\mathcal{J}} \\ \dots \end{pmatrix}, \qquad \mathbf{\Sigma} = \begin{pmatrix} \mathbf{\Sigma}_{\mathcal{I}\mathcal{I}} & \mathbf{\Sigma}_{\mathcal{I}\mathcal{J}} & \dots \\ \mathbf{\Sigma}_{\mathcal{I}\mathcal{J}}^{T} & \mathbf{\Sigma}_{\mathcal{J}\mathcal{J}} & \dots \\ \dots & \dots & \dots \end{pmatrix}$$

where ... means the rest of the variables.

Recall the well-known facts that if $\mathbf{t} \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{\Sigma})$ then (section A.3; Mardia et al., 1979):

- $\mathbf{t}_{\mathcal{I}} \sim \mathcal{N}(\boldsymbol{\mu}_{\mathcal{I}}, \mathbf{\Sigma}_{\mathcal{I}\mathcal{I}})$, i.e., any marginal is normal with the marginalised variables removed.
- $\mathbf{t}_{\mathcal{J}}|\mathbf{t}_{\mathcal{I}} \sim \mathcal{N}(\boldsymbol{\mu}_{\mathcal{J}|\mathcal{I}}, \mathbf{\Sigma}_{\mathcal{J}|\mathcal{I}})$ with $\boldsymbol{\mu}_{\mathcal{J}|\mathcal{I}} = \boldsymbol{\mu}_{\mathcal{J}} + \mathbf{\Sigma}_{\mathcal{I}\mathcal{J}}^{T}\mathbf{\Sigma}_{\mathcal{I}\mathcal{I}}^{-1}(\mathbf{t}_{\mathcal{I}} - \boldsymbol{\mu}_{\mathcal{I}})$ and $\mathbf{\Sigma}_{\mathcal{J}|\mathcal{I}} = \mathbf{\Sigma}_{\mathcal{J}\mathcal{J}} - \mathbf{\Sigma}_{\mathcal{I}\mathcal{J}}^{T}\mathbf{\Sigma}_{\mathcal{I}\mathcal{I}}^{-1}\mathbf{\Sigma}_{\mathcal{I}\mathcal{J}}$, i.e., any conditional is normal.

Thus, for a Gaussian mixture $p(\mathbf{t}) = \sum_{m=1}^{M} p(m)p(\mathbf{t}|m)$ with $\mathbf{t}|m \sim \mathcal{N}(\boldsymbol{\mu}_m, \mathbf{\Sigma}_m)$, we obtain immediately the following exact formulae for the marginal and conditional distributions:

- $p(\mathbf{t}_{\mathcal{I}}) = \sum_{m=1}^{M} p(m)p(\mathbf{t}_{\mathcal{I}}|m)$, with $\mathbf{t}_{\mathcal{I}}|m \sim \mathcal{N}(\boldsymbol{\mu}_{m,\mathcal{I}}, \mathbf{\Sigma}_{m,\mathcal{I}\mathcal{I}})$, i.e., any marginal is a mixture of normal distributions with the marginalised variables removed.
- $p(\mathbf{t}_{\mathcal{J}}|\mathbf{t}_{\mathcal{I}}) = \sum_{m=1}^{M} p(m|\mathbf{t}_{\mathcal{I}})p(\mathbf{t}_{\mathcal{J}}|\mathbf{t}_{\mathcal{I}}, m)$, with $p(m|\mathbf{t}_{\mathcal{I}}) = p(\mathbf{t}_{\mathcal{I}}|m)p(m)/p(\mathbf{t}_{\mathcal{I}})$ and $\mathbf{t}_{\mathcal{J}}|\mathbf{t}_{\mathcal{I}}, m \sim \mathcal{N}(\boldsymbol{\mu}_{m,\mathcal{J}|\mathcal{I}}, \mathbf{\Sigma}_{m,\mathcal{J}|\mathcal{I}})$, i.e., any conditional is a mixture of normal distributions.

For diagonal components $\mathbf{\Sigma}_{m,\mathcal{I}\mathcal{J}} = \mathbf{0}$ and $\mathbf{\Sigma}_{m,\mathcal{I}\mathcal{I}}$ is diagonal for all m and any $\mathcal{I}$, $\mathcal{J}$. Therefore, in the conditional distribution we avoid the computationally costly M matrix inversions of $\mathbf{\Sigma}_{m,\mathcal{I}\mathcal{I}}$ and the distribution of each component is obtained by crossing out rows and columns: $\mathbf{t}_{\mathcal{J}}|\mathbf{t}_{\mathcal{I}}, m \sim \mathcal{N}(\boldsymbol{\mu}_{m,\mathcal{J}}, \mathbf{\Sigma}_{m,\mathcal{J}\mathcal{J}})$.

7.12.2 Marginal and conditional distributions of mixtures of factorised distributions

Conditioning and marginalising on mixtures of factorised distributions is straightforward. In section 2.4 we saw that marginalising a multivariate distribution can be hard if the integral cannot be solved analytically. However, consider a D-variate distribution having the form of a mixture of factorised distributions with M components:

$$p(\mathbf{t}) = \sum_{m=1}^{M} p(m)p(\mathbf{t}|m) \quad \mathbf{t} = (t_1, \dots, t_D) \in \mathcal{T} \subset \mathbb{R}^D, \text{ with } p(\mathbf{t}|m) = \prod_{d=1}^{D} p(t_d|m) \quad \forall m = 1, \dots, M.$$

From the joint distribution $p(\mathbf{t})$ we can compute the marginal distribution of $\mathbf{t}_{\mathcal{I}}$ as follows:

$$p(\mathbf{t}_{\mathcal{I}}) = \int p(\mathbf{t})\, d\mathbf{t}_{\{1,\dots,D\}\setminus\mathcal{I}} = \int \sum_{m=1}^{M} p(m)p(\mathbf{t}|m)\, d\mathbf{t}_{\{1,\dots,D\}\setminus\mathcal{I}} = \sum_{m=1}^{M} p(m)p(\mathbf{t}_{\mathcal{I}}|m) \tag{7.7}$$

where $p(\mathbf{t}_\mathcal{I}|m) = \prod_{d\in\mathcal{I}} p(t_d|m)$, since marginalising a factorial distribution amounts to removing the terms associated with the variables on which we marginalise (e.g. if $p(\mathbf{t}|m)$ is diagonal normal, we cross out all rows and columns except those of $\mathcal{I}$ from its covariance matrix and mean vector, as seen in section 7.12.1):

$$\begin{aligned} p(\mathbf{t}_\mathcal{I}|m) &= \int p(\mathbf{t}|m)\, d\mathbf{t}_{\{1,\dots,D\}\setminus\mathcal{I}} = \int \prod_{d=1}^{D} p(t_d|m)\, d\mathbf{t}_{\{1,\dots,D\}\setminus\mathcal{I}} \\ &= \left(\prod_{d\in\mathcal{I}} p(t_d|m)\right)\left(\prod_{d\in\{1,\dots,D\}\setminus\mathcal{I}} \int p(t_d|m)\, dt_d\right) = \prod_{d\in\mathcal{I}} p(t_d|m). \end{aligned} \tag{7.8}$$

Equation (7.7) shows that marginalising a mixture of factorised distributions results in another mixture of factorised distributions with the corresponding variables removed.

Computing conditional distributions is also straightforward. The distribution of $\mathbf{t}_\mathcal{J}$ conditional on $\mathbf{t}_\mathcal{I}$ can be computed as

$$p(\mathbf{t}_\mathcal{J}|\mathbf{t}_\mathcal{I}) = \frac{p(\mathbf{t}_\mathcal{J},\mathbf{t}_\mathcal{I})}{p(\mathbf{t}_\mathcal{I})} = \frac{p(\mathbf{t}_{\mathcal{I}\cup\mathcal{J}})}{p(\mathbf{t}_\mathcal{I})} = \frac{\sum_{m=1}^{M} p(m)p(\mathbf{t}_{\mathcal{I}\cup\mathcal{J}}|m)}{\sum_{m=1}^{M} p(m)p(\mathbf{t}_\mathcal{I}|m)} = \frac{\sum_{m=1}^{M} p(m)p(\mathbf{t}_\mathcal{I}|m)p(\mathbf{t}_\mathcal{J}|m)}{\sum_{m=1}^{M} p(m)p(\mathbf{t}_\mathcal{I}|m)}. \tag{7.9}$$

Thus we have exact formulae to marginalise and condition on mixtures of factorised distributions.

Using Bayes' theorem on the component index m,

$$p(m|\mathbf{t}_\mathcal{I}) = \frac{p(\mathbf{t}_\mathcal{I}|m)p(m)}{\sum_{m=1}^{M} p(m)p(\mathbf{t}_\mathcal{I}|m)}$$

eq. (7.9) can also be put in the form

$$p(\mathbf{t}_\mathcal{J}|\mathbf{t}_\mathcal{I}) = \sum_{m=1}^{M} p(m|\mathbf{t}_\mathcal{I})p(\mathbf{t}_\mathcal{J}|m)$$

which shows that the conditional distribution of a mixture of factorised distributions is another mixture of factorised distributions, where the component weights are the posterior probabilities or *responsibilities* of the component given vector $\mathbf{t}_\mathcal{I}$. The mean of this distribution is immediately obtained as the weighted intracomponent marginal mean:

$$\mathrm{E}_{p(\mathbf{t}_\mathcal{J}|\mathbf{t}_\mathcal{I})}\{\mathbf{t}_\mathcal{J}\} = \sum_{m=1}^{M} p(m|\mathbf{t}_\mathcal{I})\,\mathrm{E}_{p(\mathbf{t}_\mathcal{J}|m)}\{\mathbf{t}_\mathcal{J}\},$$

which is a point in the convex hull of the intracomponent marginal means $\{\mathrm{E}_{p(\mathbf{t}_\mathcal{J}|m)}\{\mathbf{t}_\mathcal{J}\}\}_{m=1}^{M}$, since all the $p(m|\mathbf{t}_\mathcal{I})$ are nonnegative and add to 1.

7.12.3 Marginal and conditional distributions of latent variable models in data space

In section 2.3 we saw that the marginal distribution in data space, $p(\mathbf{t})$, was obtained from the joint distribution $p(\mathbf{x},\mathbf{t}) = p(\mathbf{t}|\mathbf{x})p(\mathbf{x})$ by integrating it over the latent variables. Two situations were possible:

- The integral in latent space was analytically possible. The distribution in data space $p(\mathbf{t})$ has a closed form and has to be integrated again to marginalise and condition between variables in data space. Notice that there is no guarantee that this can be done analytically again. However, for the linear-normal models (factor analysis and PCA), this is indeed possible and the resulting marginal and conditional distributions are also normal, as is well known, with appropriate values of the mean vector and covariance matrix (given in section 7.12.3.1).

- The integral in latent space was approximated by a Monte Carlo or fixed sampling, resulting in a sum over a number of fixed points in latent space (section 2.4). Owing to the axiom of local independence, this distribution in data space is a mixture of factorised distributions and we can apply the results of section 7.12.2. For the GTM case, the marginal and conditional distributions are also mixtures of factorised normal distributions, and the very simple resulting formulae are given in section 7.12.3.2.

7.12.3.1 Linear-normal latent variable models

With the same matrix partitioning notation as in section 7.12.1:

Factor analysis The distribution in data space is normal with mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma} = \boldsymbol{\Lambda}\boldsymbol{\Lambda}^T + \boldsymbol{\Psi}$, eq. (2.16). The submatrices of the covariance matrix are $\boldsymbol{\Sigma}_{\mathcal{I}\mathcal{I}} = \boldsymbol{\Lambda}_{\mathcal{I}}\boldsymbol{\Lambda}_{\mathcal{I}}^T + \boldsymbol{\Psi}_{\mathcal{I}}$, $\boldsymbol{\Sigma}_{\mathcal{J}\mathcal{J}} = \boldsymbol{\Lambda}_{\mathcal{J}}\boldsymbol{\Lambda}_{\mathcal{J}}^T + \boldsymbol{\Psi}_{\mathcal{J}}$, $\boldsymbol{\Sigma}_{\mathcal{I}\mathcal{J}} = \boldsymbol{\Lambda}_{\mathcal{I}}\boldsymbol{\Lambda}_{\mathcal{J}}^T$ with

$$\boldsymbol{\Lambda} = \begin{pmatrix} \boldsymbol{\Lambda}_{\mathcal{I}} \\ \boldsymbol{\Lambda}_{\mathcal{J}} \\ \cdots \end{pmatrix}, \qquad \boldsymbol{\Psi} = \begin{pmatrix} \boldsymbol{\Psi}_{\mathcal{I}} & \mathbf{0} & \cdots \\ \mathbf{0} & \boldsymbol{\Psi}_{\mathcal{J}} & \cdots \\ \cdots & \cdots & \cdots \end{pmatrix}.$$

Note that $\boldsymbol{\Lambda}$ is a $D \times L$ matrix and $\boldsymbol{\Psi}$ a diagonal $D \times D$ matrix. Therefore:

- $\mathbf{t}_{\mathcal{I}} \sim \mathcal{N}(\boldsymbol{\mu}_{\mathcal{I}}, \boldsymbol{\Lambda}_{\mathcal{I}}\boldsymbol{\Lambda}_{\mathcal{I}}^T + \boldsymbol{\Psi}_{\mathcal{I}})$.
- $\mathbf{t}_{\mathcal{J}}|\mathbf{t}_{\mathcal{I}} \sim \mathcal{N}(\boldsymbol{\mu}'_{\mathcal{J}}, \boldsymbol{\Sigma}'_{\mathcal{J}\mathcal{J}})$, where

$$\begin{aligned}
\boldsymbol{\mu}'_{\mathcal{J}} &= \boldsymbol{\mu}_{\mathcal{J}} + \boldsymbol{\Lambda}_{\mathcal{J}}(\mathbf{I} + \boldsymbol{\Lambda}_{\mathcal{I}}^T\boldsymbol{\Psi}_{\mathcal{I}}^{-1}\boldsymbol{\Lambda}_{\mathcal{I}})^{-1}\boldsymbol{\Lambda}_{\mathcal{I}}^T\boldsymbol{\Psi}_{\mathcal{I}}^{-1}(\mathbf{t}_{\mathcal{I}} - \boldsymbol{\mu}_{\mathcal{I}}) \\
&= \boldsymbol{\mu}_{\mathcal{J}} + \boldsymbol{\Lambda}_{\mathcal{J}}\boldsymbol{\Lambda}_{\mathcal{I}}^T(\boldsymbol{\Lambda}_{\mathcal{I}}\boldsymbol{\Lambda}_{\mathcal{I}}^T + \boldsymbol{\Psi}_{\mathcal{I}})^{-1}(\mathbf{t}_{\mathcal{I}} - \boldsymbol{\mu}_{\mathcal{I}}) \\
\boldsymbol{\Sigma}'_{\mathcal{J}\mathcal{J}} &= \boldsymbol{\Lambda}_{\mathcal{J}}(\mathbf{I} + \boldsymbol{\Lambda}_{\mathcal{I}}^T\boldsymbol{\Psi}_{\mathcal{I}}^{-1}\boldsymbol{\Lambda}_{\mathcal{I}})^{-1}\boldsymbol{\Lambda}_{\mathcal{J}}^T + \boldsymbol{\Psi}_{\mathcal{J}}.
\end{aligned}$$

Principal component analysis Now the covariance matrix is $\boldsymbol{\Sigma} = \boldsymbol{\Lambda}\boldsymbol{\Lambda}^T + \sigma^2\mathbf{I}$. Similarly:

- $\mathbf{t}_{\mathcal{I}} \sim \mathcal{N}(\boldsymbol{\mu}_{\mathcal{I}}, \boldsymbol{\Lambda}_{\mathcal{I}}\boldsymbol{\Lambda}_{\mathcal{I}}^T + \sigma^2\mathbf{I})$.
- $\mathbf{t}_{\mathcal{J}}|\mathbf{t}_{\mathcal{I}} \sim \mathcal{N}(\boldsymbol{\mu}'_{\mathcal{J}}, \boldsymbol{\Sigma}'_{\mathcal{J}\mathcal{J}})$, where

$$\begin{aligned}
\boldsymbol{\mu}'_{\mathcal{J}} &= \boldsymbol{\mu}_{\mathcal{J}} + \boldsymbol{\Lambda}_{\mathcal{J}}(\boldsymbol{\Lambda}_{\mathcal{I}}^T\boldsymbol{\Lambda}_{\mathcal{I}} + \sigma^2\mathbf{I})^{-1}\boldsymbol{\Lambda}_{\mathcal{I}}^T(\mathbf{t}_{\mathcal{I}} - \boldsymbol{\mu}_{\mathcal{I}}) \\
&= \boldsymbol{\mu}_{\mathcal{J}} + \boldsymbol{\Lambda}_{\mathcal{J}}\boldsymbol{\Lambda}_{\mathcal{I}}^T(\boldsymbol{\Lambda}_{\mathcal{I}}\boldsymbol{\Lambda}_{\mathcal{I}}^T + \sigma^2\mathbf{I})^{-1}(\mathbf{t}_{\mathcal{I}} - \boldsymbol{\mu}_{\mathcal{I}}) \\
\boldsymbol{\Sigma}'_{\mathcal{J}\mathcal{J}} &= \sigma^2\boldsymbol{\Lambda}_{\mathcal{J}}(\boldsymbol{\Lambda}_{\mathcal{I}}^T\boldsymbol{\Lambda}_{\mathcal{I}} + \sigma^2\mathbf{I})^{-1}\boldsymbol{\Lambda}_{\mathcal{J}}^T + \sigma^2\mathbf{I}.
\end{aligned}$$

For both models, the regression of $\mathbf{t}_{\mathcal{J}}$ on $\mathbf{t}_{\mathcal{I}}$, where $\mathbf{t}_{\mathcal{J}}$ takes the value $\boldsymbol{\mu}'_{\mathcal{J}}$, is a linear manifold passing through the conditional mean. Note that $\boldsymbol{\mu}'_{\mathcal{J}}$ depends (linearly) on $\mathbf{t}_{\mathcal{I}}$, but $\boldsymbol{\Sigma}'_{\mathcal{J}\mathcal{J}}$ (and therefore covariance-based error bars derived from it, as in section 8.4) does not depend on $\mathbf{t}_{\mathcal{I}}$ but just on $\mathcal{I}$ and $\mathcal{J}$.

7.12.3.2 Latent variable models with fixed sampling in the latent space

Generative topographic mapping (GTM) Applying eqs. (7.7) and (7.9) to eq. (2.43) gives:

- $p(\mathbf{t}_{\mathcal{I}}) = \sum_{k=1}^K p(\mathbf{x}_k)(2\pi\sigma^2)^{-\frac{I}{2}} e^{-\frac{1}{2\sigma^2}\|\mathbf{f}_{\mathcal{I}}(\mathbf{x}_k) - \mathbf{t}_{\mathcal{I}}\|^2} = \frac{1}{K}(2\pi\sigma^2)^{-\frac{I}{2}} \sum_{k=1}^K e^{-\frac{1}{2\sigma^2}\|\mathbf{f}_{\mathcal{I}}(\mathbf{x}_k) - \mathbf{t}_{\mathcal{I}}\|^2}$.
- $p(\mathbf{t}_{\mathcal{J}}|\mathbf{t}_{\mathcal{I}}) = \sum_{k=1}^K p(\mathbf{x}_k)(2\pi\sigma^2)^{-\frac{I}{2}}(2\pi\sigma^2)^{-\frac{J}{2}} e^{-\frac{1}{2\sigma^2}\|\mathbf{f}_{\mathcal{I}}(\mathbf{x}_k) - \mathbf{t}_{\mathcal{I}}\|^2} e^{-\frac{1}{2\sigma^2}\|\mathbf{f}_{\mathcal{J}}(\mathbf{x}_k) - \mathbf{t}_{\mathcal{J}}\|^2} / p(\mathbf{t}_{\mathcal{I}}) =$

$$(2\pi\sigma^2)^{-\frac{J}{2}} \frac{\sum_{k=1}^K e^{-\frac{1}{2\sigma^2}\|\mathbf{f}_{\mathcal{I}}(\mathbf{x}_k) - \mathbf{t}_{\mathcal{I}}\|^2} e^{-\frac{1}{2\sigma^2}\|\mathbf{f}_{\mathcal{J}}(\mathbf{x}_k) - \mathbf{t}_{\mathcal{J}}\|^2}}{\sum_{k=1}^K e^{-\frac{1}{2\sigma^2}\|\mathbf{f}_{\mathcal{I}}(\mathbf{x}_k) - \mathbf{t}_{\mathcal{I}}\|^2}}.$$

where $I \stackrel{\text{def}}{=} \operatorname{card}(\mathcal{I})$ and $J \stackrel{\text{def}}{=} \operatorname{card}(\mathcal{J})$ are the cardinalities of each index set (the number of variables in each set). The rightmost part of the equations is for a uniform distribution over the latent grid, i.e., $p(\mathbf{x}_k) = \frac{1}{K}\ \forall k = 1, \dots, K$.

7.12.4 A quantitative measure of sparseness

We propose here a preliminary study of quantitative measures of sparseness. Call a particular such measure $\mathscr{S}$ (not to be confused with the smoothness constraint of section 7.6.1). $\mathscr{S}$ maps a density (belonging to a given class of densities defined on a compact subset of $\mathbb{R}^D$) onto an interval, possibly infinite, of the real line; high (low) values of $\mathscr{S}$ mean more (less) sparseness[45]. $\mathscr{S}$ should have some desirable properties, such as:

1. *Maximal sparseness*: $\mathscr{S}$ should give high value to a mixture of delta distributions.

[45] A *hard* classification of densities into sparse and not sparse could be attained by thresholding the value of $\mathscr{S}$, but this always entails some arbitrariness in the choice of the threshold.

2. *Minimal sparseness*: $\mathscr{S}$ should give low value to a uniform distribution.

3. *Domain increase*: $\mathscr{S}$ should increase its value if a zero-density nonempty compact subset is added to the domain of the density (i.e., if the domain becomes larger but the density remains the same). For example, a uniform distribution in the interval $[0, 1]$ looks broad if the domain is $[0, 1]$ but sparse if the domain is $[0, 100]$. However, two densities should only be compared in terms of sparseness if they are defined in the same domain.

4. *Transformations*: $\mathscr{S}$ should be invariant to rigid motions (translations, rotations, flips) but, if the domain is $\mathbb{R}^D$, not to scale changes: stretching (compressing) a variable of infinite domain decreases (increases) its sparseness.

At first sight, an **entropy**-based measure (Cover and Thomas, 1991) would seem to fit the bill; but things are a bit more complicated. Firstly, the differential entropy by itself is a relative measure and so must be compared to a reference distribution p_r:

$$\mathscr{S}(p) \stackrel{\text{def}}{=} -(h(p) - h(p_r)) \tag{7.10}$$

where the negative sign ensures that $\mathscr{S}(p)$ increases the narrower p is. If p_r is a uniform distribution, this equation can also be interpreted as the Kullback-Leibler divergence of density p to density p_r:

$$\mathrm{D}\left(p \| p_r\right) = \int p \ln \frac{p}{p_r} = \int p \ln p - \int p \ln \frac{1}{V} = -h(p) + \ln V = -(h(p) - h(p_r)).$$

For bounded domains we could take as reference a uniform distribution on the domain, whose entropy (equal to $\ln V$, $V \stackrel{\text{def}}{=} \int_{\text{domain}} d\mathbf{t}$ being the domain volume) is maximal over all distributions and its sparseness minimal. But for unbounded domains, such uniform distribution does not exist: we can make the entropy arbitrarily large and so there is not a minimal sparseness distribution (in other words, we cannot subtract $h(p_r) = \infty$ in eq. (7.10)). But we could just take a different distribution as reference—it may not be as meaningful as the "minimal sparseness distribution," but as long as we always use the same reference, this is not a problem.

Secondly, it is easy to check that the entropy-based measure satisfies most of the previous conditions:

2. *Minimal sparseness*: since $h = \ln V$ for a uniform distribution of support size V, $\mathscr{S}$ decreases indefinitely as the support increases.

3. *Domain increase*: $h(p)$ does not change, but the reference distribution does, since it must cover the new region added; thus $h(p_r)$ increases and $\mathscr{S}$ increases.

4. *Transformations*: the entropy satisfies $h(\mathbf{u}) = h(\mathbf{t}) + \ln |\mathbf{A}|$ under an affine transformation $\mathbf{u} \stackrel{\text{def}}{=} \mathbf{A}\mathbf{t} + \mathbf{b}$ (here $|\mathbf{A}|$ is the absolute value of the determinant of $\mathbf{A}$, i.e., $|\mathbf{A}| \stackrel{\text{def}}{=} \prod_{d=1}^{D} |\lambda_d|$ in terms of $\mathbf{A}$'s eigenvalues). Since $|\mathbf{A}| = 1$ for both rotations ($\mathbf{A}$ is orthogonal) and flips ($\mathbf{A}$ is diagonal with entries ± 1 in the diagonal), h is invariant to rigid motions, and so is $\mathscr{S}$. For scale changes, if the domain is bounded then the reference distribution is also scaled and $\mathscr{S}$ does not change value; but if the domain is unbounded then the reference distribution does not change (because the domain does not change either) while $\mathscr{S}$ changes by $-\ln |\mathbf{A}| = -\sum_{d=1}^{D} \ln |\lambda_d|$ (stretching the dth axis gives $|\lambda_d| > 1$ and so $\mathscr{S}$ decreases, and vice versa).

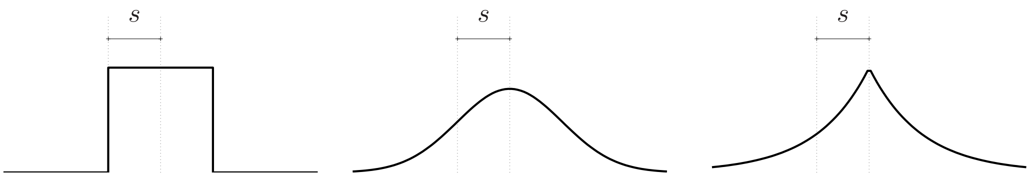
Where the measure of eq. (7.10) falls short is in the way it deals with delta functions. A delta function has $h = -\infty$ and so any distribution which is a mixture of a delta and any other density will have $h = -\infty$ as well, which is not reasonable.

In summary, an entropy-based measure of sparseness is not completely satisfactory. A topic of future research is then to identify a set of conditions that an ideal measure should satisfy, possibly including all conditions mentioned above, and then determine what quantitative measures verify them.

7.12.4.1 Other measures of sparseness

Kurtosis, the fourth-order cumulant of a distribution, has been proposed and used by Field (1994) and others (Olshausen and Field, 1996, 1997; Bell and Sejnowski, 1997; Vinje and Gallant, 2000) as a measure of sparseness in the context of neural codes, based on the experimental observation that the receptive fields of neurons in primary visual cortex typically have positive kurtosis—although this fact has been debated both theoretically and experimentally (e.g. Fyfe and Baddeley, 1995; Baddeley, 1996; Willmore et al., 2000). However, kurtosis

Distribution



	Uniform	Normal	Laplace (double exponential)
Density	$p_{\mathcal{U}}(x) = \begin{cases} 0 & x \notin [-s,s] \\ \frac{1}{2s} & x \in [-s,s] \end{cases}$	$p_{\mathcal{N}}(x) = \frac{1}{\sqrt{2\pi s^2}} e^{-\frac{1}{2}\left(\frac{x}{s}\right)^2}$	$p_{\mathcal{L}}(x) = \frac{1}{2s} e^{-\left\lvert \frac{x}{s} \right\rvert}$
Variance	$\mu_2 = \frac{s^2}{3}$	$\mu_2 = s^2$	$\mu_2 = 2s^2$
Entropy	$h(p_{\mathcal{U}}) = \ln 2s$	$h(p_{\mathcal{N}}) = \ln \sqrt{2\pi e} s$	$h(p_{\mathcal{L}}) = \ln 2es$
Kurtosis	$k_4 = -\frac{2}{15}s^4 < 0$	$k_4 = 0$	$k_4 = 12s^4 > 0$

Table 7.3: The kurtosis is not a good sparseness measure. Each of the densities shown takes values in $(-\infty, \infty)$ and depends on a width parameter $s > 0$. The entropy of each density can vary between $-\infty$ (for $s \to 0$, maximal sparseness) and $+\infty$ (for $s \to \infty$, minimal sparseness) but the kurtosis type (negative, zero or positive) does not depend on the width s. Thus, kurtosis is unrelated to entropy and does not measure distribution sparseness as defined in sections 7.3.1 and 7.12.4.

is not a good measure of sparseness in the sense we have described above, since we are not interested in the shape (or skew) of a peak, but in its narrowness and in the narrowness of the other peaks. In fact, as table 7.3 shows, we can have a distribution depending on one variance parameter (uniform, normal, Laplace) with the same kurtosis type independent of that parameter (negative, zero, positive) but whose width can vary from zero (maximum sparseness, entropy $= -\infty$) to infinity (minimum sparseness, entropy $= +\infty$).

As for the **variance**, it would be an appropriate measure of sparseness for unimodal distributions, but not for multimodal ones, since in this case the variance depends not just on the peaks' width but also on their separations (this also applies to the kurtosis), as fig. 7.19 demonstrates. In fact, by taking the extreme case of sparse distribution, a mixture of delta functions, we can obtain any desired value for its variance and kurtosis by varying the separation between individual deltas, while its differential entropy remains constant at $-\infty$.

Yet other definitions of sparseness are based on the **coefficient of variation** $c \stackrel{\text{def}}{=} \frac{\sigma}{\mu}$ (for univariate distributions), where σ is the standard deviation and μ the mean; Treves and Rolls (1991) and Rolls and Treves (1998, p. 326–329) define sparseness as $\frac{1}{1+c^2}$ to measure the proportion of very active cells at any particular moment in time or the proportion of time that a particular cell is very active, in the context of sparse coding and cell tuning. In all these cases, the entropy gives a more natural measure of sparseness as we define it, notwithstanding the caveat discussed earlier.

Finally, it is important to remark that we use the word "information" in the opposite sense as in the traditional, information theory definition of information contained in a distribution. In the latter view, the more values a random variable can take, the more unpredictable its outcome is and the higher the information (and the higher the entropy). In our view, the fewer the possible values, the more predictable the outcome(s) are and the more informative the distribution is[46].

[46]Finding good names is a difficult art. Apart from "sparse" and "informative" distributions, we also considered "spread out" and "unconstrained," all of which ring unwanted bells.

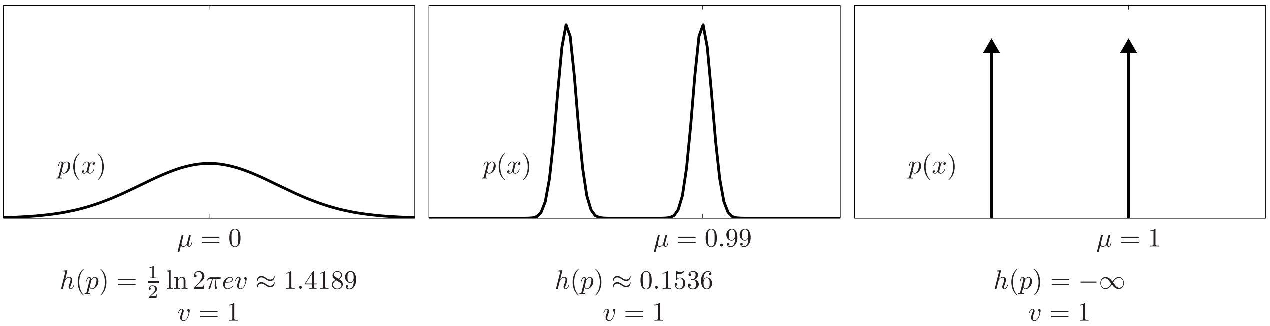


Figure 7.19: The variance is not a good sparseness measure for multimodal distributions. The graphs show distributions with the same variance v but different sparseness (and entropy ranging from $-\infty$ to $\frac{1}{2}\ln 2\pi e v$): $p(x) = \frac{1}{2}\mathcal{N}(-\mu, \sigma^2(\mu)) + \frac{1}{2}\mathcal{N}(\mu, \sigma^2(\mu))$ with $\sigma^2(\mu) = v - \mu^2$.

Chapter 8

Exhaustive mode finding in Gaussian mixtures

This chapter is organised as follows. Section 8.1 motivates the problem. Sections 8.2–8.3 describe algorithms for locating the modes. The significance of the modes thus obtained is quantified locally by computing error bars for each mode (section 8.4) and globally by measuring the sparseness of the mixture via the entropy (section 8.5). Section 8.6 concludes and mentions some applications. The chapter relies heavily on section 8.7, which gives formulae for the moments, gradient, Hessian and entropy of a Gaussian mixture, as well as for bounds on them.

8.1 Introduction

Gaussian mixtures (Titterington et al., 1985; Everitt and Hand, 1981; McLachlan and Peel, 2000) are ubiquitous probabilistic models for density estimation in machine learning applications. Their popularity is due to several reasons:

- Since they are a linear combination of Gaussian densities, they inherit some of the advantages of the Gaussian distribution: they are analytically tractable for many types of computations, have desirable asymptotic properties (e.g. the central limit theorem) and scale well with the data dimensionality. Furthermore, many natural data sets occur in clusters which are approximately Gaussian.

- The family of Gaussian mixtures is a universal approximator for continuous densities. In fact, Gaussian kernel density estimation (spherical Gaussian mixtures) can approximate any continuous density given enough kernels (Titterington et al., 1985; Scott, 1992), in the sense of (uniform or L_2-norm) convergence in probability of the estimator to the density. In particular, they can model multimodal distributions.

- Many complex models result in a Gaussian mixture after some assumptions are made in order to obtain tractable models. For example, Monte Carlo approximations to integrals of the type:

$$p(\mathbf{t}) = \int p(\mathbf{t}|\boldsymbol{\theta})p(\boldsymbol{\theta})\, d\boldsymbol{\theta} \approx \frac{1}{N}\sum_{n=1}^{N} p(\mathbf{t}|\boldsymbol{\theta}_n)$$

  where $\{\boldsymbol{\theta}_n\}_{n=1}^N$ are samples from the distribution $p(\boldsymbol{\theta})$ and $p(\mathbf{t}|\boldsymbol{\theta}_n)$ is assumed Gaussian. This is the case of continuous latent variable models that sample the latent variables, described in section 2.4, such as the generative topographic mapping (GTM).

- A number of convenient algorithms for estimating the parameters (means, covariance matrices and mixing proportions) exist, such as the traditional EM for maximum-likelihood or more recent varieties, including Bayesian (and non-Bayesian) methods that, in theory, can tune the number of components needed as well as the other parameters (e.g. Richardson and Green, 1997; Roberts et al., 1998).

This chapter is mainly based on references Carreira-Perpiñán (1999a, 2000a) and can be read independently of the rest of the thesis.

Examples of models (not only for density estimation, but also for regression and classification) that often result in a Gaussian mixture include the GTM model, kernel density estimation (also called Parzen estimation; Scott, 1992), radial basis function networks (Bishop, 1995), mixtures of probabilistic principal component analysers (Tipping and Bishop, 1999a), mixtures of factor analysers (Hinton et al., 1997), support vector machines for density estimation (Weston et al., 1999; Vapnik and Mukherjee, 2000), models for conditional density estimation such as mixtures of experts (Jacobs et al., 1991) and mixture density networks (Bishop, 1995), the emission distribution of hidden Markov models for automatic speech recognition and other applications (Rabiner and Juang, 1993) and, of course, the Gaussian mixture model itself. Titterington et al. (1985) give an extensive list of successful applications of Gaussian mixtures.

Mixture models are not the only way to combine densities, though—for example, individual components may be combined multiplicatively rather than additively, as in logarithmic opinion pools (Genest and Zidek, 1986) or in the recent product-of-experts model (Hinton, 1999). This may be a more efficient way to model high-dimensional data which simultaneously satisfies several low-dimensional constraints: each expert is associated with a single constraint and gives high probability to regions that satisfy it and low probability elsewhere, so that the product acts as an AND operation.

Gaussian mixtures have often been praised for their ability to model multimodal distributions, where each mode represents a certain entity. For example, in visual modelling or object detection, a probabilistic model of a class of objects should account for multimodal distributions in order to prepresent multiple views of the object (Moghaddam and Pentland, 1997) or multiple object shapes in a cluttered scene (Isard and Blake, 1998). In missing data reconstruction and inverse problems, which is the case we are interested in here, multivalued mappings can be derived from the modes of the conditional distribution of the missing variables given the present ones. Finding the modes of posterior distributions is also important in Bayesian analysis (Gelman et al., 1995, chapter 9). However, the problem of finding the modes of this important class of densities seems to have received little attention—although the problem of finding modes in a data sample, related to clustering, has been studied (see section 7.9.9).

Thus, the problem addressed in this chapter is to find all the modes of a given Gaussian mixture (of known parameters). No direct methods exist for this even in the simplest special case of one-dimensional bi-component mixtures, so iterative numerical algorithms are necessary. Intuitively, it seems reasonable that the number of modes will be smaller than or equal to the number of components in the mixture: the more the different components interact (depending on their mutual separation and on their covariance matrices), the more they will coalesce and the fewer modes will appear. Besides, modes should always appear inside the region enclosed by the component centroids—more precisely, in their convex hull.

We formalise these notions in the following conjecture, for which we provide a partial proof in section 8.7.2. First, let us recall that the convex hull of the vectors $\{\boldsymbol{\mu}_m\}_{m=1}^M$ is defined as the set

$$\left\{\mathbf{t} : \ \mathbf{t} = \sum_{m=1}^{M} \lambda_m \boldsymbol{\mu}_m \text{ with } \{\lambda_m\}_{m=1}^M \subset [0,1] \text{ and } \sum_{m=1}^{M} \lambda_m = 1\right\}.$$

Conjecture 8.1. *Let $p(\mathbf{t}) = \sum_{m=1}^M p(m)p(\mathbf{t}|m)$, where $\mathbf{t}|m \sim \mathcal{N}_D(\boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m)$, be a mixture of M D-variate normal distributions. Then $p(\mathbf{t})$ has M modes at most, all of which are in the convex hull of $\{\boldsymbol{\mu}_m\}_{m=1}^M$, if one of the following conditions holds:*

1. $D = 1$ (one-dimensional mixture)
2. *$D \geq 1$ and the covariance matrices are arbitrary but equal: $\boldsymbol{\Sigma}_m = \boldsymbol{\Sigma}$ $\forall m = 1, \dots, M$* (homoscedastic mixture).
3. *$D \geq 1$ and the covariance matrices are isotropic: $\boldsymbol{\Sigma}_m = \sigma_m^2 \mathbf{I}_D$* (spherical or isotropic mixture).

This conjecture suggests that, in the cases for which it holds, a hill-climbing algorithm starting from every centroid will not miss any mode. The analytical tractability of Gaussian mixtures allows a straightforward application of convenient optimisation algorithms and the computation of error bars. To our knowledge, this is the first time that the problem of finding all the modes of a Gaussian mixture has been investigated, although certainly the idea of using the gradient as mode locator is not new (e.g. Wilson and Spann, 1990).

The treatment in the following sections considers general Gaussian mixtures for simplicity of notation, but it only applies to the cases where conjecture 8.1 holds. The case of multivariate, arbitrary, different covariance matrices—for which the conjecture does not hold—is discussed in sections 8.6–8.7.

8.2 Exhaustive mode search by a gradient-quadratic search

Consider a Gaussian mixture p with $M > 1$ components as in equation (8.6). Since the family of Gaussian mixtures is a density universal approximator, the landscape of p could be very complex. However, assuming that conjecture 8.1 is true, there are at most M modes and it is clear that every centroid $\boldsymbol{\mu}_m$ of the mixture must be near, if not coincident, with one of the modes, since the modes are contained in the convex hull of the centroids (as the mean is). Thus, an obvious procedure to locate all the modes is to use a hill-climbing algorithm starting from every one of the centroids (and so from every vertex of the convex hull).

Due to the ease of calculation of the gradient (8.11) and the Hessian (8.12), it is straightforward to use quadratic maximisation (i.e., Newton's method) combined with gradient ascent (Nocedal and Wright, 1999). Assuming we are at a point $\mathbf{t}_0$, let us expand $p(\mathbf{t})$ around $\mathbf{t}_0$ as a Taylor series to second order:

$$p(\mathbf{t}) \approx p(\mathbf{t}_0) + (\mathbf{t} - \mathbf{t}_0)^T \mathbf{g}(\mathbf{t}_0) + \frac{1}{2}(\mathbf{t} - \mathbf{t}_0)^T \mathbf{H}(\mathbf{t}_0)(\mathbf{t} - \mathbf{t}_0)$$

where $\mathbf{g}(\mathbf{t}_0)$ and $\mathbf{H}(\mathbf{t}_0)$ are the gradient and Hessian of p at $\mathbf{t}_0$, respectively. The zero-gradient point of the previous quadratic form is given by:

$$\nabla p(\mathbf{t}) = \mathbf{g}(\mathbf{t}_0) + \mathbf{H}(\mathbf{t}_0)(\mathbf{t} - \mathbf{t}_0) = \mathbf{0} \Longrightarrow \mathbf{t} = \mathbf{t}_0 - \mathbf{H}^{-1}(\mathbf{t}_0)\mathbf{g}(\mathbf{t}_0) \tag{8.1}$$

which jumps from $\mathbf{t}_0$ to the maximum (or minimum, or saddle point) of the quadratic form in a single leap. Thus, for maximisation, the Hessian can only be used if it is negative definite, i.e., if all its eigenvalues are negative.

If the Hessian is not negative definite, which means that we are not yet in a hill-cap (defined as the region around a mode where $\mathbf{H} < 0$), we use gradient ascent:

$$\mathbf{t} = \mathbf{t}_0 + s\mathbf{g}(\mathbf{t}_0) \tag{8.2}$$

where $s > 0$ is the step size. That is, we jump a distance $s \left\| \mathbf{g}(\mathbf{t}_0) \right\|$ in the direction of the gradient (which does not necessarily point towards the maximum). For comments about the choice of the step size, see section 8.2.1.

Once a point for which $\mathbf{g} = \mathbf{0}$ is found, the Hessian (8.12) can confirm that this point is indeed a maximum by checking that $\mathbf{H} < 0$. Of course, both the nullity of the gradient and the negativity of the Hessian can only be ascertained to a certain numerical accuracy, but due to the simplicity of the surface of $p(\mathbf{t})$ this should not be a problem (at least for a small dimensionality D). Section 8.2.1 discusses the control parameters for the gradient ascent. Fig. 8.1 shows the pseudocode for the algorithm. Fig. 8.2 illustrates the case with a two-dimensional example.

Some remarks:

- It is not convenient here to use multidimensional optimisation methods based on line searches, such as the conjugate gradient method, because the line search may discard local maxima. Since we are interested in finding all the modes, we need a method that does not abandon the region of influence of a maximum. Gradient ascent with a small step followed by quadratic optimisation in a hill-cap should not miss local maxima.

- If the starting point is at or close to a stationary point, i.e., with near-zero gradient, the method will not iterate. Examination of the Hessian will determine if the point is a maximum, a minimum or a saddle point. In the latter two cases it will be discarded.

- The gradient ascent should not suffer too much in higher dimensions because the search follows a one-dimensional path. Of course, this path can twist itself in many more dimensions and thus became longer, but once it reaches a hill-cap, quadratic maximisation converges quickly. If the dimension of the space is D, computing the gradient and the Hessian is $\mathcal{O}(D)$ and $\mathcal{O}(D^2)$, respectively. Inverting the Hessian is $\mathcal{O}(D^3)$, but this may be reduced by the techniques of section 8.7.4.

- Other optimisation strategies based on the gradient or the Hessian, such as the Levenberg-Marquardt algorithm (Nocedal and Wright, 1999), can also be easily constructed.

```
inputs
    Gaussian mixture defined by {p(m), μ_m, Σ_m}_{m=1}^M
constants
    σ ← sqrt(min_{m=1,...,M} {min{eigenvalues(Σ_m)}})          Minimal standard deviation
    ε ← 10^{-4}                                                 Small number 0 < ε ≪ 1
    θ ← 10^{-2}                                                 Rejection threshold 0 < θ ≪ 1
control parameters
    min_step ← σ^2 (2πσ^2)^{D/2}
    min_grad ← σ^{-1} (2πσ^2)^{-D/2} ε e^{-(1/2)ε^2}
    min_diff ← 100εσ
    max_eig ← 0
    max_it ← 1000
initialise
    s ← 64*min_step                                             Step size
    M ← ∅                                                       Mode set
optionally
    Remove all components for which p(m) / max_{m=1,...,M} p(m) < θ and renormalise p(m)
for m = 1, ..., M                                               For each centroid
    τ ← 0                                                       Iteration counter
    t ← μ_m                                                     Starting point
    p ← p(t)                                                    From eq. (8.6)
    repeat                                                      Gradient-quadratic search loop
        g ← ∇ ln p(t)                                           Gradient from eq. (8.13)
        H ← (∇∇^T) ln p(t)                                      Hessian from eq. (8.14)
        t_old ← t
        p_old ← p
        if H < 0                                                Hessian negative definite?
            t ← t_old − H^{-1} g                                Quadratic step
            p ← p(t)                                            From eq. (8.6)
        end
        if H ≮ 0 or p ≤ p_old
            t ← t_old + s g                                     Gradient step
            p ← p(t)                                            From eq. (8.6)
            while p < p_old
                s ← s/2
                t ← t_old + s g                                 Gradient step
                p ← p(t)                                        From eq. (8.6)
            end
        end
        τ ← τ + 1
    until τ ≥ max_it or ‖g‖ < min_grad
    if max{eigenvalues(H)} < max_eig                            Update mode set
        N ← {ν ∈ M : ‖ν − t‖ ≤ min_diff} ∪ {t}
        M ← (M \ N) ∪ {arg max_{ν∈N} p(ν)}
    end
end
return M
```

Figure 8.1: Pseudocode of the gradient-quadratic mode-finding algorithm described in section 8.2. Instead of for $L(\mathbf{t}) = \ln p(\mathbf{t})$, the gradient and the Hessian can be computed for $p(\mathbf{t})$ using eqs. (8.11) and (8.12): $\mathbf{g} \leftarrow \nabla p(\mathbf{t})$, $\mathbf{H} \leftarrow (\nabla\nabla^T)p(\mathbf{t})$. Also, at each step one can compute the gradient and Hessian for both $p(\mathbf{t})$ and $\ln \mathbf{p}(\mathbf{t})$ and choose the one for which the new point has the highest probability.

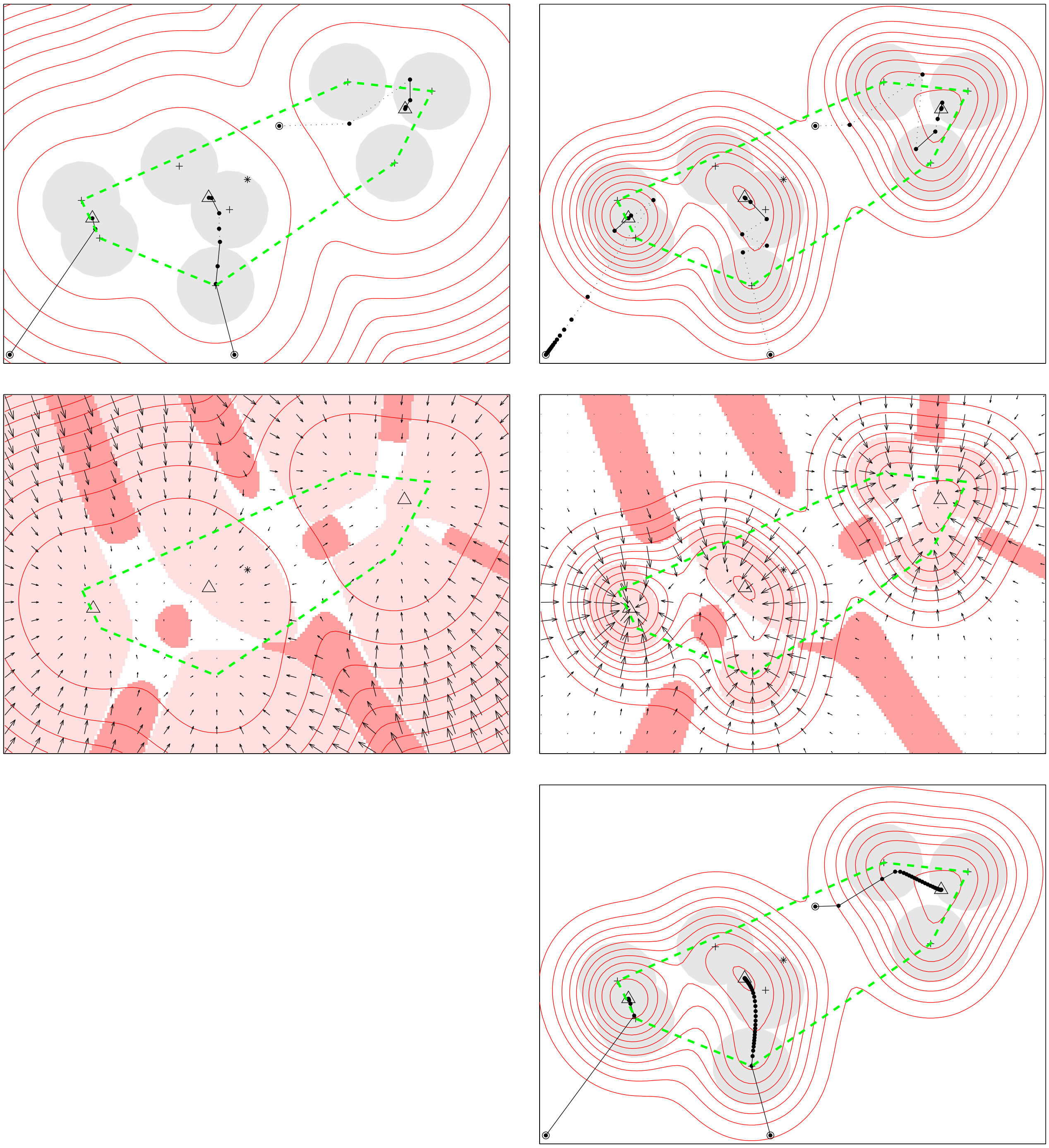


Figure 8.2: Example of mode searching in a two-dimensional Gaussian mixture. In this example, the mixing proportions are equal and the components are isotropic with equal covariance $\sigma^2\mathbf{I}_2$. The surface has 3 modes and various other features (saddle points, ridges, plateaux, etc.). The mixture modes are marked "△" and the mixture mean "$*$". The dashed, thick-line polygon is the convex hull of the centroids. The left column shows the surface of $L(\mathbf{t}) = \ln p(\mathbf{t})$ and the right column the surface of $p(\mathbf{t})$. *Top row*: contour plot of the objective function. Each original component is indicated by a grey disk of radius σ centred on the corresponding mean vector $\boldsymbol{\mu}_m$ (marked "+"). A few search paths from different starting points (marked "∘") are given for illustrative purposes (paths from the centroids are much shorter); continuous lines indicate gradient steps and dotted lines quadratic steps. *Middle row*: plot of the gradient (arrows) and the Hessian character (dark colour: positive definite; white: indefinite; light colour: negative definite). *Bottom row*: like the top row, but here the fixed-point iterative algorithm was used.

8.2.1 Control parameters for the gradient-quadratic mode-finding algorithm

We consider here a single D-dimensional isotropic Gaussian distribution $\mathcal{N}(\boldsymbol{\mu}, \sigma^2 \mathbf{I}_D)$ and derive some control parameters for the gradient ascent which may be transferrable to the mixture case: `min_step`, `min_grad`, `min_diff` and `max_eig`.

- For the isotropic Gaussian, the gradient always points to the mode, by symmetry. Thus, $\mathbf{g} = \|\mathbf{g}\| \frac{\boldsymbol{\mu}-\mathbf{t}}{\|\boldsymbol{\mu}-\mathbf{t}\|}$. Consider a point at a normalised distance $\rho = \left\|\frac{\mathbf{t}-\boldsymbol{\mu}}{\sigma}\right\|$ from the mode $\boldsymbol{\mu}$. Then $\mathbf{g} = \frac{\|\mathbf{g}\|}{\rho\sigma}(\boldsymbol{\mu}-\mathbf{t})$. Thus, a step $s = \frac{\rho\sigma}{\|\mathbf{g}\|}$ would jump directly to the mode: $\mathbf{t}_{\text{new}} = \mathbf{t} + s\mathbf{g} = \mathbf{t} + \frac{\rho\sigma}{\|\mathbf{g}\|}\frac{\|\mathbf{g}\|}{\rho\sigma}(\boldsymbol{\mu}-\mathbf{t}) = \boldsymbol{\mu}$. From eq. (8.9) $\|\mathbf{g}\| = p(\mathbf{t}) \left\|\frac{\mathbf{t}-\boldsymbol{\mu}}{\sigma^2}\right\| = \sigma^{-1}(2\pi\sigma^2)^{-\frac{D}{2}} e^{-\frac{1}{2}\rho^2}\rho \le \sigma^{-1}(2\pi\sigma^2)^{-\frac{D}{2}}\rho$ and so $s \ge \sigma^2(2\pi\sigma^2)^{\frac{D}{2}}$. Therefore, a step of `min_step` $= \sigma^2(2\pi\sigma^2)^{\frac{D}{2}}$ times the gradient would never overshoot, that is, would climb up the hill monotonically. However, it would be too small for points a few normalised distances away from the mode. Moreover, theorem 8.7.8 shows that the gradient norm for the mixture is never larger than that of any isolated component, which suggests using even larger step sizes. Thus, our gradient ascent algorithm starts with a step size of several (64, corresponding to a point at 2.88 normalised distances away from the mode) times the previous step size and halves it every time the new point has a worse probability.

- To determine when a gradient norm is considered numerically zero, we choose the points for which the normalised distance is less than a small value ϵ (set to 10^{-4}). This gives a minimum gradient norm `min_grad` $= \sigma^{-1}(2\pi\sigma^2)^{-\frac{D}{2}}\epsilon e^{-\frac{1}{2}\epsilon^2} \approx \sigma^{-1}(2\pi\sigma^2)^{-\frac{D}{2}}\epsilon$. Since for the mixtures the gradient will be smaller than for the components, points with a gradient norm of `min_grad` may be farther from the mode than they would be in the case of a single mixture, so ϵ should be small compared to 1.

- Thus, since the algorithm will not get closer to the mode than a normalised distance of ϵ, the normalised distance below which two points are considered the same may be taken as several times larger than ϵ. We take `min_diff` $= 100\epsilon\sigma$ as the minimum absolute difference between two modes to be assimilated as one.

  Updating the mode set $\mathcal{M}$ in the algorithm after a new mode $\mathbf{t}$ has been found is achieved by identifying all modes previously found that are closer to $\mathbf{t}$ than a distance `min_diff`, including $\mathbf{t}$ (the $\mathcal{N}$ set in figures 8.1 and 8.3), removing them from the mode set $\mathcal{M}$ and adding to $\mathcal{M}$ the mode in $\mathcal{N}$ with highest probability p.

- The Hessian will be considered negative definite if its algebraically largest eigenvalue is less than a nonnegative parameter `max_eig`. We take `max_eig` $= 0$, since theorem 8.7.9 shows that not too far from a mode (for the case of one component, inside a radius of one normalised distance), the Hessian will already be negative definite. This strict value of `max_eig` will rule out all minima (for which $\|\mathbf{g}\| = 0$ but $\mathbf{H} > 0$) and should not miss any maximum.

- To limit the computation time, we define `max_it` as the maximum number of iterations to be performed.

These results can be easily generalised to a mixture of full-covariance Gaussians. Theorems 8.7.8 and 8.7.8 show that the mixture gradient anywhere is bounded above and depends on the smallest eigenvalue of the covariance matrix of any of the components. Calling this minimal eigenvalue σ^2, the control parameter definitions given above remain the same.

In high dimensions, these parameters may require manual tuning (perhaps using knowledge of the particular problem being tackled), specially if too many modes are obtained—due to the nature of the geometry of high-dimensional spaces (section 4.3.1; Scott, 1992). For example, both `min_step` and `min_grad` depend exponentially on D, which can lead to very large or very small values depending on the value of σ. For `min_diff`, consider the following situation: vectors $\mathbf{t}_1$ and $\mathbf{t}_2$ differ in a small value δ in each component, so that $\|\mathbf{t}_1 - \mathbf{t}_2\| = \sqrt{D}\delta$. For high D, $\|\mathbf{t}_1 - \mathbf{t}_2\|$ will be large even though one would probably consider $\mathbf{t}_1$ and $\mathbf{t}_2$ as the same vector. However, if that difference was concentrated on a single component, one would probably consider them as very different vectors.

Finally, we remark that there is a lower bound in the precision achievable by any numerical algorithm due to the finite-precision arithmetic (Press et al., 1992, pp. 398–399; Nocedal and Wright, 1999, pp.167–168), so that in general we cannot get arbitrarily close to a scalar value μ: at best, our estimate t we will get to $|t-\mu| \sim \mu\sqrt{\epsilon_m}$, where ϵ_m is the machine accuracy (usually $\epsilon_m \approx 3 \times 10^{-8}$ for simple precision and $\epsilon_m \approx 10^{-15}$ for double precision). This gives a limit in how small to make all the control parameters mentioned. Furthermore, converging to many decimals is a waste, since the mode is at best only a (nonrobust) statistical estimate based on our model—whose parameters were also estimated to some precision.

8.2.2 Maximising the density p vs maximising the log-density $L = \ln p$

Experimental results show that, when the component centroids $\boldsymbol{\mu}_m$ are used as starting points, there is not much difference in speed of convergence between using the gradient and Hessian of $p(\mathbf{t})$ and using those of $L(\mathbf{t}) = \ln p(\mathbf{t})$; although there is difference from other starting points, e.g. far from the convex hull, where $p(\mathbf{t})$ is very small. Fig. 8.2 illustrates this: in the top row, observe the slow search in points lying in areas of near-zero probability in the case of $p(\mathbf{t})$ and the switch from gradient to quadratic search when the point is in a hill-cap, where the Hessian is negative definite. In the middle row, observe how much bigger the areas with negative definite Hessian are for the surface of $L(\mathbf{t})$ in regions where $p(\mathbf{t})$ is small, as noted in section 8.7.1.1. This means that, for starting points in regions where $p(\mathbf{t})$ is small, quadratic steps can be taken more often and thus convergence is faster. However, the centroids are usually in areas of high $p(\mathbf{t})$ and thus there is no improvement for our mode-finding algorithm.

In any case, at each step one can compute the gradient and Hessian for both $p(\mathbf{t})$ and $\ln p(\mathbf{t})$ and choose the one for which the new point has the highest probability.

It may be argued that L is a quadratic form if p is Gaussian, in which case a quadratic optimiser would find the maximum in a single step. However, p will be far from Gaussian even near the centroids or modes if the mixture components interact strongly (i.e., if they are close enough with respect to their covariance matrices, as in fig. 8.2) and this will be the case when the mixture is acting as a density approximator (as in kernel estimation).

8.2.3 Low-probability components

Gaussian mixtures are usually applied to high-dimensional data. Due to computational difficulties and to the proverbial lack of sufficient training data (both issues arising from the curse of the dimensionality; section 4.3, Scott, 1992), the estimated mixture may not be a good approximation to the density of the data. If this is the case, some of the modes found may be spurious, due to artifacts of the model. A convenient way to filter some of them out is to reject all modes whose probability (normalised by the probability of the highest mode) is smaller than a certain small threshold $\theta > 0$ (e.g. $\theta = 0.01$). This does not remove all spurious modes; for example, spurious ripple superimposed on a bump implies spurious modes of high probability density value. See sections 7.9.1 and 9.2.3 for discussions on the effect of spurious modes in missing data reconstruction.

Consider the particular case when the mixture whose modes are to be found is the result of computing the conditional distribution of a joint Gaussian mixture given the values of certain variables $\mathbf{y}$. It is easy to see (section 7.12.1) that the conditional distribution is again a Gaussian mixture where the new mixing proportions are

$$p(m|\mathbf{y}) = \frac{p(\mathbf{y}|m)p(m)}{p(\mathbf{y})} \propto p(m) e^{-\frac{1}{2}(\mathbf{y}-\boldsymbol{\mu}_{m,y})^T \boldsymbol{\Sigma}_{m,yy}^{-1} (\mathbf{y}-\boldsymbol{\mu}_{m,y})}$$

for certain $\boldsymbol{\mu}_{m,y}$ and $\boldsymbol{\Sigma}_{m,yy}$. Thus, the means (projected in the $\mathbf{y}$ axes) of most components will be far from the value of $\mathbf{y}$ and will have a negligible mixing proportion $p(m|\mathbf{y})$. Filtering out such low-probability components will accelerate considerably the mode search without missing any important mode.

8.3 Exhaustive mode search by a fixed-point search

Equating the gradient expression (8.11) to zero we obtain immediately a fixed-point iterative scheme:

$$\begin{aligned} \mathbf{g} &= \sum_{m=1}^{M} p(\mathbf{t}, m) \boldsymbol{\Sigma}_m^{-1} (\boldsymbol{\mu}_m - \mathbf{t}) = \mathbf{0} \\ \Longrightarrow \mathbf{t} &= \mathbf{f}(\mathbf{t}) \text{ with } \mathbf{f}(\mathbf{t}) \stackrel{\text{def}}{=} \left(\sum_{m=1}^{M} p(m|\mathbf{t}) \boldsymbol{\Sigma}_m^{-1} \right)^{-1} \sum_{m=1}^{M} p(m|\mathbf{t}) \boldsymbol{\Sigma}_m^{-1} \boldsymbol{\mu}_m \end{aligned} \tag{8.3}$$

where we have used Bayes' theorem. Note that using the gradient of $L(\mathbf{t}) = \ln p(\mathbf{t})$ makes no difference here. A fixed point $\mathbf{t}$ of the mapping $\mathbf{f}$ verifies by definition $\mathbf{t} = \mathbf{f}(\mathbf{t})$. The fixed points of $\mathbf{f}$ are thus the stationary points of the mixture density p, including maxima, minima and saddle points. The iterative scheme $\mathbf{t}^{(\tau+1)} = \mathbf{f}(\mathbf{t}^{(\tau)})$ converges to a fixed point of $\mathbf{f}$, as we prove in section 8.7.3. In a number of experiments it has found exactly the same modes as the gradient-quadratic method. Thus, as in section 8.2, iterating from each centroid should find all maxima, since at least some of the centroids are likely to be near the modes. Checking the eigenvalues of the Hessian of p with eq. (8.12) will determine whether the point found is actually a maximum.

inputs
  Gaussian mixture defined by $\{p(m), \boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m\}_{m=1}^M$
constants
  $\sigma \leftarrow \sqrt{\min_{m=1,\dots,M}\{\min\{\text{eigenvalues}(\boldsymbol{\Sigma}_m)\}\}}$ — Minimal standard deviation
  $\epsilon \leftarrow 10^{-4}$ — Small number $0 < \epsilon \ll 1$
  $\theta \leftarrow 10^{-2}$ — Rejection threshold $0 < \theta \ll 1$
control parameters
  $\texttt{tol} \leftarrow \epsilon\sigma$
  $\texttt{min\_diff} \leftarrow 100\epsilon\sigma$
  $\texttt{max\_eig} \leftarrow 0$
  $\texttt{max\_it} \leftarrow 1000$
initialise
  $\mathcal{M} \leftarrow \varnothing$ — Mode set
optionally
  Remove all components for which $\frac{p(m)}{\max_{m=1,\dots,M} p(m)} < \theta$ and renormalise $p(m)$
for $m = 1, \dots, M$ — For each centroid
  $\tau \leftarrow 0$ — Iteration counter
  $\mathbf{t} \leftarrow \boldsymbol{\mu}_m$ — Starting point
  repeat — Fixed-point iteration loop
    $\mathbf{t}_{\text{old}} \leftarrow \mathbf{t}$
    $\mathbf{t} \leftarrow \left(\sum_{m=1}^M p(m|\mathbf{t})\boldsymbol{\Sigma}_m^{-1}\right)^{-1} \sum_{m=1}^M p(m|\mathbf{t})\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\mu}_m$ — From eq. (8.3)
    $\tau \leftarrow \tau + 1$
  until $\tau \geq \texttt{max\_it}$ **or** $\|\mathbf{t} - \mathbf{t}_{\text{old}}\| < \texttt{tol}$
  $\mathbf{H} \leftarrow (\nabla\nabla^T)p(\mathbf{t})$ — Hessian from eq. (8.12)
  if $\max\{\text{eigenvalues}(\mathbf{H})\} < \texttt{max\_eig}$ — Update mode set
    $\mathcal{N} \leftarrow \{\boldsymbol{\nu} \in \mathcal{M} : \|\boldsymbol{\nu} - \mathbf{t}\| \leq \texttt{min\_diff}\} \cup \{\mathbf{t}\}$
    $\mathcal{M} \leftarrow (\mathcal{M} \setminus \mathcal{N}) \cup \{\arg\max_{\boldsymbol{\nu}\in\mathcal{N}} p(\boldsymbol{\nu})\}$
  end
end
return $\mathcal{M}$

Figure 8.3: Pseudocode of the fixed-point mode-finding algorithm described in section 8.3.

The inverse matrix $\left(\sum_{m=1}^M p(m|\mathbf{t})\boldsymbol{\Sigma}_m^{-1}\right)^{-1}$ may be trivially computed in some cases (e.g. if all the components are diagonal). In the particular[1] case where $\boldsymbol{\Sigma}_m = \boldsymbol{\Sigma}$ for all $m = 1, \dots, M$, the fixed-point scheme reduces to the extremely simple form

$$\mathbf{t}^{(\tau+1)} = \sum_{m=1}^M p(m|\mathbf{t}^{(\tau)})\boldsymbol{\mu}_m, \tag{8.4}$$

i.e., the new point $\mathbf{t}^{(\tau+1)}$ is the conditional mean of the mixture under the current point $\mathbf{t}^{(\tau)}$. This is formally akin to EM algorithms for parameter estimation of mixture distributions (Dempster et al., 1977), to clustering by deterministic annealing (Rose, 1998) and to algorithms for finding pre-images in kernel-based methods (Schölkopf et al., 1999b).

The fixed-point iterative algorithm is much simpler than the gradient-quadratic one, but it also requires many more iterations to converge inside the convex hull of $\{\boldsymbol{\mu}_m\}_{m=1}^M$, as can be seen experimentally (observe in fig. 8.2 (bottom) the quick jump to the area of high probability and the slow convergence thereafter). This is due to the fact that the fixed-point iterative scheme is really an EM algorithm (see section 8.7.3), which is only a first-order method (Dempster et al., 1977; McLachlan and Krishnan, 1997); perhaps it could be accelerated by some of the techniques described by Meng and van Dyk (1997) and McLachlan and Krishnan

[1]Important models fall in this case, such as Gaussian kernel density estimation (Parzen estimators) or the generative topographic mapping (GTM), as well as Gaussian radial basis function networks.

(1997). Whether the fixed-point algorithm is faster than the gradient-quadratic one has to be determined for each particular case, depending on the values of D and M and the numerical routines used for matrix inversion. Note that the update of $\mathbf{t}$ in eq. (8.3) should be computed via the Cholesky factorisation of $\sum_{m=1}^{M} p(m|\mathbf{t})\boldsymbol{\Sigma}_m^{-1}$ (which is a positive definite matrix) followed by the solution of two triangular systems, rather than via its inversion, which is less accurate.

8.3.1 Control parameters for the fixed-point mode-finding algorithm

A theoretical advantage of the fixed-point scheme over gradient ascent is that no step size is needed. As in section 8.2.1, call σ^2 the smallest eigenvalue of the covariance matrix of any of the components. A new tolerance parameter `tol` $= \epsilon\sigma$ is defined for some small ϵ (10^{-4}), so that if the distance between two successive points is smaller than `tol`, we stop iterating. Alternatively, we could use the `min_grad` control parameter of section 8.2.1. The following control parameters from section 8.2.1 remain unchanged: `min_diff` $= 100\epsilon\sigma$, `max_eig` and `max_it`. Note that `min_diff` should be several times larger than `tol`.

8.4 Error bars for the modes

In this section we deal with the problem of deriving error bars, or confidence intervals, for a mode of a mixture of Gaussian distributions. That is, the shape of the distribution around that mode (how peaked or how spread out) contains information about the certainty of the value of the mode. These error bars are not related in any way to the numerical precision with which that mode was found by the iterative algorithm; they are related to the statistical dispersion around it.

The confidence interval, or in higher dimensions, the confidence hyperrectangle, means here a hyperrectangle containing the mode and with a probability under the mixture distribution of value P fixed in advance. For example, for $P = 0.9$ in one dimension we speak of a 90% confidence interval. Since computing error bars for the mixture distribution is analytically difficult, we follow an approximate computational approach: we replace the mixture distribution around the mode by a normal distribution centred in that mode and with a certain covariance matrix. Then we compute symmetric error bars for this normal distribution, which is easy, as we show in section 8.4.1. Section 8.4.2 deals with the problem of selecting the covariance matrix of the approximating normal.

Ideally we would like to have asymmetric bars, accounting for possible skewness of the distribution around the mode, but this is difficult in several dimensions. Also, in high dimensions the error bars become very wide, since due to the curse of the dimensionality the probability contained in a fixed hypercube decreases exponentially with the dimension (Scott, 1992).

8.4.1 Confidence intervals at the mode of a normal distribution

Consider a D-variate normal distribution $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ where $\boldsymbol{\Sigma} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^T$ is the singular value decomposition[2] of its covariance matrix $\boldsymbol{\Sigma}$. Given $\rho > 0$, $\mathcal{R} = \|\boldsymbol{\Lambda}^{-1/2}\mathbf{U}^T(\mathbf{t} - \boldsymbol{\mu})\|_\infty \leq \rho$ represents a hyperrectangle[3] with its centre on $\mathbf{t} = \boldsymbol{\mu}$. Its sides are aligned with the principal axes of $\boldsymbol{\Sigma}$, that is, $\mathbf{U} = (\mathbf{u}_1, \dots, \mathbf{u}_D)$, and have lengths $2\rho\sqrt{\lambda_1}, \dots, 2\rho\sqrt{\lambda_D}$ (see fig. 8.4a). The probability $P(\rho)$ contained in this hyperrectangle $\mathcal{R}$ can be computed as follows:

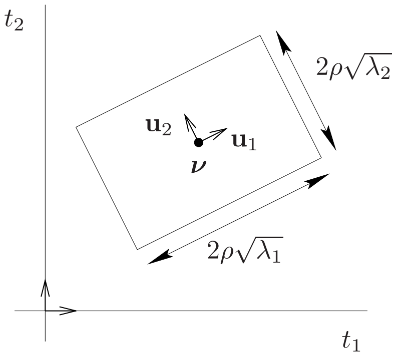
$$\begin{aligned} P(\rho) &= \int_{\mathcal{R}} |2\pi\boldsymbol{\Sigma}|^{-\frac{1}{2}} \, e^{-\frac{1}{2}(\mathbf{t}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{t}-\boldsymbol{\mu})} \, d\mathbf{t} \\ &= \int_{\|\mathbf{z}\|_\infty \leq \rho} |2\pi\boldsymbol{\Lambda}|^{-\frac{1}{2}} \, e^{-\frac{1}{2}\mathbf{z}^T\mathbf{z}} \, |\boldsymbol{\Lambda}|^{\frac{1}{2}} \, d\mathbf{z} \\ &= \prod_{d=1}^{D} P_{\mathcal{N}(0,1)} \left\{ z_d \in [-\rho, \rho] \right\} = \left(\operatorname{erf}\left(\frac{\rho}{\sqrt{2}} \right) \right)^D \end{aligned} \tag{8.5}$$

where we have changed $\mathbf{z} = \boldsymbol{\Lambda}^{-1/2}\mathbf{U}^T(\mathbf{t} - \boldsymbol{\mu})$, with Jacobian $|\boldsymbol{\Lambda}^{-1/2}\mathbf{U}^T| = |\boldsymbol{\Lambda}|^{-1/2}$, and the error function erf is defined as

$$\operatorname{erf}(x) \stackrel{\text{def}}{=} \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} \, dt = P_{\mathcal{N}(0,1)} \left\{ [-\sqrt{2}x, \sqrt{2}x] \right\}.$$

[2] In general, the singular value decomposition of a rectangular matrix $\mathbf{A}$ is $\mathbf{A} = \mathbf{U}\mathbf{S}\mathbf{V}^T$ with $\mathbf{U}$, $\mathbf{V}$ orthogonal and $\mathbf{S}$ diagonal, but for a symmetric square matrix $\mathbf{U} = \mathbf{V}$.

[3] Considering a hyperellipse $\mathcal{E} = \|\boldsymbol{\Lambda}^{-1/2}\mathbf{U}^T(\mathbf{t} - \boldsymbol{\mu})\|_2 \leq \rho$ instead of a hyperrectangle simplifies the analysis, but we are interested in separate intervals along each direction.



$$\boldsymbol{\Sigma} = \underbrace{\begin{pmatrix} \mathbf{u}_1 & \mathbf{u}_2 \end{pmatrix}}_{\mathbf{U}} \underbrace{\begin{pmatrix} \lambda_1 & \\ & \lambda_2 \end{pmatrix}}_{\boldsymbol{\Lambda}} \underbrace{\begin{pmatrix} \mathbf{u}_1^T \\ \mathbf{u}_2^T \end{pmatrix}}_{\mathbf{U}^T}$$

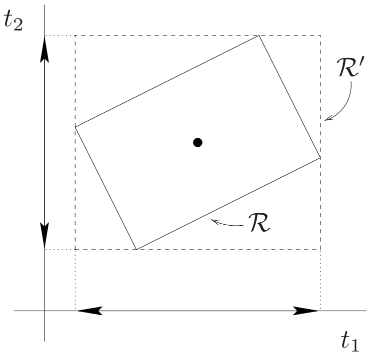


Figure 8.4: *Left*: schematic of the error bars, or hyperrectangle $\mathcal{R}$, in two dimensions. *Right*: how to obtain error bars in the original axes by circumscribing another hyperrectangle $\mathcal{R}'$ to $\mathcal{R}$. It is clear that $P(\mathcal{R}') \geq P(\mathcal{R})$.

$P^{1/D}$	ρ
1	∞
0.9973	3
0.9545	2
0.6827	1
0.5	0.6745

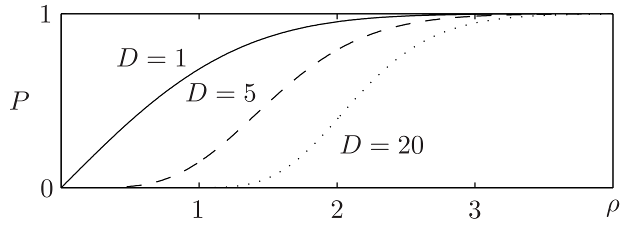


Figure 8.5: Probability P of a D-dimensional hypercube of side 2ρ centred in the mode of a D-dimensional normal distribution, from eq. (8.5). To obtain P from the table one has to raise the numbers on the left column to power D.

From (8.5), $\rho = \sqrt{2}\,\mathrm{arg\,erf}\,(P^{1/D})$, so that for a desired confidence level given by the probability P we can obtain the appropriate interval (see table in figure 8.5). The natural confidence intervals, or error bars, that $\mathcal{R}$ gives us are of the form $\left|\sum_{c=1}^{D} u_{cd}(t_c - \mu_c)\right| \leq \rho\sqrt{\lambda_d}$. They follow the directions of the principal axes of $\boldsymbol{\Sigma}$, which in general will not coincide with the original axes of the $t_1, \dots, t_D$ variables. Of course, we can obtain a new rectangle $\mathcal{R}'$ aligned with the $t_1, \dots, t_D$ axes by taking intervals ranging from the minimal to the maximal corner of $\mathcal{R}$ in each direction, but obviously $P(\mathcal{R}') \geq P(\mathcal{R})$ and it is difficult to find $P(\mathcal{R}')$ exactly or to bound the error (see fig. 8.4b).

For the mixture with $\boldsymbol{\Sigma}_m = \sigma^2\mathbf{I}_D$, we have $\boldsymbol{\Lambda} - \sigma^2\mathbf{I}_D \geq 0$ always and so the error bars have a minimal length of $2\rho\sigma$ in each principal direction.

8.4.2 Approximation by a normal distribution near a mode of the mixture

If the mixture distribution is unimodal, then the best estimate of the mixture distribution using a normal distribution has the mean and the covariance equal to those of the mixture (eqs. (8.7) and (8.8), respectively). However, since we want the mode to be contained in the confidence interval, we estimate the normal mean with the mixture mode. This will be a good approximation unless the distribution is very skewed. Thus, the covariance $\boldsymbol{\Sigma}$ of the approximating normal is given by eq. (8.8).

If the mixture distribution is multimodal, we can use local information to obtain the covariance $\boldsymbol{\Sigma}$ of the approximating normal. In particular, at each mode of the mixture, the value of the Hessian contains information of how flat or how peaked the distribution $p(\mathbf{t})$ is around that mode. Consider a mode of the mixture at $\mathbf{t} = \boldsymbol{\nu}$ with Hessian $\mathbf{H}$. Since it is a maximum, $\mathbf{H} < 0$, and so we can write the singular value decomposition of $\mathbf{H}$ as $\mathbf{H} = -\mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^T$ where $\boldsymbol{\Lambda} > 0$. Since the Hessian of a normal distribution $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ at its mode is equal to $-\left|2\pi\boldsymbol{\Sigma}\right|^{-1/2}\boldsymbol{\Sigma}^{-1}$, from eq. (8.10), equating this to our known mixture Hessian $\mathbf{H}$ we obtain $\boldsymbol{\Sigma} = \mathbf{U}\mathbf{S}^{-1}\mathbf{U}^T$ where $\mathbf{S} = (2\pi)^{\frac{D}{D+2}}\left|\boldsymbol{\Lambda}\right|^{-\frac{1}{D+2}}\boldsymbol{\Lambda} = \left|2\pi\boldsymbol{\Lambda}^{-1}\right|^{\frac{1}{D+2}}\boldsymbol{\Lambda}$, as can be easily confirmed by substitution. So $\boldsymbol{\Sigma} = \left|2\pi(-\mathbf{H})^{-1}\right|^{-\frac{1}{D+2}}(-\mathbf{H})^{-1}$. Thus, we can approximate the mixture probability to second order in a neighbourhood near its mode $\boldsymbol{\nu}$ as

$$p(\mathbf{t}) \approx p(\boldsymbol{\nu}) + \frac{1}{2}(\mathbf{t} - \boldsymbol{\nu})^T\mathbf{H}(\mathbf{t} - \boldsymbol{\nu}) = p(\boldsymbol{\nu}) - \frac{1}{2}(\mathbf{t} - \boldsymbol{\nu})^T\left(\left|2\pi\boldsymbol{\Sigma}\right|^{-\frac{1}{2}}\boldsymbol{\Sigma}^{-1}\right)(\mathbf{t} - \boldsymbol{\nu}).$$

Using the log-density $L(\mathbf{t}) = \ln p(\mathbf{t})$, call $\mathbf{H}'$ the Hessian of L at a mode $\boldsymbol{\nu}$. The second-order approximation near the mode $\boldsymbol{\nu}$

$$L(\mathbf{t}) \approx L(\boldsymbol{\nu}) + \frac{1}{2}(\mathbf{t}-\boldsymbol{\nu})^T\mathbf{H}'(\mathbf{t}-\boldsymbol{\nu}) \Longrightarrow p(\mathbf{t}) = e^{L(\mathbf{t})} \approx p(\boldsymbol{\nu})e^{\frac{1}{2}(\mathbf{t}-\boldsymbol{\nu})^T\mathbf{H}'(\mathbf{t}-\boldsymbol{\nu})}$$

gives $\boldsymbol{\Sigma} = (-\mathbf{H}')^{-1}$. Note that, from eq. (8.14), $\mathbf{H}' = \frac{1}{p(\boldsymbol{\nu})}\mathbf{H}$.

8.4.3 Error bars at the mode of the mixture

From the previous discussion we can derive the following algorithm. Choose a confidence level $0 < P < 1$ and compute $\rho = \sqrt{2}\arg\operatorname{erf}(P^{1/D})$. Given a vector $\boldsymbol{\nu}$ and a negative definite matrix $\mathbf{H}$ representing a mode of the mixture and the Hessian at that mode, respectively, and calling $\boldsymbol{\Sigma}$ the covariance matrix of the mixture (from eq. (8.8)):

- If the mixture is unimodal, then (*mixture covariance method*) decompose $\boldsymbol{\Sigma} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^T$ with $\mathbf{U}$ orthogonal and $\boldsymbol{\Lambda} > 0$ diagonal. The D principal error bars are centred in $\boldsymbol{\nu}$, directed along the vectors $\mathbf{u}_1, \dots, \mathbf{u}_D$ and with lengths $2\rho\sqrt{\lambda_1}, \dots, 2\rho\sqrt{\lambda_D}$.
- If the mixture is multimodal, then (*Hessian method*):
  - If $\mathbf{H}$ is the Hessian of $p(\mathbf{t})$, then decompose $-\mathbf{H} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^T$ with $\mathbf{U}$ orthogonal and $\boldsymbol{\Lambda} > 0$ diagonal. Compute $\mathbf{S} = |2\pi\boldsymbol{\Lambda}^{-1}|^{\frac{1}{D+2}}\,\boldsymbol{\Lambda}$. The D principal error bars are centred in $\boldsymbol{\nu}$, directed along the vectors $\mathbf{u}_1, \dots, \mathbf{u}_D$ and with lengths $\frac{2\rho}{\sqrt{s_1}}, \dots, \frac{2\rho}{\sqrt{s_D}}$.
  - If $\mathbf{H}$ is the Hessian of $L(\mathbf{t}) = \ln p(\mathbf{t})$, then decompose $-\mathbf{H} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^T$ with $\mathbf{U}$ orthogonal and $\boldsymbol{\Lambda} > 0$ diagonal. The D principal error bars are centred in $\boldsymbol{\nu}$, directed along the vectors $\mathbf{u}_1, \dots, \mathbf{u}_D$ and with lengths $\frac{2\rho}{\sqrt{\lambda_1}}, \dots, \frac{2\rho}{\sqrt{\lambda_D}}$.

8.4.4 Discussion

Since we are only using second-order local information, namely the Hessian at the mode, the best we can do is to approximate the mixture quadratically, as we have shown. This approximation is only valid in a small neighbourhood around the mode, which can lead to poor estimates of the error bars, as fig. 8.6(left) shows. In this case, the one-dimensional mixture looks like a normal distribution but with its top flattened. Thus, the Hessian there is very small, which in turn gives a very large (co)variance $\boldsymbol{\Sigma}$ for the normal with the same Hessian (dashed line). In this particular case, one can find a better normal distribution giving more accurate error bars, for example the one in dotted line (which has the same variance as the mixture). But when the mixture is multimodal, finding a better normal estimate of the mixture would require a more complex procedure (even more so in higher dimensions). At any rate, a small Hessian indicates a flat top and some uncertainty in the mode.

Observe that, while the directions of the bars obtained from $L(\mathbf{t}) = \ln p(\mathbf{t})$ coincide with those from $p(\mathbf{t})$ always, the lengths are different in general (except when $p(\mathbf{t})$ is Gaussian). Figure 8.6 illustrates the point.

8.5 Quantifying the sparseness of a Gaussian mixture

Besides finding the modes, one may also be interested in knowing whether the density is sparse—sharply peaked around the modes, with most of the probability mass concentrated around a small region around each mode—or whether its global aspect is flat. As we saw in section 7.3, if the Gaussian mixture under consideration represents a conditional distribution, a sparse distribution would correspond to a functional relationship (perhaps multivalued) while a flat distribution would correspond to independence (fig. 8.7). Of course, these are just the two extremes of a continuous spectrum.

While the error bars locally characterise the peak widths, we can globally characterise the degree of sparseness of a distribution $p(\mathbf{t})$ by its differential entropy[4] $h(p) \stackrel{\text{def}}{=} \mathrm{E}\{-\ln p\}$: high entropy corresponds to flat distributions, where the variable $\mathbf{t}$ can assume practically any value in its domain (fig. 8.7, right); and low entropy corresponds to peaked distributions, where $\mathbf{t}$ can only assume a finite set of values (fig. 8.7, left). This differential entropy value should be compared to the differential entropy of a reference distribution, e.g. a Gaussian distribution of the same covariance or a uniform distribution on the same range as the inputs.

[4]See section 7.12.4 for reasons why the entropy is not a completely satisfactory measure of sparseness.

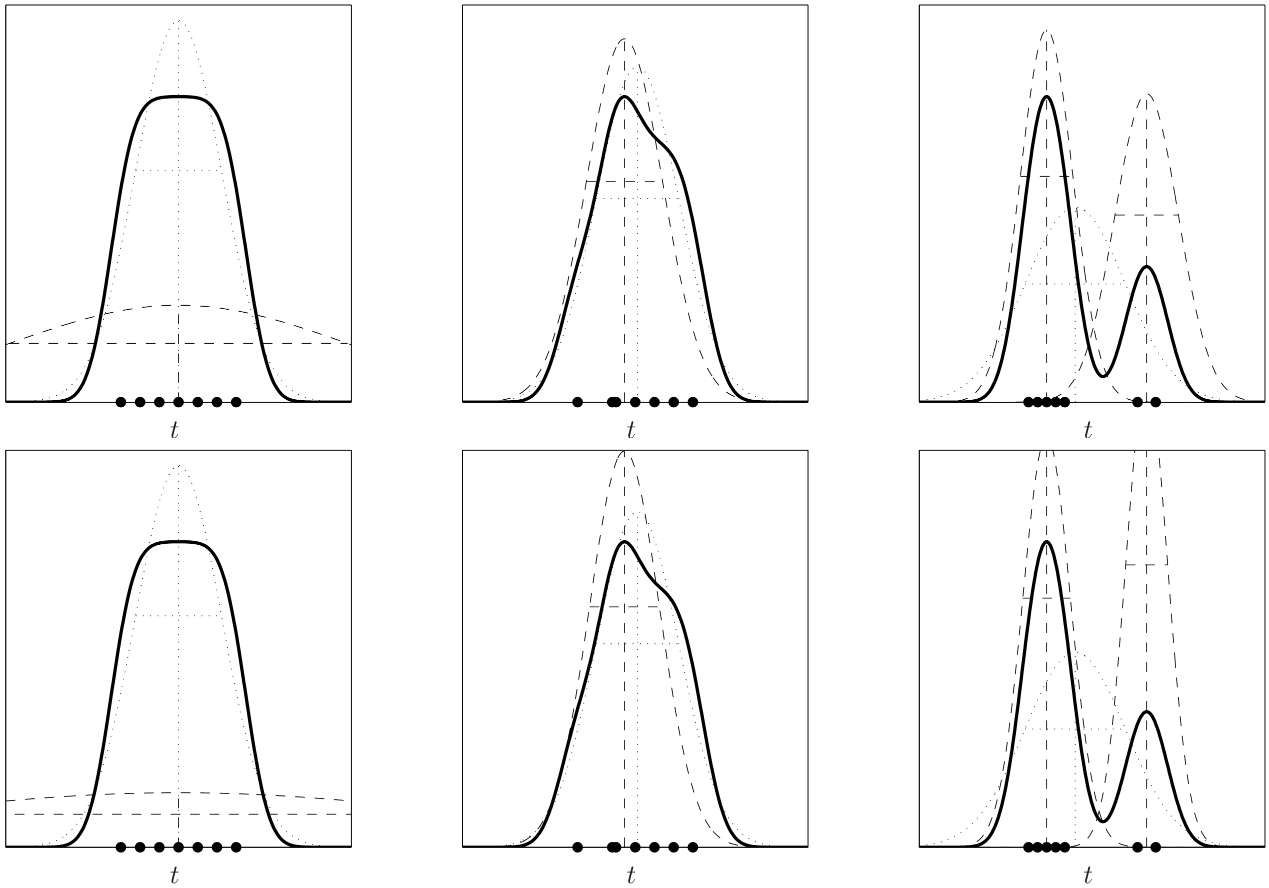


Figure 8.6: Error bars for a one-dimensional mixture of Gaussian distributions. The top graphs correspond to the bars obtained from $p(t)$ and the bottom ones to those obtained from $L(t) = \ln p(t)$. In each graph, the line codes are as follows. Thick solid line: the mixture distribution $p(t) = \sum_{m=1}^{M} \frac{1}{M}(2\pi\sigma^2)^{-\frac{1}{2}} e^{-\frac{1}{2}\left(\frac{t-\mu_m}{\sigma}\right)^2}$ with $M = 7$ components, where $\sigma = 1$ and the component centroids $\{\mu_m\}_{m=1}^{M}$ are marked on the horizontal axis. The dashed lines indicate the approximating normals using the Hessian method and the dotted ones the approximating normals using the mixture covariance method (i.e., using eq. (8.8)). In both cases, the vertical line(s) indicate the location of the mode(s) or mixture mean, respectively, and the horizontal one the error bars for a confidence of 68%, i.e., the interval is two standard deviations long. On the left graph, the Hessian gives too broad an approximation because the mixture top is very flat. On the centre graph, both methods give a similar result. On the right graph, the mixture covariance method breaks down due to the bimodality of the mixture. Observe how the bars obtained from $p(x)$ do not coincide with those from $\ln p(t)$, being sometimes narrower and sometimes wider.

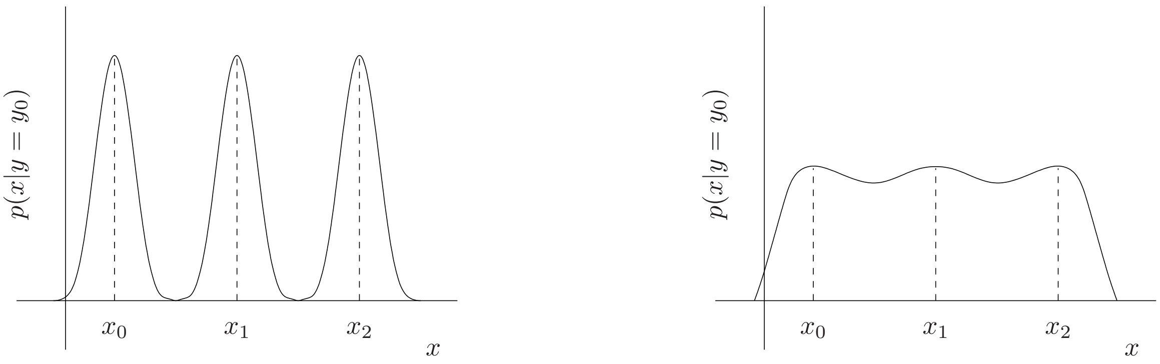


Figure 8.7: Shape of the conditional distribution of variable x given value y_0 of variable y. *Left*: sparse (multiply peaked), low entropy; x is almost functionally dependent on y for $y = y_0$, with $f(y_0) = x_0$ or x_1 or x_3. *Right*: flat, high entropy; x is almost independent of y for $y = y_0$.

There are no analytical expressions for the entropy of a Gaussian mixture, but in section 8.7.6 we derive the following upper (UB_1) and lower bounds (LB_1, LB_2):

$$\mathsf{LB}_1 \stackrel{\text{def}}{=} \frac{1}{2}\ln\left\{(2\pi e)^D \prod_{m=1}^{M} |\boldsymbol{\Sigma}_m|^{\pi_m}\right\}$$

$$\mathsf{LB}_2 \stackrel{\text{def}}{=} -\ln\left\{\sum_{m,n=1}^{M} p(m)p(n)\,|2\pi(\boldsymbol{\Sigma}_m+\boldsymbol{\Sigma}_n)|^{-\frac{1}{2}}\, e^{-\frac{1}{2}(\boldsymbol{\mu}_m-\boldsymbol{\mu}_n)^T(\boldsymbol{\Sigma}_m+\boldsymbol{\Sigma}_n)^{-1}(\boldsymbol{\mu}_m-\boldsymbol{\mu}_n)}\right\}$$

$$\mathsf{UB}_1 \stackrel{\text{def}}{=} \frac{1}{2}\ln\left((2\pi e)^D\,|\boldsymbol{\Sigma}|\right).$$

where $\boldsymbol{\Sigma}$ is the covariance matrix of the mixture, given in eq. (8.8). Other approximations to the differential entropy of an arbitrary continuous density, that have been often used in the context of projection pursuit and independent component analysis, were mentioned in section 4.6.1.

8.6 Conclusions

We have presented algorithms to find all the modes of a given Gaussian mixture (satisfying certain conditions; see below) based on the intuitive conjecture that (1) the number of modes is upper bounded by the number of components and (2) the modes are contained in the convex hull of the component centroids. While our proof of part (1) of this conjecture, given in section 8.7.2, is only partial (for a mixture of two components of arbitrary dimension and equal covariance), no counterexample has been found in a number of simulations. The only other related works we are aware of (Behboodian, 1970; Konstantellos, 1980) also provided partial proofs under various restricted conditions. All algorithms have been extensively tested for the case of spherical, equal component covariances in simulated mixtures, in the inverse kinematics problem for the robot arm of section 9.3 and in the missing data reconstruction experiments of section 10.1 (a speech inverse problem, the acoustic-to-articulatory mapping), which require finding all the modes of a Gaussian mixture of about 1 000 components in over 60 dimensions for every speech frame in an utterance.

The conjecture *does not hold* in general when the mixture is multivariate and its components have covariance matrices that are not isotropic and are different from each other (see section 8.7.2). In this case, the mixture may have more modes than components and the modes may lie outside the convex hull of the centroids. In fact, when the covariances are elongated, extra modes can appear in the tails of the components, as fig. 8.9 shows. This makes the problem of exhaustive mode finding more difficult for this type of Gaussian mixtures, although our algorithms can still be applied to find some of the modes—and some more could be found by starting the algorithms from points other than the centroids, e.g. points in the components' tails. The problem remains open, then, for arbitrary Gaussian mixtures: bounds for the number of modes, characterisation of a region where they lie and algorithms to find all them. However, the cases for which the conjecture seems to hold are practically important, especially that of isotropic components. This is the case for (nonadaptive) kernel density estimation and for GTM, which is the model we use in chapters 7, 9 and 10. And it should also hold for the diagonal GTM model we introduced in section 2.12.4, for which the component covariance matrices are diagonal but equal. Finally, while arbitrary Gaussian mixtures can potentially have more modes than components, this may be the exception rather than the rule when Gaussian mixtures are used to approximate low-dimensional manifolds, since crisscrossing of highly elongated components is unlikely[5].

Given the current interest in the machine learning and computer vision literature in probabilistic models able to represent multimodal distributions (specially Gaussian mixtures), these algorithms could be of benefit in a number of applications or as part of other algorithms. Specifically, they could be applied to clustering and regression problems. An example of clustering application is the determination of subclustering within galaxy systems from the measured position (right ascension and declination) and redshifts of individual galaxies (Pisani, 1993). The density of the position-velocity distribution is often modelled as a Gaussian mixture (whether parametrically or nonparametrically via kernel estimation) whose modes correspond in principle to gravitationally bound galactic structures. An example of regression application is the representation of multivalued mappings (which are often the result of inverting a forward mapping) with a Gaussian mixture, which we consider in chapter 7. In this approach, all variables (inputs $\mathbf{x}$ and outputs $\mathbf{y}$ of a mapping) are jointly modelled by a Gaussian mixture and the mapping $\mathbf{x} \to \mathbf{y}$ is defined as the modes of the conditional

[5]Such crisscrossing of highly elongated components is more likely in the product-of-experts model (see fig. 1 in Hinton, 1999), where individual components are combined multiplicatively rather than additively.

distribution $p(\mathbf{y}|\mathbf{x})$, itself a Gaussian mixture. Scrutiny of the posterior modes in Bayesian methods is another possible use.

The algorithms described here can be easily adapted to find minima of the mixture rather than maxima. However one must constrain them to search only for proper minima and avoid following the improper minima at $p(\mathbf{t}) \to 0$ when $\|\mathbf{t}\| \to \infty$. Also, it should be possible to derive similar algorithms for other mixtures of bump-like distributions (localised and monotonically decreasing, but not necessarily Gaussian).

Our Matlab implementation of the mode-finding algorithms is freely available in the Internet (see appendix C).

8.7 Mathematical appendix: some results about Gaussian mixtures

Here we proof several results about finite mixtures of normal distributions mentioned in this chapter. Let us first define them and their moments.

Consider a mixture distribution (Titterington et al., 1985) of $M > 1$ components in $\mathbb{R}^D$ for $D \geq 1$:

$$p(\mathbf{t}) \stackrel{\text{def}}{=} \sum_{m=1}^{M} p(m)p(\mathbf{t}|m) \stackrel{\text{def}}{=} \sum_{m=1}^{M} \pi_m p(\mathbf{t}|m) \qquad \forall \mathbf{t} \in \mathbb{R}^D \tag{8.6}$$

where $\sum_{m=1}^{M} \pi_m = 1$, $\pi_m \in (0,1)$ $\forall m = 1, \dots, M$ and each component distribution is a normal probability distribution in $\mathbb{R}^D$. So $\mathbf{t}|m \sim \mathcal{N}_D(\boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m)$, where $\boldsymbol{\mu}_m \stackrel{\text{def}}{=} \mathrm{E}_{p(\mathbf{t}|m)}\{\mathbf{t}\}$ and $\boldsymbol{\Sigma}_m \stackrel{\text{def}}{=} \mathrm{E}_{p(\mathbf{t}|m)}\left\{(\mathbf{t}-\boldsymbol{\mu}_m)(\mathbf{t}-\boldsymbol{\mu}_m)^T\right\} > 0$ are the mean vector and covariance matrix, respectively, of component m. We write $p(\mathbf{t})$ and not $p(\mathbf{t}|\{\pi_m, \boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m\}_{m=1}^M)$ because we assume that the parameters $\{\pi_m, \boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m\}_{m=1}^M$ have been estimated previously and their values fixed. Then:

- The **mixture mean** is:

$$\boldsymbol{\mu} \stackrel{\text{def}}{=} \mathrm{E}_{p(\mathbf{t})}\{\mathbf{t}\} = \sum_{m=1}^{M} \pi_m \boldsymbol{\mu}_m. \tag{8.7}$$

- The **mixture covariance** is:

$$\begin{aligned} \boldsymbol{\Sigma} &\stackrel{\text{def}}{=} \mathrm{E}_{p(\mathbf{t})}\left\{(\mathbf{t}-\boldsymbol{\mu})(\mathbf{t}-\boldsymbol{\mu})^T\right\} = \mathrm{E}_{p(\mathbf{t})}\left\{\mathbf{t}\mathbf{t}^T\right\} - \boldsymbol{\mu}\boldsymbol{\mu}^T = \sum_{m=1}^{M} \pi_m \,\mathrm{E}_{p(\mathbf{t}|m)}\left\{\mathbf{t}\mathbf{t}^T\right\} - \boldsymbol{\mu}\boldsymbol{\mu}^T \\ &= \sum_{m=1}^{M} \pi_m \boldsymbol{\Sigma}_m + \sum_{m=1}^{M} \pi_m \boldsymbol{\mu}_m \boldsymbol{\mu}_m^T - \boldsymbol{\mu}\boldsymbol{\mu}^T = \sum_{m=1}^{M} \pi_m \left(\boldsymbol{\Sigma}_m + (\boldsymbol{\mu}_m - \boldsymbol{\mu})(\boldsymbol{\mu}_m - \boldsymbol{\mu})^T\right). \end{aligned} \tag{8.8}$$

These results are valid for any mixture, not necessarily of Gaussian distributions.

8.7.1 Gradient and Hessian with respect to the independent random variables

Here we obtain the gradient and the Hessian of the density function p with respect to the independent variables $\mathbf{t}$ (not with respect to the parameters $\{\pi_m, \boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m\}_{m=1}^M$). Firstly let us derive the gradient and Hessian for a D-variate normal distribution. Let $\mathbf{t} \sim \mathcal{N}_D(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. Then:

$$p(\mathbf{t}) = |2\pi\boldsymbol{\Sigma}|^{-\frac{1}{2}} \, e^{-\frac{1}{2}(\mathbf{t}-\boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1}(\mathbf{t}-\boldsymbol{\mu})}$$

which is differentiable and nonnegative for all $\mathbf{t} \in \mathbb{R}^D$. Let $\boldsymbol{\Sigma} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^T$ be the singular value decomposition of $\boldsymbol{\Sigma}$, so that $\mathbf{U}$ is orthogonal and $\boldsymbol{\Lambda} > 0$ diagonal. Calling $\mathbf{z} = \mathbf{U}^T(\mathbf{t}-\boldsymbol{\mu})$:

$$\begin{aligned} \frac{\partial p}{\partial t_d} &= p(\mathbf{t}) \frac{\partial}{\partial t_d}\left(-\frac{1}{2}(\mathbf{t}-\boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1}(\mathbf{t}-\boldsymbol{\mu})\right) = p(\mathbf{t}) \frac{\partial}{\partial t_d}\left(-\frac{1}{2}(\mathbf{z}^T \boldsymbol{\Lambda}^{-1}\mathbf{z})\right) \\ &= p(\mathbf{t}) \sum_{e=1}^{D} \frac{\partial}{\partial z_e}\left(-\frac{1}{2}\sum_{f=1}^{D} \frac{z_f^2}{\lambda_f}\right) \frac{\partial z_e}{\partial t_d} = p(\mathbf{t}) \sum_{e=1}^{D} -\frac{z_e}{\lambda_e} u_{de}, \end{aligned}$$

since $\partial z_e / \partial t_d = u_{de}$. The result above is the dth element of vector $-p(\mathbf{t})\mathbf{U}\boldsymbol{\Lambda}^{-1}\mathbf{z}$ and so the gradient is[6]:

$$\mathbf{g} \stackrel{\text{def}}{=} \nabla p(\mathbf{t}) = p(\mathbf{t})\boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu} - \mathbf{t}). \tag{8.9}$$

[6]For clarity of notation, we omit the dependence on $\mathbf{t}$ of both the gradient and the Hessian, writing $\mathbf{g}$ and $\mathbf{H}$ where we should write $\mathbf{g}(\mathbf{t})$ and $\mathbf{H}(\mathbf{t})$.

Taking the second derivatives:

$$\begin{aligned}\frac{\partial}{\partial t_c}\left(\frac{\partial p}{\partial t_d}\right) &= \frac{\partial p}{\partial t_c}\sum_{e=1}^{D} -\frac{z_e}{\lambda_e}u_{de} + p(\mathbf{t})\sum_{e=1}^{D} -\frac{u_{de}}{\lambda_e}\frac{\partial z_e}{\partial t_c} \\ &= p(\mathbf{t})\left(\sum_{e=1}^{D} -\frac{z_e}{\lambda_e}u_{ce}\right)\left(\sum_{e=1}^{D} -\frac{z_e}{\lambda_e}u_{de}\right) + p(\mathbf{t})\sum_{e=1}^{D} -\frac{u_{de}}{\lambda_e}u_{ce},\end{aligned}$$

which is the (c,d)th element of matrix $\frac{\mathbf{g}\mathbf{g}^T}{p(\mathbf{t})} - p(\mathbf{t})\mathbf{U}\boldsymbol{\Lambda}^{-1}\mathbf{U}^T$. So the Hessian is:

$$\mathbf{H} \stackrel{\text{def}}{=} (\nabla\nabla^T)p(\mathbf{t}) = \frac{\mathbf{g}\mathbf{g}^T}{p(\mathbf{t})} - p(\mathbf{t})\boldsymbol{\Sigma}^{-1} = p(\mathbf{t})\boldsymbol{\Sigma}^{-1}\left((\boldsymbol{\mu}-\mathbf{t})(\boldsymbol{\mu}-\mathbf{t})^T - \boldsymbol{\Sigma}\right)\boldsymbol{\Sigma}^{-1}. \tag{8.10}$$

It is clear that the gradient is zero at $\mathbf{t} = \boldsymbol{\mu}$ only and there $\mathbf{H} = -\left|2\pi\boldsymbol{\Sigma}\right|^{-1/2}\boldsymbol{\Sigma}^{-1} < 0$, thus being a maximum.

Consider now a finite mixture of D-variate normal distributions $p(\mathbf{t}) = \sum_{m=1}^{M} p(m)p(\mathbf{t}|m)$ where $\mathbf{t}|m \sim \mathcal{N}_D(\boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m)$. By the linearity of the differential operator and defining:

$$\begin{aligned}\mathbf{g}_m &\stackrel{\text{def}}{=} \nabla p(\mathbf{t}|m) = p(\mathbf{t}|m)\boldsymbol{\Sigma}_m^{-1}(\boldsymbol{\mu}_m - \mathbf{t}) \\ \mathbf{H}_m &\stackrel{\text{def}}{=} (\nabla\nabla^T)p(\mathbf{t}|m) = \frac{\mathbf{g}_m\mathbf{g}_m^T}{p(\mathbf{t}|m)} - p(\mathbf{t}|m)\boldsymbol{\Sigma}_m^{-1} = p(\mathbf{t}|m)\boldsymbol{\Sigma}_m^{-1}\left((\boldsymbol{\mu}_m-\mathbf{t})(\boldsymbol{\mu}_m-\mathbf{t})^T - \boldsymbol{\Sigma}_m\right)\boldsymbol{\Sigma}_m^{-1}\end{aligned}$$

we obtain:

$$\text{Gradient } \mathbf{g} \stackrel{\text{def}}{=} \nabla p(\mathbf{t}) = \sum_{m=1}^{M} p(m)\mathbf{g}_m = \sum_{m=1}^{M} p(\mathbf{t},m)\boldsymbol{\Sigma}_m^{-1}(\boldsymbol{\mu}_m - \mathbf{t}) \tag{8.11}$$

$$\text{Hessian } \mathbf{H} \stackrel{\text{def}}{=} (\nabla\nabla^T)p(\mathbf{t}) = \sum_{m=1}^{M} p(m)\mathbf{H}_m = \sum_{m=1}^{M} p(\mathbf{t},m)\boldsymbol{\Sigma}_m^{-1}\left((\boldsymbol{\mu}_m-\mathbf{t})(\boldsymbol{\mu}_m-\mathbf{t})^T - \boldsymbol{\Sigma}_m\right)\boldsymbol{\Sigma}_m^{-1}. \tag{8.12}$$

8.7.1.1 Gradient and Hessian of the log-density

Call $L(\mathbf{t}) \stackrel{\text{def}}{=} \ln p(\mathbf{t})$. Then, the gradient and Hessian of L are related to those of p as follows:

$$\text{Gradient } \nabla L(\mathbf{t}) = \frac{1}{p}\mathbf{g} \tag{8.13}$$

$$\text{Hessian } (\nabla\nabla^T)L(\mathbf{t}) = -\frac{1}{p^2}\mathbf{g}\mathbf{g}^T + \frac{1}{p}\mathbf{H}. \tag{8.14}$$

Note from eq. (8.14) that, if the Hessian $\mathbf{H}$ of p is definite negative, then the Hessian of L is also definite negative, since $-\frac{1}{p^2}\mathbf{g}\mathbf{g}^T$ is either a null matrix (at stationary points) or negative definite (everywhere else).

In this work, we will always implicitly refer to the gradient $\mathbf{g}$ or Hessian $\mathbf{H}$ of p, eqs. (8.11) and (8.12), rather than those of L, eqs. (8.13) and (8.14), unless otherwise noted.

8.7.2 Partial proof of conjecture 8.1

Let us recall that the convex hull of the vectors $\{\boldsymbol{\mu}_m\}_{m=1}^{M}$ is defined as the set

$$\left\{\mathbf{t}:\ \mathbf{t} = \sum_{m=1}^{M} \lambda_m\boldsymbol{\mu}_m \text{ with } \{\lambda_m\}_{m=1}^{M} \subset [0,1] \text{ and } \sum_{m=1}^{M}\lambda_m = 1\right\}.$$

Conjecture 8.1. *Let $p(\mathbf{t}) = \sum_{m=1}^{M} p(m)p(\mathbf{t}|m)$, where $\mathbf{t}|m \sim \mathcal{N}_D(\boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m)$, be a mixture of M D-variate normal distributions. Then $p(\mathbf{t})$ has M modes at most, all of which are in the convex hull of $\{\boldsymbol{\mu}_m\}_{m=1}^{M}$, if one of the following conditions holds:*

1. $D = 1$ (one-dimensional mixture)
2. *$D \geq 1$ and the covariance matrices are arbitrary but equal:* $\boldsymbol{\Sigma}_m = \boldsymbol{\Sigma}\ \forall m = 1,\dots,M$ (homoscedastic mixture).

3. $D \geq 1$ and the covariance matrices are isotropic: $\mathbf{\Sigma}_m = \sigma_m^2 \mathbf{I}_D$ (spherical or isotropic mixture).

I have not been able to prove this conjecture in general. In the rest of this section we give: a partial proof that the number of modes is less or equal than the number of centroids; a complete proof that the modes lie in the convex hull of the centroids; and an example that shows that none of the two statements need hold for arbitrary-covariance mixtures.

We first prove that (a) for $\mathbf{\Sigma}_m = \mathbf{\Sigma}\ \forall m = 1, \dots, M$, the stationary points of the gradient of $p(\mathbf{t})$ (maxima, minima and saddle points of $p(\mathbf{t})$) lie in the convex hull of the centroids; and that (b) for $M = 2$ and $\mathbf{\Sigma}_1 = \mathbf{\Sigma}_2 = \mathbf{\Sigma}$, there are at most two maxima, which lie in the convex hull of $\boldsymbol{\mu}_1$ and $\boldsymbol{\mu}_2$.

Proof. Let us prove (a). Assume without loss of generality that the centroids are all different. For $\{\lambda_m\}_{m=1}^M \subset \mathbb{R}$ and $\sum_{m=1}^M \lambda_m = 1$, the set of the points

$$\mathbf{t} = \sum_{m=1}^{M} \lambda_m \boldsymbol{\mu}_m = \sum_{m=1}^{M-1} \lambda_m \boldsymbol{\mu}_m + \left(1 - \sum_{m=1}^{M-1} \lambda_m\right) \boldsymbol{\mu}_M = \boldsymbol{\mu}_M + \sum_{m=1}^{M-1} \lambda_m (\boldsymbol{\mu}_m - \boldsymbol{\mu}_M)$$

is the minimal linear manifold containing $\{\boldsymbol{\mu}_m\}_{m=1}^M$, i.e., the hyperplane passing through all the centroids. This is not necessarily a vector subspace, because it may not contain the zero vector. However, the set

$$\left\{\mathbf{y} : \mathbf{y} = \sum_{m=1}^{M-1} \lambda_m (\boldsymbol{\mu}_m - \boldsymbol{\mu}_M) \text{ with } \{\lambda_m\}_{m=1}^{M-1} \subset \mathbb{R}\right\}$$

is a vector subspace, namely the one spanned by $\{\boldsymbol{\mu}_m - \boldsymbol{\mu}_M\}_{m=1}^{M-1}$. Then, an arbitrary point $\mathbf{t} \in \mathbb{R}^D$ can be decomposed as $\mathbf{t} = \sum_{m=1}^M \lambda_m \boldsymbol{\mu}_m + \mathbf{w}$, where $\{\lambda_m\}_{m=1}^M \subset \mathbb{R}$ with $\sum_{m=1}^M \lambda_m = 1$ and $\mathbf{w}$ is a vector orthogonal to that manifold, i.e., orthogonal to $\{\boldsymbol{\mu}_m - \boldsymbol{\mu}_M\}_{m=1}^{M-1}$. Let us now compute the zero-gradient points of $p(\mathbf{t})$ from eq. (8.11):

$$\begin{aligned}
\mathbf{g}(\mathbf{t}) &= p(\mathbf{t}) \sum_{m=1}^{M} p(m|\mathbf{t}) \mathbf{\Sigma}_m^{-1} (\boldsymbol{\mu}_m - \mathbf{t}) \\
&= p(\mathbf{t}) \left(\sum_{m=1}^{M} p(m|\mathbf{t}) \mathbf{\Sigma}_m^{-1} \left(\boldsymbol{\mu}_m - \sum_{n=1}^{M} \lambda_n \boldsymbol{\mu}_n \right) - \sum_{m=1}^{M} p(m|\mathbf{t}) \mathbf{\Sigma}_m^{-1} \mathbf{w} \right) = \mathbf{0}.
\end{aligned}$$

For $\mathbf{\Sigma}_m = \mathbf{\Sigma}\ \forall m = 1, \dots, M$ this becomes

$$\mathbf{g}(\mathbf{t}) = p(\mathbf{t}) \mathbf{\Sigma}^{-1} \left(\sum_{m=1}^{M} p(m|\mathbf{t}) \left(\boldsymbol{\mu}_m - \sum_{n=1}^{M} \lambda_n \boldsymbol{\mu}_n \right) - \mathbf{w} \right) = \mathbf{0} \Longrightarrow \mathbf{v} - \mathbf{w} = \mathbf{0}$$

where $\mathbf{v} = \sum_{m=1}^M p(m|\mathbf{t}) \left(\boldsymbol{\mu}_m - \sum_{n=1}^M \lambda_n \boldsymbol{\mu}_n \right)$ clearly lies in the linear manifold spanned by $\{\boldsymbol{\mu}_m\}_{m=1}^M$ and therefore is orthogonal to $\mathbf{w}$. So $\mathbf{v} = \mathbf{w} = 0$. This proves that $\mathbf{t} = \sum_{m=1}^M \lambda_m \boldsymbol{\mu}_m$, i.e., all stationary points must lie in the the linear manifold spanned by the centroids.

Now let us prove that (b) for $M = 2$ and $\mathbf{\Sigma}_1 = \mathbf{\Sigma}_2 = \mathbf{\Sigma}$, the gradient becomes null only in one, two or three points, which lie in the convex hull of $\boldsymbol{\mu}_1$ and $\boldsymbol{\mu}_2$.

$$\begin{aligned}
\mathbf{v} &= \sum_{m=1}^{M} p(m|\mathbf{t}) \left(\boldsymbol{\mu}_m - \sum_{n=1}^{M} \lambda_n \boldsymbol{\mu}_n \right) = \sum_{m=1}^{M} p(m|\mathbf{t}) \boldsymbol{\mu}_m - \sum_{n=1}^{M} \lambda_n \boldsymbol{\mu}_n \\
&= \sum_{m=1}^{M} (p(m|\mathbf{t}) - \lambda_m) \boldsymbol{\mu}_m = \sum_{m=1}^{M-1} (p(m|\mathbf{t}) - \lambda_m)(\boldsymbol{\mu}_m - \boldsymbol{\mu}_M) = \mathbf{0}.
\end{aligned}$$

is a nonlinear system of D equations with unknowns $\lambda_1, \dots, \lambda_{M-1}$, very difficult to study in general. For $M = 2$, call $\lambda = \lambda_1$, so that $\lambda_2 = 1 - \lambda$, and $\pi = p(1)$, so that $p(2) = 1 - \pi$. Using Bayes' theorem we get:

$$(p(1|\mathbf{t}) - \lambda)(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2) = \mathbf{0} \Longrightarrow \lambda = p(1|\mathbf{t}) = \frac{p(1) p(\mathbf{t}|1)}{p(1) p(\mathbf{t}|1) + p(2) p(\mathbf{t}|2)}$$

which reduces to the transcendental equation

$$\lambda = \frac{1}{1 + e^{-\alpha(\lambda - \lambda_0)}} \qquad \alpha = (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)^T \mathbf{\Sigma}^{-1} (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2) \in (0, \infty) \qquad \lambda_0 = \frac{1}{2} + \ln \frac{1 - \pi}{\pi} \in (-\infty, \infty). \tag{8.15}$$

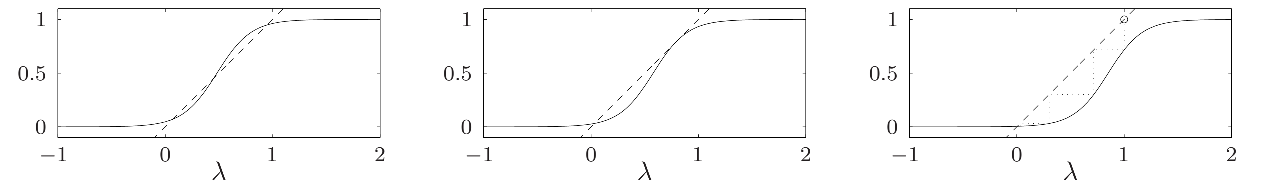


Figure 8.8: Three possible cases for the solutions of the equation $\lambda = f(\lambda)$, where $f(\lambda) = \frac{1}{1+e^{-\alpha(\lambda-\lambda_0)}}$. The solid line corresponds to $f(\lambda)$ and the dashed one to λ. The right figure also shows in dotted line the sequence of fixed-point iterations starting from $\lambda = 1$ (marked "$\circ$"), converging to a fixed point slightly larger than 0.

It is easy to see geometrically (see fig. 8.8) that this equation can only have one, two or three roots in $(0, 1)$, which proves that the stationary points $\mathbf{t} = \lambda\boldsymbol{\mu}_1 + (1 - \lambda)\boldsymbol{\mu}_2$ of p lie in the convex hull of $\{\boldsymbol{\mu}_1, \boldsymbol{\mu}_2\}$. Further, since for $M = 2$ the convex hull is a line segment, in the case with three stationary points one of them cannot be a maximum, and so the number of modes is $M = 2$ at the most.

In fact, solving eq. (8.15) gives these stationary points (e.g. by fixed-point iteration, see fig. 8.8 right), and using eq. (8.12) to compute the Hessian will determine whether they are a maximum, a minimum or a saddle-point. $\square$

Remark. Related results have been proven, in a different way, in the literature. Behboodian (1970) shows that for $M = 2$ and $D = 1$, with no restriction on $\boldsymbol{\Sigma}_m$, $p(t)$ has one, two or three stationary points which all lie in the convex hull of $\{\mu_1, \mu_2\}$. Konstantellos (1980) gives a necessary condition for unimodality for two cases: $M = 2$, $\pi_1 = \pi_2$, $\boldsymbol{\Sigma}_1 = \boldsymbol{\Sigma}_2$ and $D > 1$; and $M = 2$ and $D = 2$, with no restriction on π_m, $\boldsymbol{\Sigma}_m$.

Remark. Although the previous proof makes use of formula (8.11), which is only valid for normal components, intuitively there is nothing special about the normal distribution here. In fact, the conjecture should hold for components of other functional forms (not necessarily positive), as long as they are bounded, piecewise continuous and have a unique maximum and no minima (*bump* functions). In some cases an infinite number of stationary points may exist (as is easily seen for e.g. triangular bumps).

Remark. Alexander Heimel, from the dept. of Mathematics at King's College London, has sent me the following proof that all modes are in the convex hull of the centroids for the case of a Gaussian mixture of spherical components (of not necessarily equal variances). This result is more general than part (a) of our proof above, which only holds for spherical covariances which are all equal.

Theorem 8.7.1 (Heimel, 2000, pers. comm.). *Let $\{f_m\}_{m=1}^M$ be a set of M functions from $\mathbb{R}^D$ to $\mathbb{R}$ where each function can be written as $f_m(\mathbf{t}) = g_m(\|\mathbf{t} - \boldsymbol{\mu}_m\|)$ for strictly monotonically decreasing functions $\{g_m\}_{m=1}^M$ and points $\{\boldsymbol{\mu}_m\}_{m=1}^M$. Then all the maxima of the function $F \stackrel{\text{def}}{=} \sum_{m=1}^M f_m$ are inside the convex hull of $\{\boldsymbol{\mu}_m\}_{m=1}^M$.*

Proof. By contradiction. Call $\mathcal{H}$ the convex hull of $\{\boldsymbol{\mu}_m\}_{m=1}^M$. Suppose F takes a maximum at $\mathbf{t} \notin \mathcal{H}$. Call $\mathbf{u}$ the closest point in $\mathcal{H}$ to $\mathbf{t}$. Then, it is easy to see that any point $\mathbf{t}'$ in the segment between $\mathbf{t}$ and $\mathbf{u}$ is closer to all points of $\mathcal{H}$ than $\mathbf{t}$ is. Thus $f_m(\mathbf{t}') > f_m(\mathbf{t})\ \forall m = 1, \dots, M$ and so $F(\mathbf{t}') > F(\mathbf{t})$. Since in every neighbourhood of $\mathbf{t}$ there are points for which F is larger, $\mathbf{t}$ cannot be a maximum. $\square$

The following theorem shows the equivalence of the modes' problem for homoscedastic mixtures (where the components have the same covariance). Thus, one can try to prove a result for the simple case of isotropic, unit covariances ($\boldsymbol{\Sigma}_m = \mathbf{I}_D$) and then the result will also hold for $\boldsymbol{\Sigma}_m = \boldsymbol{\Sigma}$ arbitrary.

Theorem 8.7.2. *The mixtures $p(\mathbf{t}) = \sum_{m=1}^M \pi_m \left|2\pi\boldsymbol{\Sigma}\right|^{-\frac{1}{2}} e^{-\frac{1}{2}(\mathbf{t}-\boldsymbol{\mu}_m)^T\boldsymbol{\Sigma}^{-1}(\mathbf{t}-\boldsymbol{\mu}_m)}$ (arbitrary but equal covariances) and $p(\mathbf{u}) = \sum_{m=1}^M \pi_m (2\pi)^{-\frac{D}{2}} e^{-\frac{1}{2}\|\mathbf{u}-\boldsymbol{\nu}_m\|^2}$ (unit covariances), related by a rotation and scaling, have the same number of modes, which lie in the respective centroid convex hulls.*

Proof. Let $\boldsymbol{\Sigma}^{-1} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^T$ be the spectral decomposition of $\boldsymbol{\Sigma}^{-1}$, with $\mathbf{U}$ orthogonal and $\boldsymbol{\Lambda}$ diagonal and positive definite. Consider the transformation $\mathbf{u} \stackrel{\text{def}}{=} \boldsymbol{\Lambda}^{\frac{1}{2}}\mathbf{U}^T\mathbf{t}$ (orthogonal rotation followed by scaling), with Jacobian $\mathbf{J} = \left|\frac{d\mathbf{u}}{d\mathbf{t}}\right| = |\boldsymbol{\Lambda}|^{\frac{1}{2}}$, and define $\boldsymbol{\nu}_m \stackrel{\text{def}}{=} \boldsymbol{\Lambda}^{\frac{1}{2}}\mathbf{U}^T\boldsymbol{\mu}_m$. The density of $\mathbf{u}$ is then $p(\mathbf{u}) = p(\mathbf{t})\mathbf{J}^{-1} = \sum_{m=1}^M \pi_m (2\pi)^{-\frac{D}{2}} e^{-\frac{1}{2}\|\mathbf{u}-\boldsymbol{\nu}_m\|^2}$ (section A.5).

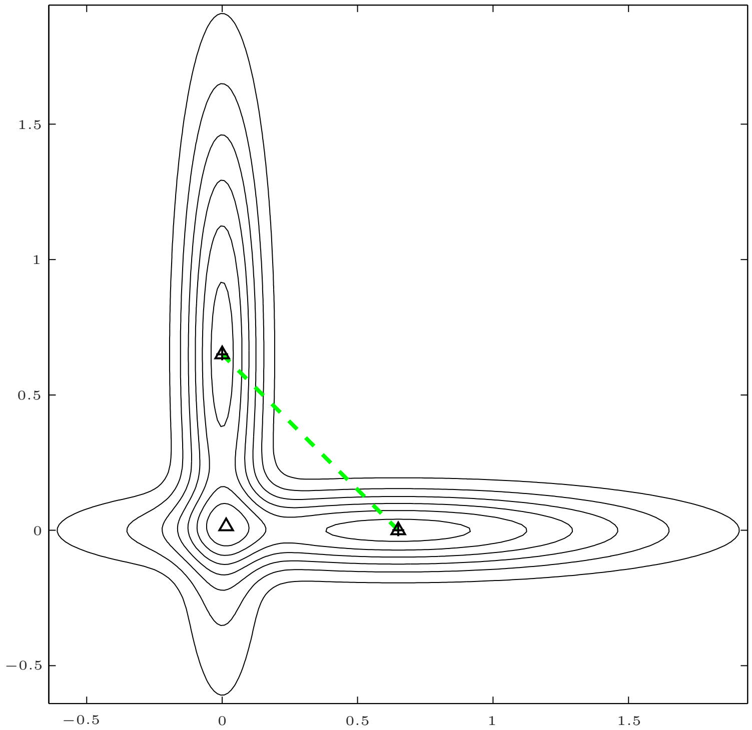


Figure 8.9: Mixtures in dimension $D \geq 2$ that have different, non-isotropic covariances do not generally verify conjecture 8.1. The graph shows a contour plot for the bicomponent mixture $p(\mathbf{t}) = \sum_{m=1}^{2} \frac{1}{2} |2\pi\mathbf{\Sigma}_m|^{-\frac{1}{2}} e^{-\frac{1}{2}(\mathbf{t}-\boldsymbol{\mu}_m)^T \mathbf{\Sigma}_m^{-1}(\mathbf{t}-\boldsymbol{\mu}_m)}$ with $\pi_1 = \pi_2 = \frac{1}{2}$, $\boldsymbol{\mu}_1 = \left(\begin{smallmatrix} 0.6 \\ 0 \end{smallmatrix}\right)$, $\boldsymbol{\mu}_2 = \left(\begin{smallmatrix} 0 \\ 0.6 \end{smallmatrix}\right)$, $\mathbf{\Sigma}_1 = \left(\begin{smallmatrix} 0.65 & 0 \\ 0 & 0.1 \end{smallmatrix}\right)$ and $\mathbf{\Sigma}_2 = \left(\begin{smallmatrix} 0.1 & 0 \\ 0 & 0.65 \end{smallmatrix}\right)$. This mixture has three modes (marked "△"): two nearly coincident with the centroids $\boldsymbol{\mu}_m$ (marked "+") and a third one near the meeting point of the components' principal axes. All the modes are outside the convex hull of the centroids.

The gradient of p can be written:

$$\frac{\partial p}{\partial t_d} = \sum_{e=1}^{D} \frac{\partial p}{\partial u_e} \frac{\partial u_e}{\partial t_d} = \sum_{e=1}^{D} \frac{\partial p}{\partial u_e} \left(\mathbf{\Lambda}^{\frac{1}{2}} \mathbf{U}^T \right)_{ed} \Longrightarrow \frac{dp}{d\mathbf{t}} = \mathbf{U}\mathbf{\Lambda}^{\frac{1}{2}} \frac{dp}{d\mathbf{u}}.$$

Since $\mathbf{U}\mathbf{\Lambda}^{\frac{1}{2}}$ is nonsingular, $\frac{dp}{d\mathbf{t}} = \mathbf{0} \Leftrightarrow \frac{dp}{d\mathbf{u}} = \mathbf{0}$ and so the stationary points are preserved by the transformation.

Now, if $\mathbf{t}$ is a point in the convex hull of $\{\boldsymbol{\mu}_m\}_{m=1}^M$ then $\mathbf{t} = \sum_{m=1}^M \lambda_m \boldsymbol{\mu}_m$ where $\{\lambda_m\}_{m=1}^M \subset [0, 1]$ and $\sum_{m=1}^M \lambda_m = 1$. So $\mathbf{u} = \mathbf{\Lambda}^{\frac{1}{2}} \mathbf{U}^T \mathbf{t} = \sum_{m=1}^M \lambda_m \mathbf{\Lambda}^{\frac{1}{2}} \mathbf{U}^T \boldsymbol{\mu}_m = \sum_{m=1}^M \lambda_m \boldsymbol{\nu}_m$ which is in the convex hull of $\{\boldsymbol{\nu}_m\}_{m=1}^M$. □

Remark. Theorems 8.7.1 and 8.7.2 show that cases 1 and 2 of conjecture 8.1 are particular cases of case 3.

Finally, figure 8.9 gives a simple example of a mixture with nonisotropic, different component covariance matrices that has more modes than components and the modes lie outside the convex hull of the centroids. I am grateful to Chris Williams for suggesting me this example. Another example can be seen in fig. 9.16(left): the mode located at $(-1.92, -4.7)$ (right at the curve elbow, revealed by the closed contours) lies just outside the convex hull of the centroids (marked $*$). Clearly, it is possible to construct more complicated examples where elongated components interact to create a variety of modes.

8.7.3 Convergence proof for the fixed-point mode search

Consider a D-dimensional Gaussian mixture of fixed parameters $\{\pi_m, \boldsymbol{\mu}_m, \mathbf{\Sigma}_m\}_{m=1}^M$:

$$p(\mathbf{t}) = \sum_{m=1}^{M} \pi_m p(\mathbf{t}|m) \text{ with } \mathbf{t}|m \sim \mathcal{N}(\boldsymbol{\mu}_m, \mathbf{\Sigma}_m) \text{ and } \mathbf{t} \in \mathbb{R}^D.$$

In section 8.3 we proposed the following iterative scheme to find modes of $p(\mathbf{t})$:

$$\mathbf{t}^{(\tau+1)} = \mathbf{f}(\mathbf{t}^{(\tau)}) \text{ with } \mathbf{f}(\mathbf{t}) \stackrel{\text{def}}{=} \left(\sum_{m=1}^{M} p(m|\mathbf{t}) \boldsymbol{\Sigma}_m^{-1} \right)^{-1} \sum_{m=1}^{M} p(m|\mathbf{t}) \boldsymbol{\Sigma}_m^{-1} \boldsymbol{\mu}_m, \tag{8.3$'$}$$

derived by equating to zero the gradient of $p(\mathbf{t})$ with respect to $\mathbf{t}$. To prove that this iterative scheme converges, one might try to use analysis techniques; e.g. it will converge to a fixed point of $\mathbf{f}$ if $\mathbf{f}$ is a contractive mapping in an environment of that fixed point (Isaacson and Keller, 1966). Instead, we give here a simpler proof based on the convergence properties of the EM algorithm. I am grateful to Chris Williams for suggesting me this proof.

Proof. The idea of the proof is to derive eq. (8.3$'$) as an EM algorithm. Consider the following density model with parameters $\mathbf{v} = (v_1, \dots, v_D)^T$ and fixed $\{\pi_m, \boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m\}_{m=1}^M$:

$$p(\mathbf{t}|\mathbf{v}) = \sum_{m=1}^{M} \pi_m \left|2\pi \boldsymbol{\Sigma}_m\right|^{-\frac{1}{2}} e^{-\frac{1}{2}(\mathbf{t}-(\boldsymbol{\mu}_m - \mathbf{v}))^T \boldsymbol{\Sigma}_m^{-1} (\mathbf{t}-(\boldsymbol{\mu}_m-\mathbf{v}))}.$$

That is, $\mathbf{t}|\mathbf{v}$ is a D-dimensional Gaussian mixture where component m has mixing proportion π_m (fixed), mean vector $\boldsymbol{\mu}_m - \mathbf{v}$ ($\boldsymbol{\mu}_m$ fixed) and covariance matrix $\boldsymbol{\Sigma}_m$ (fixed). Varying $\mathbf{v}$ results in a rigid translation of the whole mixture as a block rather than the individual components varying separately. Now consider fitting this model by maximum likelihood to a data set $\{\mathbf{t}_n\}_{n=1}^N$ and let us derive an EM algorithm to estimate the parameters $\mathbf{v}$ (see section 2.5). Call $z_n \in \{1, \dots, M\}$ the (unknown) index of the mixture component that generated data point $\mathbf{t}_n$. Then:

E step The complete-data log-likelihood, as if all $\{z_n\}_{n=1}^N$ were known, and assuming iid data, is $\sum_{n=1}^N \mathcal{L}_{n,\text{complete}}(\mathbf{v}) = \sum_{n=1}^N \ln p(\mathbf{t}_n, z_n|\mathbf{v})$ and so its expectation with respect to the current posterior distribution is

$$\begin{aligned} Q(\mathbf{v}|\mathbf{v}^{(\tau)}) &\stackrel{\text{def}}{=} \sum_{n=1}^{N} \mathrm{E}_{p(z_n|\mathbf{t}_n, \mathbf{v}^{(\tau)})} \left\{ \mathcal{L}_{n,\text{complete}}(\mathbf{v}) \right\} \\ &= \sum_{n=1}^{N} \sum_{z_n=1}^{M} p(z_n|\mathbf{t}_n, \mathbf{v}^{(\tau)}) \ln\{p(z_n|\mathbf{v}) p(\mathbf{t}_n|z_n, \mathbf{v})\} \\ &= \sum_{n=1}^{N} \sum_{z_n=1}^{M} p(z_n|\mathbf{t}_n, \mathbf{v}^{(\tau)}) \ln p(\mathbf{t}_n|z_n, \mathbf{v}) + K \end{aligned}$$

where the term $K \stackrel{\text{def}}{=} \sum_{n=1}^N \sum_{z_n=1}^M p(z_n|\mathbf{t}_n, \mathbf{v}^{(\tau)}) \ln \pi_{z_n}$ is independent of $\mathbf{v}$.

M step The new parameter estimates $\mathbf{v}^{(\tau+1)}$ are obtained from the old ones $\mathbf{v}^{(\tau)}$ as $\mathbf{v}^{(\tau+1)} = \arg\max_{\mathbf{v}} Q(\mathbf{v}|\mathbf{v}^{(\tau)})$. To perform this maximisation, we equate the gradient of Q with respect to $\mathbf{v}$ to zero:

$$\frac{\partial Q}{\partial \mathbf{v}} = \sum_{n=1}^{N} \sum_{z_n=1}^{M} p(z_n|\mathbf{t}_n, \mathbf{v}^{(\tau)}) \frac{1}{p(\mathbf{t}_n|z_n, \mathbf{v})} \frac{\partial p(\mathbf{t}_n|z_n, \mathbf{v})}{\partial \mathbf{v}} = \mathbf{0}. \tag{8.16}$$

As a function of $\mathbf{v}$, $p(\mathbf{t}_n|z_n, \mathbf{v})$ is a Gaussian density of mean $\boldsymbol{\mu}_{z_n} - \mathbf{t}_n$ and covariance $\boldsymbol{\Sigma}_{z_n}$, so from eq. (8.9) we get

$$\frac{\partial p(\mathbf{t}_n|z_n, \mathbf{v})}{\partial \mathbf{v}} = p(\mathbf{t}_n|z_n, \mathbf{v}) \boldsymbol{\Sigma}_{z_n}^{-1} (\boldsymbol{\mu}_{z_n} - \mathbf{t}_n - \mathbf{v})$$

and so solving for $\mathbf{v}$ in eq. (8.16) results in

$$\mathbf{v}^{(\tau+1)} = \left(\sum_{n=1}^{N} \sum_{z_n=1}^{M} p(z_n|\mathbf{t}_n, \mathbf{v}^{(\tau)}) \boldsymbol{\Sigma}_{z_n}^{-1} \right)^{-1} \sum_{n=1}^{N} \sum_{z_n=1}^{M} p(z_n|\mathbf{t}_n, \mathbf{v}^{(\tau)}) \boldsymbol{\Sigma}_{z_n}^{-1} (\boldsymbol{\mu}_{z_n} - \mathbf{t}_n).$$

If now we choose the data set as simply containing the origin, $\{\mathbf{t}_n\}_{n=1}^N = \{\mathbf{0}\}$, rename $z_1 = m$ and omit $\mathbf{t}_1 = \mathbf{0}$ for clarity, we obtain the M step as:

$$\mathbf{v}^{(\tau+1)} = \left(\sum_{m=1}^{M} p(m|\mathbf{v}^{(\tau)}) \boldsymbol{\Sigma}_m^{-1} \right)^{-1} \sum_{m=1}^{M} p(m|\mathbf{v}^{(\tau)}) \boldsymbol{\Sigma}_m^{-1} \boldsymbol{\mu}_m \tag{8.17}$$

which is formally identical to the iterative scheme of eq. (8.3$'$).

The EM algorithm for Gaussian mixtures converges from any starting point (Dempster et al., 1977; Redner and Walker, 1984). Specifically, at every iteration τ, the iterative scheme (8.17) will either increase the log-likelihood or leave it unchanged. The log-likelihood is

$$\sum_{n=1}^{N} \ln p(\mathbf{t}_n|\mathbf{v}) = \sum_{n=1}^{N} \ln \sum_{m=1}^{M} \pi_m p(\mathbf{t}_n|m, \mathbf{v}) = \ln \sum_{m=1}^{M} \pi_m \left|2\pi\mathbf{\Sigma}_m\right|^{-1} e^{-\frac{1}{2}(\mathbf{v}-\boldsymbol{\mu}_m)^T\mathbf{\Sigma}_m^{-1}(\mathbf{v}-\boldsymbol{\mu}_m)}$$

so, correspondingly, the iterative scheme (8.3′) will monotonically increase the density value $p(\mathbf{t})$ or leave it unchanged. Thus, (8.3′) converges from any initial value of $\mathbf{t}$. In principle, though, convergence can occur to a saddle point or to a minimum as well as to a mode (in the very unlikely case where the initial value is at a minimum, the scheme will remain stuck at it). Since both saddle points and minima are unstable for maximisation, a small random perturbation will cause the EM algorithm to diverge from them. Thus, practical convergence will almost always be to a mode. □

8.7.4 Efficient operations with the Hessian

The Hessian and the gradient of a Gaussian mixture are a linear superposition of terms. This makes possible, in some particular but useful cases (e.g. GTM or Gaussian kernel density estimation), to perform efficiently and exactly (i.e., with no approximations involved) various operations required by the optimisation procedure of section 8.2:

- If the matrix $\mathbf{S}$ defined below is easily invertible (e.g., if each covariance matrix $\mathbf{\Sigma}_m$ is diagonal, a common situation in many engineering applications), the quadratic step of eq. (8.1) can be performed without inverting the Hessian (theorem 8.7.4 and observation 8.7.5).
- If the number of mixture components is smaller than the dimensionality of the space, $M < D$, the inverse Hessian can be computed inverting an $M \times M$ matrix rather than a $D \times D$ matrix (theorem 8.7.3).

Define the following matrices:

$$\begin{aligned}
&\mathbf{S}_{D\times D} \stackrel{\text{def}}{=} -\sum_{m=1}^{M} p(\mathbf{t}, m)\mathbf{\Sigma}_m^{-1} \\
&\mathbf{R}_{D\times M} = (\mathbf{r}_1, \dots, \mathbf{r}_M) \text{ where } \mathbf{r}_m \stackrel{\text{def}}{=} \sqrt{p(\mathbf{t}, m)}\mathbf{\Sigma}_m^{-1}(\boldsymbol{\mu}_m - \mathbf{t}) \\
&\mathbf{T}_{M\times M} \stackrel{\text{def}}{=} \mathbf{I}_M + \mathbf{R}^T\mathbf{S}^{-1}\mathbf{R}
\end{aligned}$$

so that $\mathbf{R}\mathbf{R}^T = \sum_{m=1}^{M} \mathbf{r}_m\mathbf{r}_m^T$ and $\mathbf{H} = \mathbf{S} + \mathbf{R}\mathbf{R}^T$, from eq. (8.12).

The proofs in this section and in the remaining ones will often make use of the Sherman-Morrison-Woodbury (SMW) formula A.2.

Theorem 8.7.3. $\mathbf{H}^{-1} = \mathbf{S}^{-1}(\mathbf{S} - \mathbf{R}\mathbf{T}^{-1}\mathbf{R}^T)\mathbf{S}^{-1}$.

Proof. By the SMW formula. □

Theorem 8.7.4. $\mathbf{H}^{-1}\mathbf{g} = \mathbf{S}^{-1}\mathbf{H}\mathbf{S}^{-1}\mathbf{g}$. *If* $\mathbf{\Sigma}_m = \sigma^2\mathbf{I}_D$ *then* $\mathbf{H}^{-1}\mathbf{g} = \left(\frac{\sigma^2}{p(\mathbf{t})}\right)^2 \mathbf{H}\mathbf{g}$.

Proof. Define an $M \times 1$ vector $\mathbf{v}$ with components $v_m \stackrel{\text{def}}{=} \sqrt{p(\mathbf{t}, m)}$, so that $\mathbf{R}\mathbf{v} = \mathbf{g}$ (see eq. (8.11)). Then:

$$\begin{aligned}
\mathbf{H}^{-1}\mathbf{g} &= \mathbf{H}^{-1}\mathbf{R}\mathbf{v} = \mathbf{S}^{-1}\mathbf{R}\mathbf{T}^{-1}\mathbf{v} = \mathbf{S}^{-1}\mathbf{R}(\mathbf{v} + \mathbf{R}^T\mathbf{S}^{-1}\mathbf{g}) \\
&= \mathbf{S}^{-1}\mathbf{g} + \mathbf{S}^{-1}\mathbf{R}\mathbf{R}^T\mathbf{S}^{-1}\mathbf{g} = \mathbf{S}^{-1}(\mathbf{S} + \mathbf{R}\mathbf{R}^T)\mathbf{S}^{-1}\mathbf{g} = \mathbf{S}^{-1}\mathbf{H}\mathbf{S}^{-1}\mathbf{g}
\end{aligned}$$

where we have used $\mathbf{H}^{-1}\mathbf{R} = \mathbf{S}^{-1}\mathbf{R}\mathbf{T}^{-1}$, which again can be proved using the SMW formula. □

Observation 8.7.5. *Since* $\mathbf{H} = \mathbf{S} + \sum_{m=1}^{M} \mathbf{r}_m\mathbf{r}_m^T$, *the Hessian can be inverted by repeatedly using the following particular case of the SMW formula:*

$$\mathbf{A}_{D\times D}, \mathbf{v}_{D\times 1}: \quad (\mathbf{A} + \mathbf{v}\mathbf{v}^T)^{-1} = \mathbf{A}^{-1} - \frac{\mathbf{A}^{-1}\mathbf{v}\mathbf{v}^T\mathbf{A}^{-1}}{1 + \mathbf{v}^T\mathbf{A}^{-1}\mathbf{v}}.$$

The only operation not covered here that is required by the optimisation procedure is the determination of whether the Hessian is negative definite. This can be accomplished by checking the signs of its principal minors (Mirsky, 1955, p. 403) or by numerical methods, such as Cholesky decomposition (Golub and van Loan, 1996, p. 143; Press et al., 1992, p. 96).

8.7.5 Bounds for the gradient and the Hessian

Theorem 8.7.6. *For a normal distribution $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, the norm of the gradient is bounded as $0 \leq \|\mathbf{g}\| \leq (e\lambda_{min} |2\pi\boldsymbol{\Sigma}|)^{-1/2}$, where λ_{min} is the smallest eigenvalue of $\boldsymbol{\Sigma}$.*

Proof. Obviously $\|\mathbf{g}\| \geq 0$, with $\mathbf{g} = \mathbf{0}$ attained at $\mathbf{t} = \boldsymbol{\mu}$. For a one-dimensional distribution $\mathcal{N}(\mu, \lambda)$, the maximum gradient norm, of value $(2\pi\lambda^2 e)^{-1/2}$, happens at the inflexion points $\left|\frac{t-\mu}{\sqrt{\lambda}}\right| = 1$. Using these results, let us prove the theorem for the D-dimensional normal $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. Here $\mathbf{g} = \boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu} - \mathbf{t})p(\mathbf{t})$ and $\|\mathbf{g}\| = \|\boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu} - \mathbf{t})\| \, p(\mathbf{t})$. Change $\mathbf{z} = \boldsymbol{\Lambda}^{-1}\mathbf{U}^T(\boldsymbol{\mu} - \mathbf{t})$ where $\boldsymbol{\Sigma} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^T$ is the singular value decomposition of $\boldsymbol{\Sigma}$:

$$\|\mathbf{g}\| = \left\|\mathbf{U}\boldsymbol{\Lambda}^{1-}\mathbf{U}^T(\boldsymbol{\mu} - \mathbf{t})\right\| p(\mathbf{t}) = \|\mathbf{U}\mathbf{z}\| \, |2\pi\boldsymbol{\Sigma}|^{-\frac{1}{2}} \, e^{-\frac{1}{2}\mathbf{z}^T\boldsymbol{\Lambda}\mathbf{z}} = \|\mathbf{z}\| \, |2\pi\boldsymbol{\Lambda}|^{-\frac{1}{2}} \, e^{-\frac{1}{2}\mathbf{z}^T\boldsymbol{\Lambda}\mathbf{z}}$$

since $\mathbf{U} = (\mathbf{u}_1, \dots, \mathbf{u}_D)$ is orthonormal. Consider ellipses where $\mathbf{z}^T\boldsymbol{\Lambda}\mathbf{z} = \rho^2$ with $\rho > 0$. In any of those ellipses, the point with maximum norm (i.e., maximum distance to the origin) lies along the direction of the smallest $\lambda_d = \lambda_{\min}$, for some[7] $d \in \{1, \dots, D\}$. Thus, using Dirac delta notation, $z_c = \pm\delta_{cd}\rho\lambda_d^{-1/2}$ and $\|\mathbf{g}\| = \rho\lambda_d^{-1/2} \, |2\pi\boldsymbol{\Lambda}|^{-1/2} \, e^{-\frac{1}{2}\rho^2}$ there. Now, from the result for the one-dimensional case, this expression is maximum at $\rho = 1$. So the gradient norm is maximum at $\mathbf{z}^*$ with components $z_c^* = \pm\delta_{cd}\rho\lambda_d^{-1/2}$ for $c = 1, \dots, c$, or $\mathbf{t}^* = \boldsymbol{\mu} - \mathbf{U}\boldsymbol{\Lambda}\mathbf{z}^* = \boldsymbol{\mu} \pm \lambda_d^{1/2}\mathbf{u}_d$ and $\|\mathbf{g}^*\| = (e\lambda_{\min} |2\pi\boldsymbol{\Lambda}|)^{-1/2} = (e\lambda_{\min} |2\pi\boldsymbol{\Sigma}|)^{-1/2}$ there. □

Corollary 8.7.7. *If $\boldsymbol{\Sigma} = \sigma^2\mathbf{I}_D$, then $\|\mathbf{g}\|$ is maximum at the hypersphere $\left\|\frac{\boldsymbol{\mu}-\mathbf{t}}{\sigma}\right\| = 1$ and there $\|\mathbf{g}_{max}\| = (e\sigma^2(2\pi\sigma^2)^D)^{-1/2}$.*

Theorem 8.7.8. *For a mixture of Gaussians, the gradient norm is smaller or equal than the maximum gradient norm achievable by any of the components. If $\boldsymbol{\Sigma}_m = \sigma^2\mathbf{I}_D$ for all $m = 1, \dots, M$, then $\|\mathbf{g}\| \leq (e\sigma^2(2\pi\sigma^2)^D)^{-1/2}$.*

Proof. For the mixture and using the triangle inequality:

$$\|\mathbf{g}\| = \left\|\sum_{m=1}^{M} p(m)\mathbf{g}_m\right\| \leq \sum_{m=1}^{M} p(m) \, \|\mathbf{g}_m\| \leq \sum_{m=1}^{M} p(m) \, \|\mathbf{g}_{m,\max}\| \leq \sum_{m=1}^{M} p(m) \, \|\mathbf{g}\|_{\max} = \|\mathbf{g}\|_{\max}$$

where $\|\mathbf{g}_{m,\max}\|$ is given by theorem 8.7.8 for each component m and $\|\mathbf{g}\|_{\max} = \max_{m=1\dots,M} \|\mathbf{g}_{m,\max}\|$. Geometrically, this is a result of the coalescence of the individual components. □

The following theorem gives a sufficient condition for the Hessian of the mixture of isotropic Gaussians to be negative definite.

Theorem 8.7.9. *If $\boldsymbol{\Sigma}_m = \sigma^2\mathbf{I}_D$, then $\mathbf{H} < 0$ if $\sum_{m=1}^{M} p(m|\mathbf{t}) \left\|\frac{\boldsymbol{\mu}_m-\mathbf{t}}{\sigma}\right\|^2 < 1$.*

Proof. Call $\boldsymbol{\rho}_m = \frac{\boldsymbol{\mu}_m-\mathbf{t}}{\sigma}$, $\rho_m = \|\boldsymbol{\rho}_m\|$ and $\mathbf{H}_0 = -\mathbf{I}_D + \sum_{m=1}^{M} p(m|\mathbf{t})\boldsymbol{\rho}_m\boldsymbol{\rho}_m^T$. From eq. (8.12) $\mathbf{H} = \frac{p(\mathbf{t})}{\sigma^2}\mathbf{H}_0$. From lemma 8.7.10, each eigenvalue of $\mathbf{H}_0$ is $\lambda_d = -1 + \sum_{m=1}^{M} \pi_{md}p(m|\mathbf{t})\rho_m^2$ where $\sum_{d=1}^{D} \pi_{md} = 1$ for each $m = 1, \dots, M$. A worst-case analysis gives $\lambda_d \leq -1 + \sum_{m=1}^{M} p(m|\mathbf{t})\rho_m^2$ for a certain $d \in \{1, \dots, D\}$. Thus the Hessian will be negative definite if $\sum_{m=1}^{M} p(m|\mathbf{t})\rho_m^2 < 1$. □

The following lemma shows the effect on the eigenvalues of a symmetric matrix of a series of unit-rank perturbations and is necessary to prove theorem 8.7.9.

Lemma 8.7.10. *Let $\mathbf{A}$ be a symmetric $D \times D$ matrix with eigenvalues $\lambda_1, \dots, \lambda_D$ and $\{\mathbf{u}_m\}_{m=1}^{M}$ a set of M vectors in $\mathbb{R}^D$. Then, the eigenvalues of $\mathbf{A} + \sum_{m=1}^{M} \mathbf{u}_m\mathbf{u}_m^T$ are $\lambda'_d = \lambda_d + \sum_{m=1}^{M} \pi_{md}\mathbf{u}_m^T\mathbf{u}_m$ for some unknown coefficients π_{md} satisfying $\pi_{md} \in [0, 1]$ for $m = 1, \dots, M$, $d = 1, \dots, D$, and $\sum_{d=1}^{D} \pi_{md} = 1$ for each $m = 1, \dots, M$. Thus, every eigenvalue is shifted by an amount which lies between 0 and the squared norm of the perturbing vector, for each perturbation.*

Proof. A proof for the case $M = 1$ is given by Wilkinson (1965, pp. 97–98). The case of arbitrary M follows by repeated application of the case $M = 1$. □

[7] There may be several values of d with the same $\lambda_d = \lambda_{\min}$, but this is irrelevant for the proof.

8.7.6 Bounds for the entropy

Theorem 8.7.11. *The entropy and L_2-norm for a normal distribution of mean vector $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$ are:*

- $h(\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})) = \frac{1}{2}\ln|2\pi e\boldsymbol{\Sigma}|$.
- $\|\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})\|_2 = |4\pi\boldsymbol{\Sigma}|^{-1/4}$.

Theorem 8.7.12 (Information theory bounds on the entropy of a mixture). *For a finite mixture $p(\mathbf{t})$ not necessarily Gaussian with mean vector $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$: $\sum_{m=1}^{M}\pi_m h(p(\mathbf{t}|m)) \leq h(p(\mathbf{t})) \leq \frac{1}{2}\ln\left((2\pi e)^D|\boldsymbol{\Sigma}|\right)$. Equality can only be obtained in trivial mixtures where $M = 1$ or $\mathbf{t}$ and m are independent, i.e., all components are equal.*

Proof.

- LHS inequality: by the fact that conditioning reduces the entropy (or by the concavity of the entropy; Cover and Thomas, 1991, p. 27, 232, 483): $h(p(\mathbf{t})) \geq -\operatorname{E}_{p(\mathbf{t},m)}\{\ln p(\mathbf{t}|m)\} = \sum_{m=1}^{M}\pi_m h(p(\mathbf{t}|m))$.
- RHS inequality: by the fact that, for fixed mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$, the maximum entropy distribution is the normal distribution of mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$, $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, whose entropy is $\frac{1}{2}\ln\left((2\pi e)^D|\boldsymbol{\Sigma}|\right)$ and the logarithm is in any base (Cover and Thomas, 1991, p. 225, 270). $\square$

Theorem 8.7.13 (L_2-norm lower bound on the entropy). [8] *For any density p: $h(p) \geq -2\ln\|p\|_2$.*

Proof. Since the function $-\ln t$ is convex, Jensen's inequality gives $h(p) \stackrel{\text{def}}{=} \operatorname{E}_p\{-\ln p(\mathbf{t})\} \geq -\ln \operatorname{E}_p\{p(\mathbf{t})\} = -2\ln\|p\|_2$. $\square$

Theorem 8.7.14. *For a Gaussian mixture $p(\mathbf{t}) = \sum_{m=1}^{M} p(m)p(\mathbf{t}|m)$:*

$$\|p\|_2^2 = \sum_{m,n=1}^{M} p(m)p(n)\left|2\pi(\boldsymbol{\Sigma}_m + \boldsymbol{\Sigma}_n)\right|^{-\frac{1}{2}} e^{-\frac{1}{2}(\boldsymbol{\mu}_m-\boldsymbol{\mu}_n)^T(\boldsymbol{\Sigma}_m+\boldsymbol{\Sigma}_n)^{-1}(\boldsymbol{\mu}_m-\boldsymbol{\mu}_n)}.$$

Proof. $\|p\|_2^2 = \langle p, p\rangle = \sum_{m,n=1}^{M} p(m)p(n)\langle\mathcal{N}(\boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m)\mathcal{N}(\boldsymbol{\mu}_n, \boldsymbol{\Sigma}_n)\rangle$. Computing the scalar product is tedious and is summarised as follows:

$$\begin{aligned}
&\langle\mathcal{N}(\boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m)\mathcal{N}(\boldsymbol{\mu}_n, \boldsymbol{\Sigma}_n)\rangle = \int_{\mathbb{R}^D} p(\mathbf{t}|m)p(\mathbf{t}|n)\,d\mathbf{t} = \\
&\int_{\mathbb{R}^D} |2\pi\boldsymbol{\Sigma}_m|^{-\frac{1}{2}}|2\pi\boldsymbol{\Sigma}_n|^{-\frac{1}{2}} e^{-\frac{1}{2}\left((\mathbf{t}-\boldsymbol{\mu}_m)^T\boldsymbol{\Sigma}_m^{-1}(\mathbf{t}-\boldsymbol{\mu}_m)+(\mathbf{t}-\boldsymbol{\mu}_n)^T\boldsymbol{\Sigma}_n^{-1}(\mathbf{t}-\boldsymbol{\mu}_n)\right)}\,d\mathbf{t} = \\
&|2\pi\boldsymbol{\Sigma}_m|^{-\frac{1}{2}}|2\pi\boldsymbol{\Sigma}_n|^{-\frac{1}{2}}\int_{\mathbb{R}^D} e^{-\frac{1}{2}(\mathbf{t}-\boldsymbol{\mu})^T(\boldsymbol{\Sigma}_m^{-1}+\boldsymbol{\Sigma}_n^{-1})(\mathbf{t}-\boldsymbol{\mu})-(\boldsymbol{\mu}_m^T\boldsymbol{\Sigma}_n^{-1}+\boldsymbol{\mu}_n^T\boldsymbol{\Sigma}_m^{-1})(\mathbf{t}-\boldsymbol{\mu})-\frac{1}{2}(\boldsymbol{\mu}_m^T\boldsymbol{\Sigma}_n^{-1}\boldsymbol{\mu}_m+\boldsymbol{\mu}_n^T\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\mu}_n)}\,d\mathbf{t} = \\
&|2\pi\boldsymbol{\Sigma}_m|^{-\frac{1}{2}}|2\pi\boldsymbol{\Sigma}_n|^{-\frac{1}{2}}\left|2\pi(\boldsymbol{\Sigma}_m^{-1}+\boldsymbol{\Sigma}_n^{-1})^{-1}\right|^{\frac{1}{2}}\times \\
&e^{-\frac{1}{2}(\boldsymbol{\mu}_m^T\boldsymbol{\Sigma}_n^{-1}\boldsymbol{\mu}_m+\boldsymbol{\mu}_n^T\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\mu}_n)} e^{\frac{1}{2}(\boldsymbol{\mu}_m^T\boldsymbol{\Sigma}_n^{-1}+\boldsymbol{\mu}_n^T\boldsymbol{\Sigma}_m^{-1})(\boldsymbol{\Sigma}_m^{-1}+\boldsymbol{\Sigma}_n^{-1})^{-1}(\boldsymbol{\mu}_m^T\boldsymbol{\Sigma}_n^{-1}+\boldsymbol{\mu}_n^T\boldsymbol{\Sigma}_m^{-1})^T} = \\
&|2\pi(\boldsymbol{\Sigma}_m+\boldsymbol{\Sigma}_n)|^{-\frac{1}{2}} e^{-\frac{1}{2}(\boldsymbol{\mu}_m-\boldsymbol{\mu}_n)^T(\boldsymbol{\Sigma}_m+\boldsymbol{\Sigma}_n)^{-1}(\boldsymbol{\mu}_m-\boldsymbol{\mu}_n)}
\end{aligned}$$

where we have used the following facts:

$$\int_{\mathbb{R}^D} e^{-\frac{1}{2}(\mathbf{t}-\mathbf{a})^T\mathbf{A}(\mathbf{t}-\mathbf{a})+\mathbf{b}^T(\mathbf{t}-\mathbf{a})}\,d\mathbf{t} = \left|\frac{\mathbf{A}}{2\pi}\right|^{-\frac{1}{2}} e^{\frac{1}{2}\mathbf{b}^T\mathbf{A}^{-1}\mathbf{b}}$$

$$\boldsymbol{\Sigma}_m^{-1}(\boldsymbol{\Sigma}_m^{-1}+\boldsymbol{\Sigma}_n^{-1})^{-1}\boldsymbol{\Sigma}_n^{-1} = (\boldsymbol{\Sigma}_m+\boldsymbol{\Sigma}_n)^{-1}$$

$$\boldsymbol{\Sigma}_m^{-1}(\boldsymbol{\Sigma}_m^{-1}+\boldsymbol{\Sigma}_n^{-1})^{-1}\boldsymbol{\Sigma}_m^{-1} = (\boldsymbol{\Sigma}_m(\boldsymbol{\Sigma}_m^{-1}+\boldsymbol{\Sigma}_n^{-1})\boldsymbol{\Sigma}_m^{-1})^{-1} = (\boldsymbol{\Sigma}_m(\mathbf{I}+\boldsymbol{\Sigma}_n^{-1}\boldsymbol{\Sigma}_m))^{-1} = \boldsymbol{\Sigma}_m^{-1}-(\boldsymbol{\Sigma}_m+\boldsymbol{\Sigma}_n)^{-1}$$

and the SMW formula. $\square$

[8] I am grateful to Chris Williams for suggesting to me the use of the L_2-norm.

Corollary 8.7.15. *For a finite Gaussian mixture with components* $\mathbf{t}|m \sim \mathcal{N}(\boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m)$*,* $m = 1, \dots, M$ *and with mean vector* $\boldsymbol{\mu}$ *and covariance matrix* $\boldsymbol{\Sigma}$ *given in section 8.7:* $\max(\mathsf{LB}_1, \mathsf{LB}_2) \leq h(p(\mathbf{t})) \leq \mathsf{UB}_1$*, where:*

$$\mathsf{LB}_1 \stackrel{\text{def}}{=} \frac{1}{2} \ln \left\{ (2\pi e)^D \prod_{m=1}^{M} |\boldsymbol{\Sigma}_m|^{\pi_m} \right\}$$

$$\mathsf{LB}_2 \stackrel{\text{def}}{=} -\ln \left\{ \sum_{m,n=1}^{M} p(m)p(n) \left|2\pi(\boldsymbol{\Sigma}_m + \boldsymbol{\Sigma}_n)\right|^{-\frac{1}{2}} e^{-\frac{1}{2}(\boldsymbol{\mu}_m - \boldsymbol{\mu}_n)^T(\boldsymbol{\Sigma}_m + \boldsymbol{\Sigma}_n)^{-1}(\boldsymbol{\mu}_m - \boldsymbol{\mu}_n)} \right\}$$

$$\mathsf{UB}_1 \stackrel{\text{def}}{=} \frac{1}{2} \ln \left((2\pi e)^D |\boldsymbol{\Sigma}| \right).$$

Equality $\mathsf{LB}_1 = h(p(\mathbf{t})) = \mathsf{UB}_1$ *can only be obtained in trivial mixtures where* $M = 1$ *or* $\boldsymbol{\mu}_m = \boldsymbol{\mu}$ *and* $\boldsymbol{\Sigma}_m = \boldsymbol{\Sigma}$ *for all* $m = 1, \dots, M$*, in which case* $\mathsf{LB}_2 = \frac{1}{2} \ln |4\pi\boldsymbol{\Sigma}| < \frac{1}{2} \ln |2\pi e\boldsymbol{\Sigma}| = \mathsf{LB}_1 = h(p(\mathbf{t})) = \mathsf{UB}_1$*.*

Note the limit cases:

- $\Sigma_m \to 0$: the upper and lower bounds and the entropy tend to $-\infty$.
- $\Sigma_m \to \infty$: the upper and lower bounds and the entropy tend to ∞.

Observe that LB_1 can be smaller or greater than LB_2 depending on the values of $\{\pi_m, \boldsymbol{\Sigma}_m\}_{m=1}^M$.

Remark. From corollary 8.7.15, it is easy to proof the following generalised matrix formulation of the inequality between the geometric mean and the arithmetic mean. If $\{\boldsymbol{\Sigma}_m\}_{m=1}^M$ are symmetric positive definite matrices with real elements and $\{\pi_m\}_{m=1}^M$ are arbitrary real numbers in $[0, 1]$ with $\sum_{m=1}^M \pi_m = 1$, then:

$$\prod_{m=1}^{M} |\boldsymbol{\Sigma}_m|^{\pi_m} \leq \left| \sum_{m=1}^{M} \pi_m \boldsymbol{\Sigma}_m \right|$$

with equality if and only if $M = 1$ or some $\pi_m = 1$. This result, which guarantees the concavity of the function $\ln\det(\cdot)$, was proved by Fan (1950) using linear algebra rather than information theory.

8.7.7 Additional results

Theorem 8.7.16. $\operatorname{tr}(\mathbf{H}) = \sum_{m=1}^M p(\mathbf{t}, m) \left((\boldsymbol{\mu}_m - \mathbf{t})^T \boldsymbol{\Sigma}_m^{-2} (\boldsymbol{\mu}_m - \mathbf{t}) - \operatorname{tr}\left(\boldsymbol{\Sigma}_m^{-1}\right) \right)$.

Proof.

$$\begin{aligned}
\operatorname{tr}(\mathbf{H}) &= \operatorname{tr}\left(\sum_{m=1}^{M} p(\mathbf{t}, m) \boldsymbol{\Sigma}_m^{-1} ((\boldsymbol{\mu}_m - \mathbf{t})(\boldsymbol{\mu}_m - \mathbf{t})^T - \boldsymbol{\Sigma}_m) \boldsymbol{\Sigma}_m^{-1} \right) \\
&= \sum_{m=1}^{M} p(\mathbf{t}, m) \left(\operatorname{tr}\left((\boldsymbol{\mu}_m - \mathbf{t})(\boldsymbol{\mu}_m - \mathbf{t})^T \boldsymbol{\Sigma}_m^{-2} \right) - \operatorname{tr}\left(\boldsymbol{\Sigma}_m^{-1}\right) \right) \\
&= \sum_{m=1}^{M} p(\mathbf{t}, m) \left((\boldsymbol{\mu}_m - \mathbf{t})^T \boldsymbol{\Sigma}_m^{-2} (\boldsymbol{\mu}_m - \mathbf{t}) - \operatorname{tr}\left(\boldsymbol{\Sigma}_m^{-1}\right) \right). \quad \square
\end{aligned}$$

Corollary 8.7.17. *If* $\boldsymbol{\Sigma}_m = \sigma^2 \mathbf{I}_D$ *for all* $m = 1, \dots, M$*, then* $\operatorname{tr}(\mathbf{H}) = -\frac{D}{\sigma^2} p(\mathbf{t}) + \frac{1}{\sigma^2} \sum_{m=1}^M p(\mathbf{t}, m) \left\| \frac{\boldsymbol{\mu}_m - \mathbf{t}}{\sigma} \right\|^2$*.*

Theorem 8.7.18. $\int_{\mathbb{R}^D} \operatorname{tr}(\mathbf{H}) \, d\mathbf{t} = 0$*.*

Proof. From theorem 8.7.16:

$$\begin{aligned}
\int_{\mathbb{R}^D} \operatorname{tr}(\mathbf{H}) \, d\mathbf{t} &= \int_{\mathbb{R}^D} \sum_{m=1}^{M} -p(\mathbf{t}, m) \operatorname{tr}\left(\boldsymbol{\Sigma}_m^{-1}\right) \, d\mathbf{t} + \int_{\mathbb{R}^D} \sum_{m=1}^{M} p(\mathbf{t}, m) (\boldsymbol{\mu}_m - \mathbf{t})^T \boldsymbol{\Sigma}_m^{-2} (\boldsymbol{\mu}_m - \mathbf{t}) \, d\mathbf{t} \\
&= -\sum_{m=1}^{M} p(m) \operatorname{tr}\left(\boldsymbol{\Sigma}_m^{-1}\right) + \sum_{m=1}^{M} p(m) \int_{\mathbb{R}^D} p(\mathbf{t}|m) (\boldsymbol{\mu}_m - \mathbf{t})^T \boldsymbol{\Sigma}_m^{-2} (\boldsymbol{\mu}_m - \mathbf{t}) \, d\mathbf{t}.
\end{aligned}$$

The latter integral can be solved by changing $\mathbf{z}_m = \mathbf{U}_m^T \mathbf{\Sigma}_m^{-1/2}(\boldsymbol{\mu}_m - \mathbf{t})$, where $\mathbf{\Sigma}_m = \mathbf{U}_m \mathbf{\Lambda}_m \mathbf{U}_m^T$ is the singular value decomposition of $\mathbf{\Sigma}_m$, and introducing the reciprocal of the determinant of the Jacobian, $\left|\mathbf{U}_m^T \mathbf{\Sigma}_m^{-1/2}\right|^{-1} = \left|\mathbf{\Sigma}_m\right|^{1/2}$ (since $\mathbf{U}_m$ is orthogonal):

$$
\begin{aligned}
&\int_{\mathbb{R}^D} |2\pi\mathbf{\Sigma}_m|^{-\frac{1}{2}} \, e^{-\frac{1}{2}(\mathbf{t}-\boldsymbol{\mu}_m)^T \mathbf{\Sigma}_m^{-1}(\mathbf{t}-\boldsymbol{\mu}_m)} (\boldsymbol{\mu}_m - \mathbf{t})^T \mathbf{\Sigma}_m^{-2} (\boldsymbol{\mu}_m - \mathbf{t}) \, d\mathbf{t} \\
&= \int_{\mathbb{R}^D} |2\pi\mathbf{\Sigma}_m|^{-\frac{1}{2}} \, e^{-\frac{1}{2}\mathbf{z}_m^T \mathbf{z}_m} \mathbf{z}_m^T \mathbf{\Lambda}_m^{-1} \mathbf{z}_m |\mathbf{\Sigma}_m|^{\frac{1}{2}} \, d\mathbf{z}_m = \mathrm{E}_{\mathcal{N}(\mathbf{0},\mathbf{I}_D)} \left\{ \sum_{d=1}^{D} \frac{z_{md}^2}{\lambda_{md}} \right\} = \sum_{d=1}^{D} \frac{1}{\lambda_{md}} \, \mathrm{E}_{\mathcal{N}(\mathbf{0},\mathbf{I}_D)} \left\{ z_{md}^2 \right\} \\
&= \sum_{d=1}^{D} \frac{1}{\lambda_{md}} = \operatorname{tr}\left(\mathbf{\Lambda}_m^{-1}\right) = \operatorname{tr}\left(\mathbf{\Lambda}_m^{-1} \mathbf{U}_m^T \mathbf{U}_m\right) = \operatorname{tr}\left(\mathbf{U}_m \mathbf{\Lambda}_m^{-1} \mathbf{U}_m^T\right) = \operatorname{tr}\left(\mathbf{\Sigma}_m^{-1}\right).
\end{aligned}
$$

Thus:

$$
\int_{\mathbb{R}^D} \operatorname{tr}(\mathbf{H}) \, d\mathbf{t} = -\sum_{m=1}^{M} p(m) \operatorname{tr}\left(\mathbf{\Sigma}_m^{-1}\right) + \sum_{m=1}^{M} p(m) \operatorname{tr}\left(\mathbf{\Sigma}_m^{-1}\right) = 0. \quad \square
$$

Theorem 8.7.19. $\int_{\mathbb{R}^D} \mathbf{g} \, d\mathbf{t} = \mathbf{0}$.

Proof. From eq. (8.11):

$$
\begin{aligned}
\int_{\mathbb{R}^D} \mathbf{g} \, d\mathbf{t} &= \int_{\mathbb{R}^D} \sum_{m=1}^{M} p(\mathbf{t}, m) \mathbf{\Sigma}_m^{-1} (\boldsymbol{\mu}_m - \mathbf{t}) \, d\mathbf{t} = \sum_{m=1}^{M} p(m) \mathbf{\Sigma}_m^{-1} \int_{\mathbb{R}^D} p(\mathbf{t}|m) (\boldsymbol{\mu}_m - \mathbf{t}) \, d\mathbf{t} \\
&= \sum_{m=1}^{M} p(m) \mathbf{\Sigma}_m^{-1} \, \mathrm{E}_{\mathcal{N}(\boldsymbol{\mu}_m, \mathbf{\Sigma}_m)} \left\{ \boldsymbol{\mu}_m - \mathbf{t} \right\} \, d\mathbf{t} = \mathbf{0}. \quad \square
\end{aligned}
$$

Chapter 9

Experiments with synthetic data

In this chapter we demonstrate the use of the missing data reconstruction method developed in chapter 7 with two synthetic data examples: a two-dimensional toy problem and the inverse kinematics of a two-link robot arm. They allow investigation of the method and of the conditions under which it may fail, as well as easy visualisation of the results. Section 9.1 describes the methodological setup followed to carry out the experiments, while sections 9.2 and 9.3 deal with the toy and robot arm examples, respectively.

9.1 Methodological setup

Both examples are based on a known smooth forward mapping $\mathbf{x} \xrightarrow{\mathbf{g}} \mathbf{y}$ where the $\mathbf{x}$-variables are restricted to a given domain. The experiment setup is the following:

- Generate a sample of data for the $\mathbf{x}$-variables, compute $\mathbf{y} = \mathbf{g}(\mathbf{x})$ and perturb with noise $\mathbf{e}$ to obtain $\mathbf{t} = \left(\begin{smallmatrix}\mathbf{x}\\ \mathbf{y}\end{smallmatrix}\right) + \mathbf{e}$. Permute at random the point indices $1, \dots, N$ to obtain shuffled data and avoid any effects related to sequential correlation (although this is not really necessary). We then have a sample $\{\mathbf{t}_n\}_{n=1}^{N'}$ that we use as training set for the density model.
- By sampling a trajectory in the $\mathbf{x}$-domain, generate a sequence $\{\mathbf{t}^{(n)}\}_{n=1}^{N}$ where continuity holds. By continuity we mean that the sampling rate is high enough (that the original trajectory could be recovered from the sequence). To do this, generate data for the $\mathbf{x}$-variables at a high enough rate and then compute $\mathbf{y} = \mathbf{g}(\mathbf{x})$. This gives a noiseless, continuous sequence. We can obtain a noisy, continuous sequence by adding a small amount of noise.
- Generate a missing data mask tailored to that sequence, that is, of size $N \times D$ (section 7.2.2). Seven different types of mask are considered to account for forward mapping approximation, mapping inversion and general missing data patterns, as described in fig. 9.1.
- Apply the missing data mask to the original, complete sequence to obtain an incomplete sequence. Reconstruct the incomplete sequence with our method and with other, standard methods and compare the results in terms of squared reconstruction error and by plotting the reconstructed sequences.

As density models we used a latent variable model trained on the sample $\{\mathbf{t}_n\}_{n=1}^{N'}$ (we discarded the latent variables and only used the distribution in observed space). The intrinsic dimensionality L was taken as the true one, equal to the number of $\mathbf{x}$-variables (strictly, equal to the number of $\mathbf{x}$- or $\mathbf{y}$-variables, whichever is smaller). We used two different models:

- Factor analysis, so that the density model is a Gaussian constrained to a linear mapping. This is used as a baseline for comparison.
- GTM, so that the density model is an isotropic Gaussian mixture constrained to a nonlinear mapping.

We used the continuity constraint $\mathscr{C}_1$, based on the unweighted Euclidean distance between consecutive points (table 7.2).

The methods compared are of three types (the identifiers in `typewriter` font face will be used for convenience in this and the next chapter):

This chapter is partly based on references Carreira-Perpiñán (1999b, 2000b).

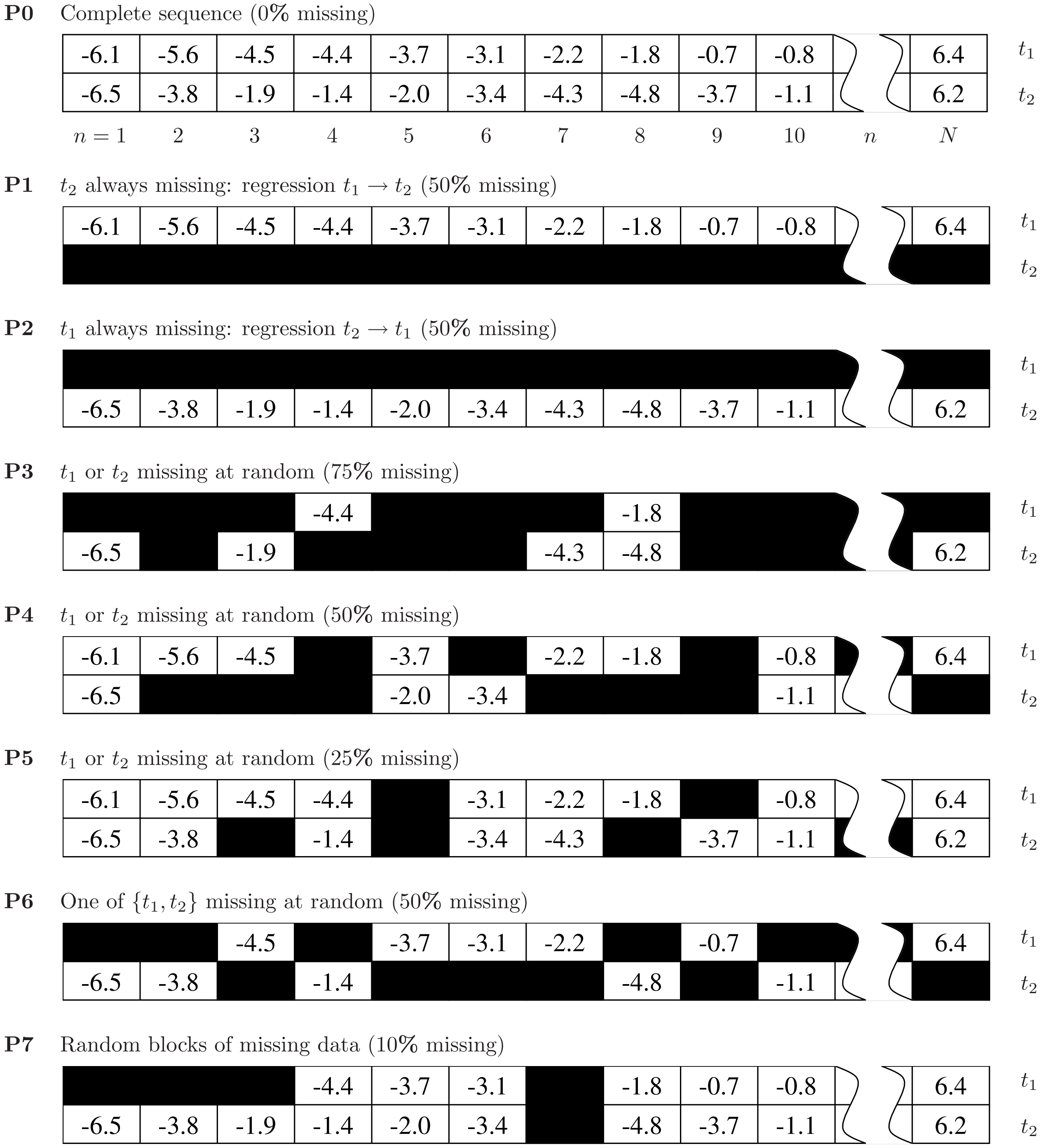


Figure 9.1: Masks used for the synthetic data examples. The variables mentioned (t_1, t_2) are for the toy example (section 9.2). The same scheme is used for the robot arm (section 9.3) and the articulatory-acoustic speech data of chapter 10 as follows: in P1, P2 and P6 replace t_1 with $\mathbf{x}$ and t_2 with $\mathbf{y}$ as a block; in P3–P5 and P7 replace t_1 and t_2 with any scalar variable. Thus, P1 means the regression $\mathbf{x} \to \mathbf{y}$, P2 the regression $\mathbf{y} \to \mathbf{x}$, P6 either $\mathbf{x}$ or $\mathbf{y}$ are missing (but not both) and P3–P5 and P7 are general, random missing data patterns, treating $x_1, \dots, x_{D_1}, y_1, \dots, y_{D_2}$ equally. "Missing at random" means here that the probability that variable $t_d^{(n)}$ is missing does not depend on the values of the other variables or on whether other variables are missing; in the parlance of the statistical treatment of missing data, this is MCAR (section 7.11.1.2). P7 represents the practically frequent case in engineering applications where a whole run of data goes missing (e.g. a scratch on a film or band-pass noise in speech).

- Based on **factor analysis**:

`fa` For which `mean` and the mode-based methods coincide.

- Based on **GTM**[1]:

`mean` Single pointwise reconstruction by the conditional mean for the density model used.

`gmode` Single pointwise reconstruction by the global mode of the conditional distribution for the density model used. The global mode is computed with our algorithm of chapter 8 for mode finding in Gaussian mixtures (specifically, the gradient-quadratic one, although the fixed-point one performed just as well). Because it finds all the modes, our algorithm guarantees that `gmode` really is the global mode—unlike other algorithms (section 7.3.3), where it usually is a random mode (as `rmode` below).

`rmode` Single pointwise reconstruction by a random mode of the conditional distribution for the density model used. All modes of the conditional distribution are taken equally likely.

`cmode` Single pointwise reconstruction by the closest mode of the conditional distribution for the density model used. The "closest mode" means the mode of the conditional distribution that is closest in Euclidean distance to the corresponding value of the original sequence. Thus, it gives the minimal reconstruction error (both pointwise and for the global reconstruction). It gives a lower bound of the reconstruction error achievable by any mode-based method (`gmode`, `rmode`, `grmode`, `dpmode`) and tells us how much usable reconstruction information is contained in the conditional modes. Of course, in a practical problem we do not know what mode is the closest one.

`grmode` Single pointwise reconstruction by the mode of the conditional distribution for the density model used that is closest in Euclidean distance to the previously reconstructed point, i.e., a greedy minimisation of the continuity constraint (section 7.6.4).

`dpmode` Multiple pointwise reconstruction by the modes of the conditional distribution for the density model used and dynamic programming minimisation of the continuity constraint to select the global reconstruction (section 7.6.3).

`sampdp` Multiple pointwise reconstruction by S samples of the conditional distribution for the density model used and dynamic programming minimisation of the continuity constraint to select the global reconstruction (section 7.3.7). We took S slightly larger than the maximal number of inverse branches of $\mathbf{g}^{-1}$ so that all the branches have a chance to contribute but without facilitating the appearance of outliers.

`meandp` A combination of `mean` and `dpmode`: multiple pointwise reconstruction by the modes of the conditional distribution for the density model used and dynamic programming minimisation of the continuity constraint to select the global reconstruction except that if the conditional distribution is unimodal, we use its mean rather than its mode. This is intended to account for skewed unimodal conditional distributions (e.g. figure 9.4(right)).

- Based on **multilayer perceptrons**. We trained five different MLPs, each with a single layer of sigmoidal hidden units but differing in the number of hidden units. The MLPs were trained to minimise the squared reconstruction error of the same training set used to estimate the density models, using stochastic gradient descent and small, random starting values for the weights[2]. By combining the outputs of these five MLPs in different ways we defined the following methods:

`mlpavg` Single pointwise reconstruction by the arithmetic mean.

`mlpdp` Multiple pointwise reconstruction by the five MLP outputs and dynamic programming minimisation of the continuity constraint to select the global reconstruction (i.e., like `dpmode` but using the MLP outputs instead of the conditional modes).

`mlpbest` The global reconstruction with minimal error of all five. Like `cmode`, this is a lower bound, generally unattainable in practice.

[1] We used Markus Svensén's Matlab implementation of GTM (available in the Internet at `http://www.ncrg.aston.ac.uk/GTM`) to estimate the GTM model parameters. We then developed our own Matlab software for construction of conditional distributions, exhaustive mode finding, constraint optimisation by dynamic programming and other operations required by the methods mentioned.

[2] We used the Netlab neural network software for Matlab, written by Ian Nabney and Christopher Bishop and freely available at `http://www.ncrg.aston.ac.uk/netlab`.

These methods are only applicable to regression problems (where the missing data pattern does not depend on the point index n): masks P1 and P2.

The number of hidden units h was chosen so that the number of weights was similar to the number of trainable parameters of the GTM model. If D_1 and D_2 are the dimensions of the $\mathbf{x}$- and $\mathbf{y}$-variables, respectively (with $D_1 + D_2 = D$, the dimension of the joint variables $\mathbf{t}$) then a D_1-h-D_2 MLP has $(D_1 + 1)h + (h + 1)D_2 \approx Dh$ parameters (weights and biases), while a GTM model with F radial basis functions (and biases) has $(F + 1)D + 1 \approx FD$ parameters. Thus we took h of the order of F.

Our a priori expectations are that `mean` and the MLP-based methods will perform similarly (see the discussion in section 7.3.6) and that `cmode` will outperform `gmode`, `rmode`, `grmode` and `dpmode` (by definition). The interesting results will be how much better `cmode` will be and how the mode-based methods will fare compared to `mean`, particularly `dpmode`.

9.2 2D toy example

9.2.1 Problem setup

We consider the forward mapping $g(x) \stackrel{\text{def}}{=} x + 3\sin x$ for $x \in [-2\pi, 2\pi]$, which results in one-dimensional data ($L = 1$) observed in $D = 2$ dimensions, i.e., $\mathbf{t} = (t_1, t_2)$ (fig. 9.2(left)). We generated a shuffled training set $\{\mathbf{t}_n\}_{n=1}^{N'}$ with $N' = 1\,000$ points sampled from the curve with additive $\mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I})$ noise for $\sigma = 0.2$ (fig. 9.2(centre)) and sample noisy (also with $\sigma = 0.2$) and noiseless trajectories to be reconstructed (e.g. fig. 9.2(right)). The forward mapping g is injective only in parts of the domain and so the inverse mapping g^{-1} is sometimes multivalued (specifically, it can take up to three values). The model parameters were taken as follows (the reader is referred to sections 2.6.1, 2.6.5 and 5.4 for details):

- Factor analysis: dimension of latent space $L = 1$, fig. 9.3(left).
- GTM: grid of $K = 200$ points (also $K = 20$ and 60) in a latent space of dimension $L = 1$, scaled to the $[-1, 1]$ interval and grid in the same square of $F = 9$ Gaussian basis functions of width equal to the separation between basis functions centres, fig. 9.3(right).
- MLP: one MLP with a single layer of $h = 48$ hidden units (mask P1) and five MLPs with a single layer of $h = 5, 15, 25, 35, 48$ hidden units, respectively (mask P2).

For `sampdp`, we generated $S = 6$ samples per conditional distribution.

In the following figures and tables we illustrate the performance of the methods in a gamut of situations, including reconstruction of noiseless and noisy trajectories of different lengths (N). Here is a little guided tour. Fig. 9.3 shows the density models: GTM perfectly approximates the data density and underlying low-dimensional manifold, unlike factor analysis. Figures 9.4 and 9.5 demonstrate the derivation of a univalued (forward) mapping and a multivalued (inverse) mapping, respectively, with the conditional distribution of the GTM model. Fig. 9.6 demonstrates the use of a continuity constraint to break the ambiguity of multiple pointwise reconstructions. Fig. 9.7 compares various methods with the forward mapping (mask P1): except for `fa`, all methods perform well. Fig. 9.8, for the inverse mapping (mask P2), shows the limitations of methods based on single pointwise reconstruction (`mean`, `gmode`, `mlp`), while `dpmode` still succeeds in recovering the original trajectory. The same happens with mask P4 in fig. 9.9. Fig. 9.10 shows for mask P2 (but the same happens for the other masks) that `grmode` and `sampdp` do not perform well. Table 9.1 gives the errors and lengths of the reconstructed trajectories for all methods and masks. The reader is encouraged to examine the figure captions and then proceed to the next section.

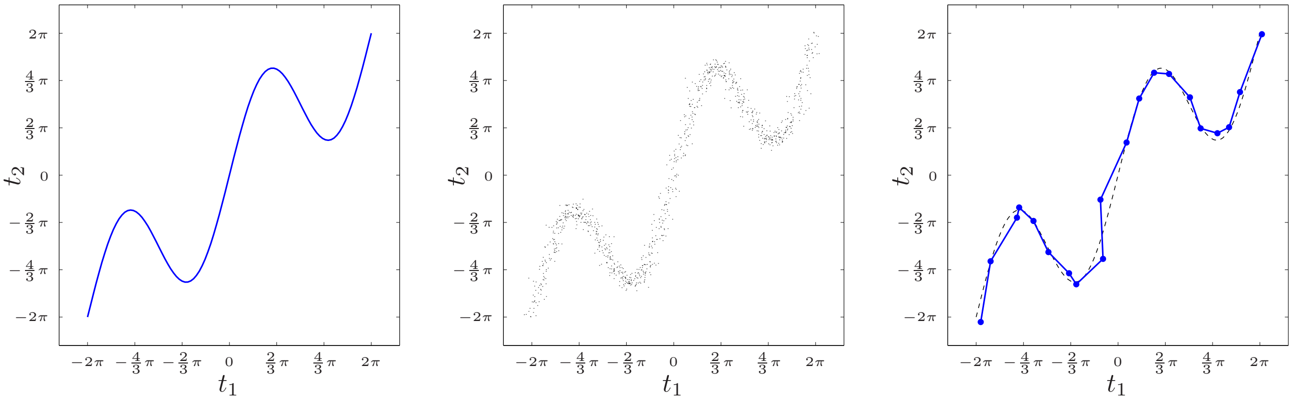


Figure 9.2: Data for the toy problem: univalued forward mapping $t_2 = g(t_1) = t_1 + 3\sin t_1$ with sometimes multivalued inverse $t_1 = g^{-1}(t_2)$. *Left*: curve $\mathbf{t} = (x, x + 3\sin x)^T$ for $x \in [-2\pi, 2\pi]$. *Centre*: point cloud $\{\mathbf{t}_n\}_{n=1}^{N'}$, sampled from the curve with additive $\mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I})$ noise for $\sigma = 0.2$ and $N' = 1\,000$. *Right*: sample noisy trajectory $\{\mathbf{t}^{(n)}\}_{n=1}^{N}$ with $N = 20$.

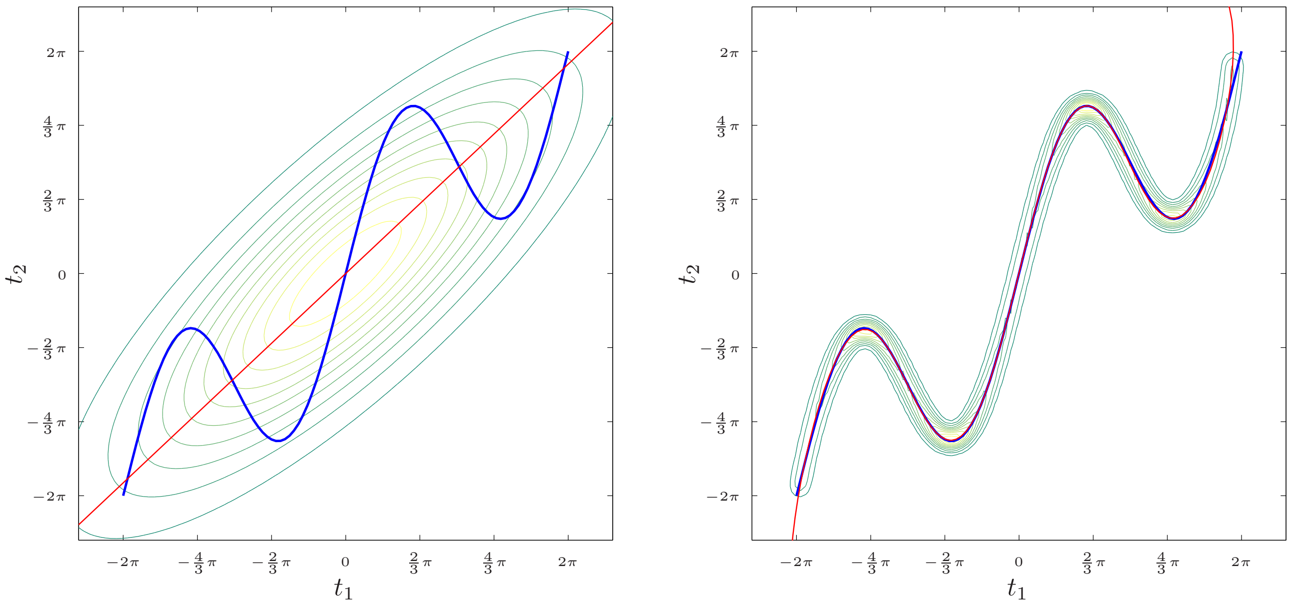


Figure 9.3: Density models used for the toy problem. Both figures show the density surface (contour lines), the trace of $g : t_1 \to t_2$ (blue) and the trace of the latent variable model mapping $\mathbf{f} : \mathbf{x} \to (t_1, t_2)^T$ (red). *Left*: factor analysis: $L = 1$, 6 parameters, log-likelihood $= -4\,807$, linear mapping $\mathbf{f}$, Gaussian density $p(\mathbf{t})$. *Right*: GTM: $L = 1$, $K = 200$, 21 parameters, log-likelihood $= -3\,109$, nonlinear mapping $\mathbf{f}$, Gaussian mixture density $p(\mathbf{t})$. One 48-hidden-unit MLP (145 parameters) was trained for the forward mapping (mask P1) and five MLPs with 5 (16), 15 (46), 25 (76), 35 (106) and 48 (145) hidden units (parameters), respectively, for the inverse mapping (mask P2).

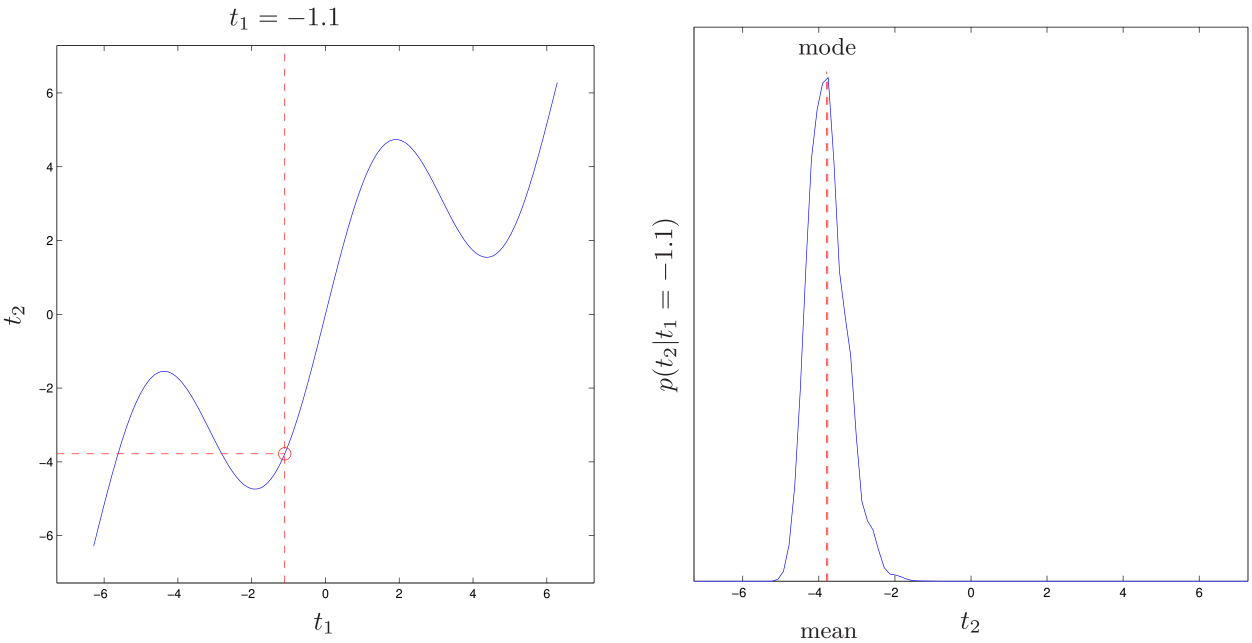


Figure 9.4: Demonstration of the derivation of functional relationships from conditional distributions: unimodal distribution. The conditional distribution $p(t_2|t_1)$ is unimodal (right graph; the scale of the vertical axis is irrelevant and thus omitted), so both the mean and the mode are good approximations to the real value. The conditional distribution is slightly skewed, so the mean and the mode do not coincide. The example is for $t_1 = -1.1$, for which $g(t_1) = -3.8$ (left graph); compare with fig. 7.4(b). In this and the following two figures the trace of the forward function g is drawn as a visual aid in the (t_1, t_2) plane; the values for t_1 or t_2 are always obtained from the conditional distribution.

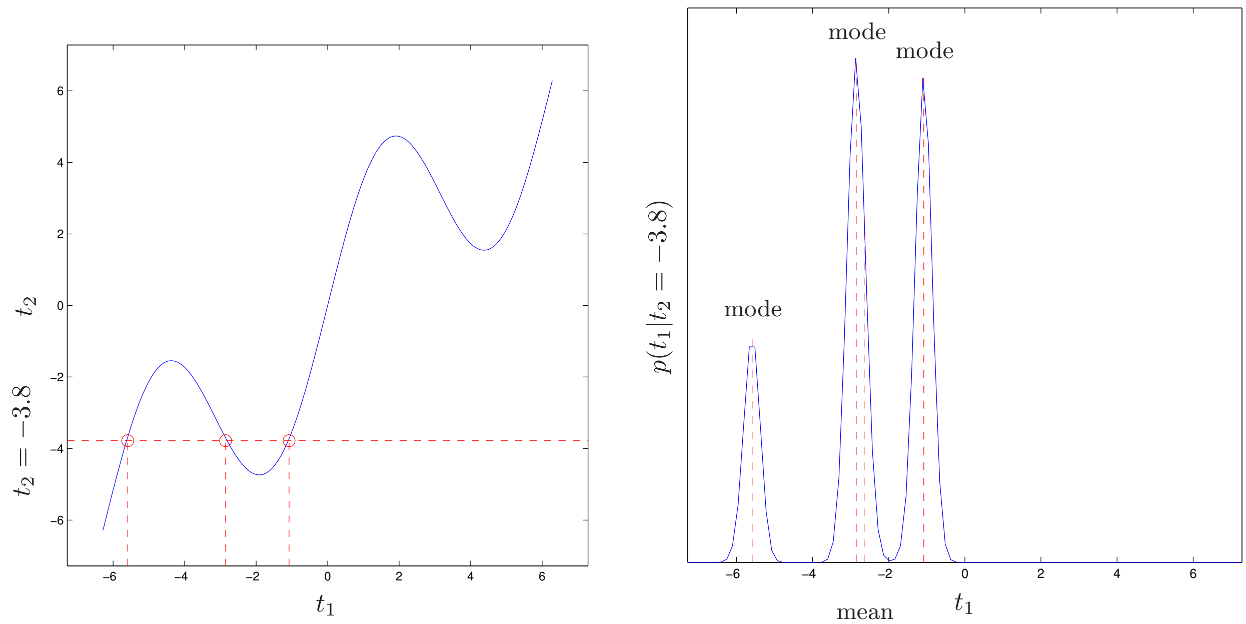
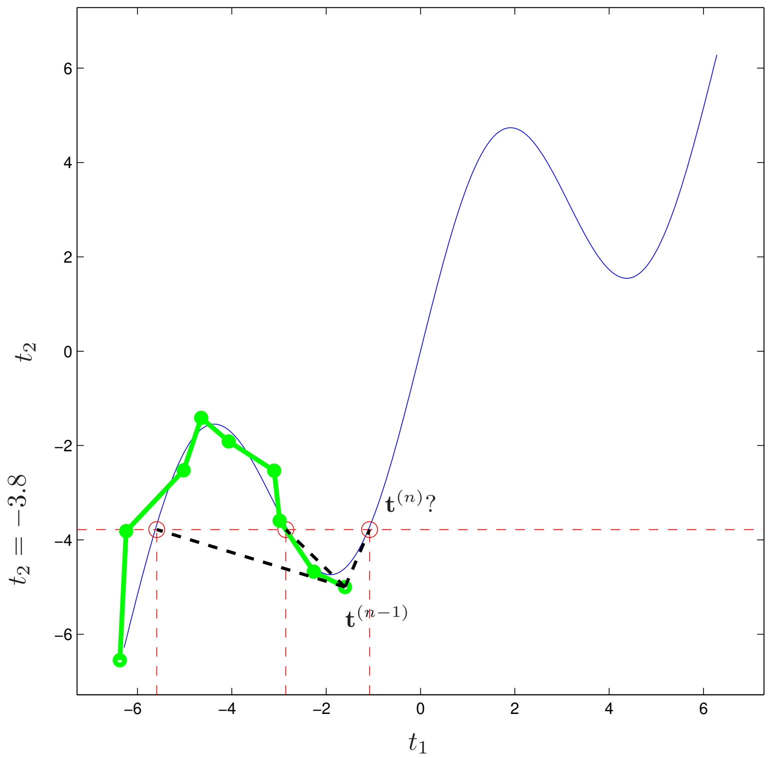


Figure 9.5: Demonstration of the derivation of functional relationships from conditional distributions: multimodal distribution. The conditional distribution $p(t_1|t_2)$ can be multimodal (right graph; the scale of the vertical axis is irrelevant and thus omitted), so one of the modes is the right one, but which one? We need extra information. The example is for $t_2 = -3.8$, for which $g^{-1}(t_2) = \{-5.6280, -2.8027, -1.1112\}$ (left graph); compare with fig. 7.4(d). Observe that the modes may have different heights and still correspond to valid inverse values.



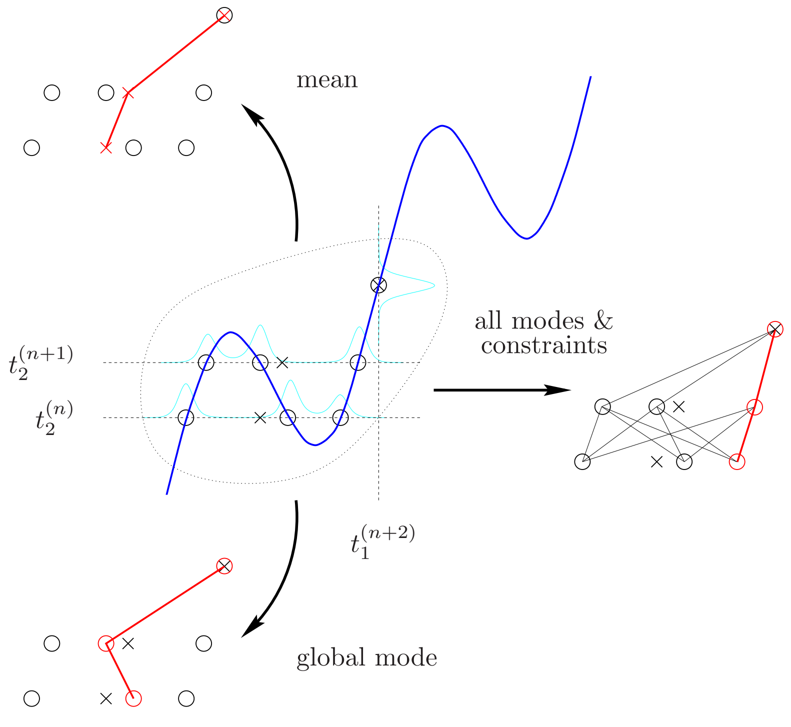


Figure 9.6: Demonstration of the use of a continuity constraint. *Top*: the case of fig. 9.5 with $t_2^{(n)} = -3.8$. If we knew that $(t_1^{(n-1)}, t_2^{(n-1)})^T = (-1.6, -5)^T$, then the most reasonable possibility would be $t_1^{(n)} = -1.1$, i.e., the closest one to $t^{(n-1)}$, rather than $t_1^{(n)} = -2.9$ or -5.6. More generally: of all the possible combinations (of reconstructed trajectories), pick the one that minimises a global constraint. *Bottom*: construction of the layered (sub)graph of fig. 7.8 for a trajectory fragment n, $n+1$, $n+2$ where $\{t_2^{(n)}, t_2^{(n+1)}, t_1^{(n+2)}\}$ are present and $\{t_1^{(n)}, t_1^{(n+1)}, t_2^{(n+2)}\}$ are missing and comparison between `mean`, `gmode` and `dpmode`. In each case, the red, thick trajectory is the reconstructed one.

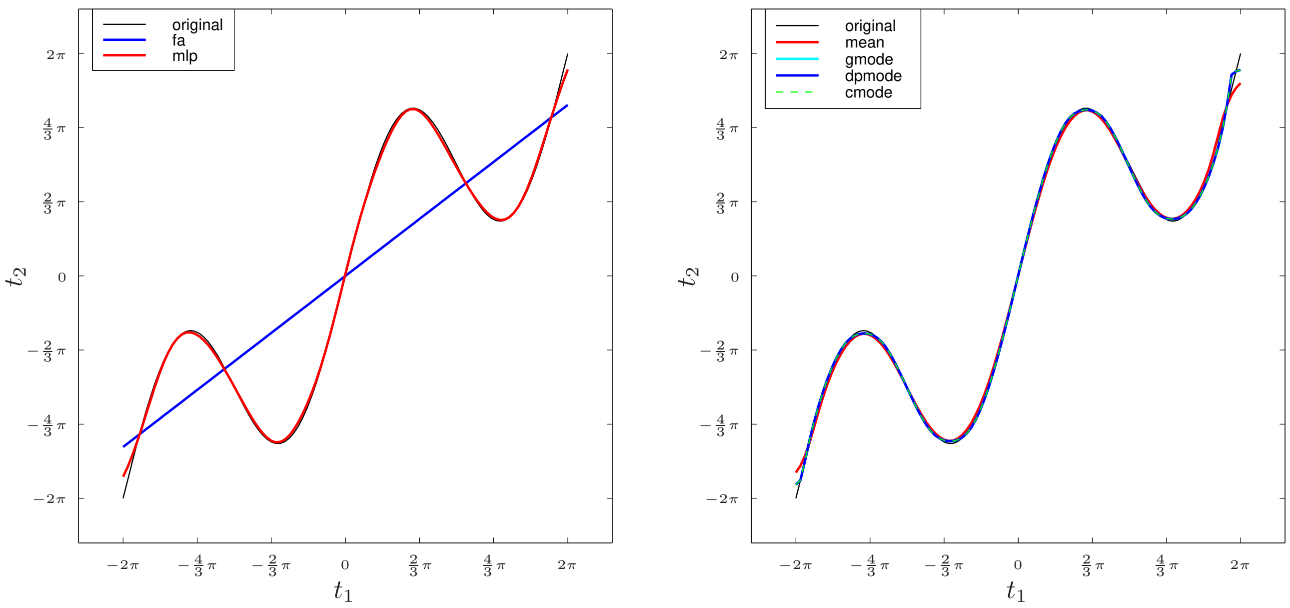


Figure 9.7: Reconstruction results for the toy problem: forward mapping (mask P1). The original trajectory is noiseless with $N = 100$ points and looks like the curve of fig. 9.2(left). *Left*: factor analysis and multilayer perceptron. *Right*: GTM with $K = 200$, various methods. Factor analysis gives the linear regression of t_2 on t_1 while the rest of the methods give the correct, nonlinear mapping.

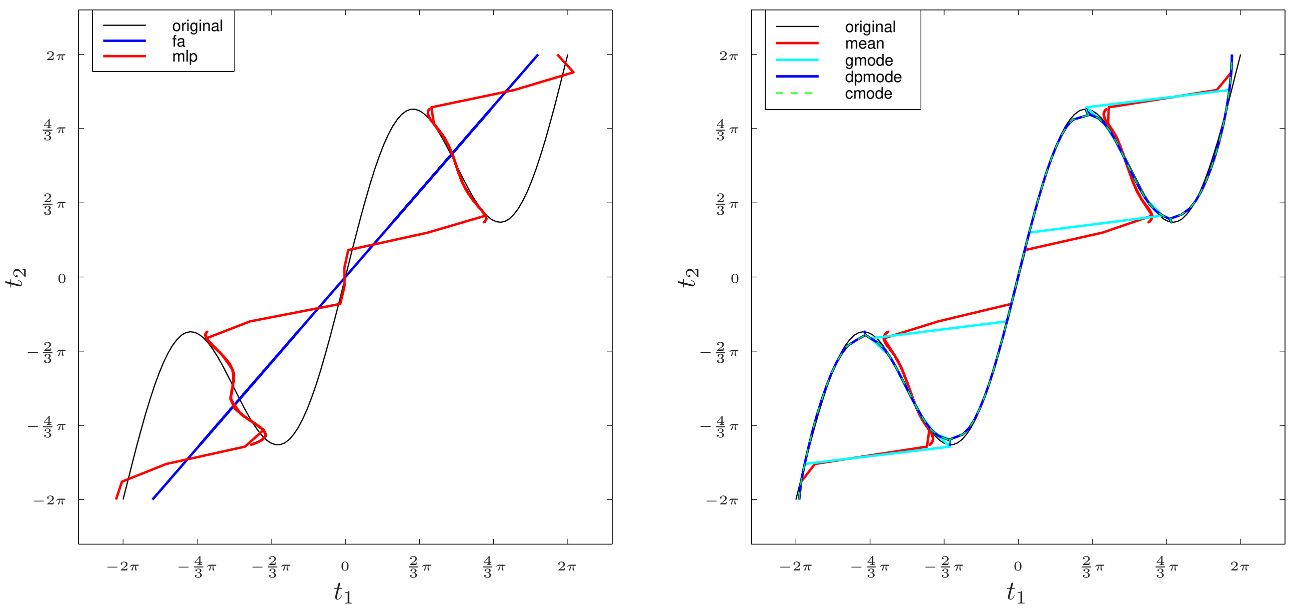


Figure 9.8: Reconstruction results for the toy problem: inverse mapping (mask P2). Original trajectory as in fig. 9.7. *Left*: factor analysis and multilayer perceptron. *Right*: GTM with $K = 200$, various methods. Factor analysis gives the linear regression of t_1 on t_2 (which has a different slope from that of t_2 on t_1); `gmode` always tracks some part of a branch but switches discontinuously between branches; `mean` and `mlp` coincide and give a continuous mapping that happens to track one branch due to the symmetry of the forward mapping g; `dpmode`, `cmode` and the original trajectory practically coincide. `mlp` means `mlpbest` (which corresponded to the MLP with 15 hidden units).

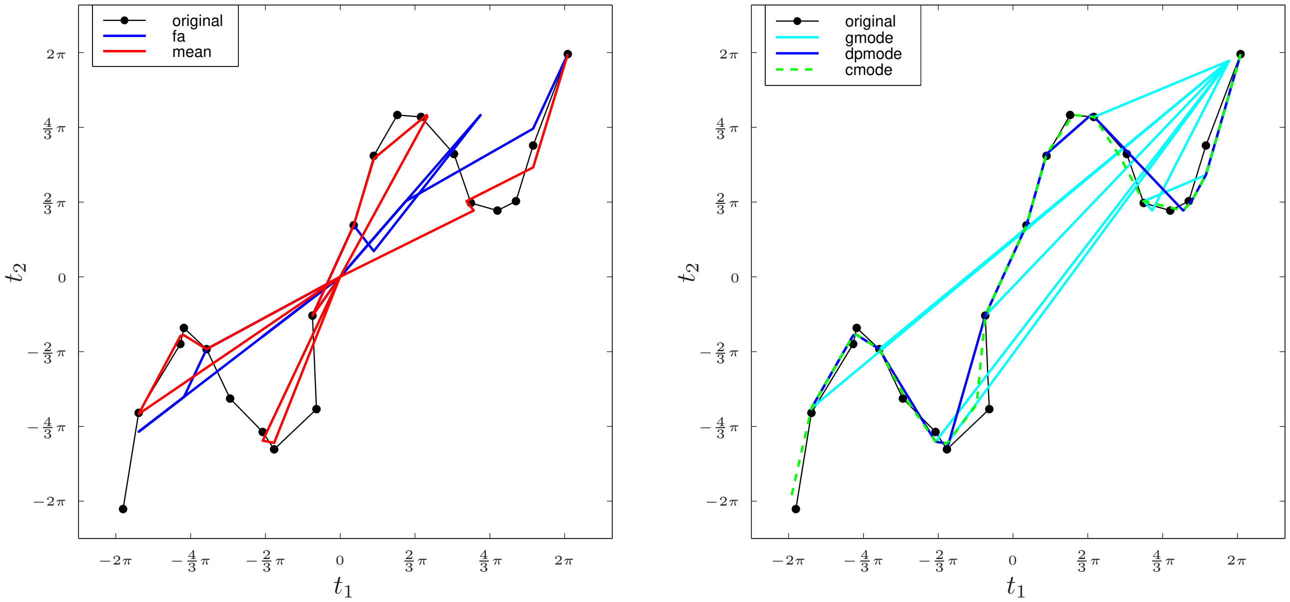


Figure 9.9: Reconstruction results for the toy problem: 50% missing data at random (mask P4). The original trajectory is noisy with $N = 20$ points, the same one as that of fig. 9.2(right). *Left*: factor analysis and $\mathrm{GTM}_{K=200}$ (`mean`). *Right*: $\mathrm{GTM}_{K=200}$ (`gmode`, `dpmode`, `cmode`). The multilayer perceptron cannot deal with varying patterns of missing data and so is not shown. The abrupt jumps to the global mean near $(0, 0)$ for `fa` and `mean` and to the global mode near $(2\pi, 2\pi)$ for `gmode` occur when both $t_1^{(n)}$ and $t_2^{(n)}$ are missing. `cmode` and the original trajectory practically coincide and `dpmode` recovers the original trajectory reasonably well.

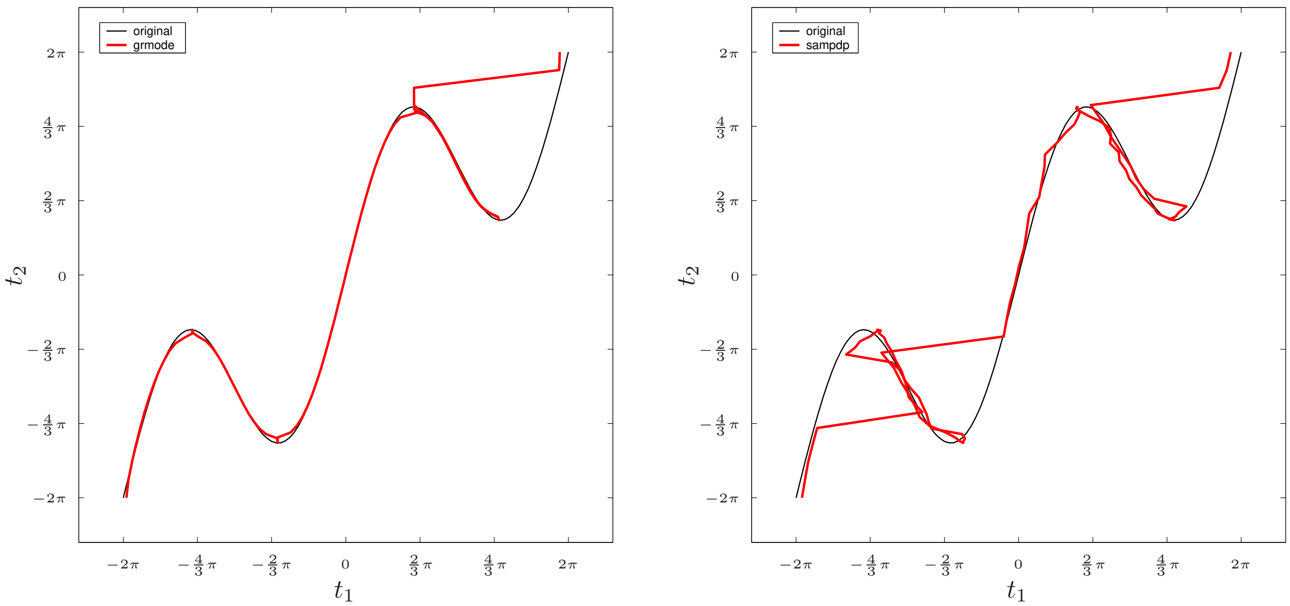


Figure 9.10: Reconstruction results for the toy problem: methods `grmode` and `sampdp`, inverse mapping (mask P2). Original trajectory as in fig. 9.7. *Left*: `grmode`. *Right*: `sampdp`. Both methods perform badly, with retracings and shortcuts on the original trajectory.

9.2.2 Main conclusions

The main result is that, for a good density model and if continuity holds, `dpmode` can greatly improve over the traditional methods `mean` and `mlp`, approaching the limit of `cmode` (which is close to zero error) for all patterns of missing data; and is particularly successful for general ones even for large amounts of missing data, poor density models or oversampled trajectories. This means that the modes contain important information about the possible options to predict the missing values, and that application of the continuity constraint allows to recover that information. Table 9.2 summarises the method comparison for each pattern of missing data.

Although the previous figures and tables are for particular cases, we ran the methods with many different training sets, trajectories and masks (all randomly generated) and there were no qualitative discrepancies. Based on them we can draw specific conclusions about the methods:

- `fa` is always much worse than `mean` for any mask, as expected, given that the forward mapping is nonlinear.
- MLP methods: for mask P2, `mlpbest` comes first (as it should, by its definition), closely followed by `mlpavg` and somewhat farther by `mlpdp`. Thus, there was no gain in using an ensemble where the members simply differed in the architecture. Clearly, we could have tried harder to get a good ensemble, with each member learning a different branch of the inverse mapping; but, as discussed in section 7.11.2.1, branch determination is a tough problem.

  For both masks P1 and P2, the MLP methods were practically equal to the `mean` (with a slightly larger difference in multivalued mappings due to local minima). This was expected from the equivalence between universal mapping approximators and conditional mean (section 7.3.6). However, variations can occur depending on the actual mapping approximator and the actual density model (e.g. our MLPs do better than the conditional mean of `fa` because the latter does not approximate well the true density).
- Since our GTM model is very close to the true density, the `mean` approximates extremely well the forward mapping (mask P1), being univalued. It fails with the inverse mapping, being multivalued: the univalued `mean` mapping travels through the midpoints of the inverse branches, blending them into a single branch. Because of the symmetry of the forward mapping, the midpoint of these branches always happens to coincide with one of the branches and so the result is better than it should be in a general case lacking symmetry (where the `mean` will not be a valid inverse). The `mean` also fails for general masks, although it is still the best[3] method based on single pointwise reconstruction (the others being `gmode` and `rmode`).
- Both `gmode` and `rmode` have a larger reconstruction error than `mean` on the average, as predicted by the theory (section 7.3.6). They result in discontinuous mappings, often switching branches frequently (specially `rmode`), but they always provide with valid inverses, unlike the `mean`, because they track branches. `gmode` generally outperforms `rmode`—the latter can be considered as the chance baseline for single pointwise reconstruction by the modes.
- `cmode` was defined as the minimal-error limit for `gmode`, `rmode`, `grmode` and `dpmode`. In fact, for all masks considered, it achieves practically zero reconstruction error, thus vastly outperforming the `mean` too (except, marginally, sometimes with mask P1, where `mean` is optimal). Even in shuffled data (where `dpmode` cannot work, since continuity is lost) `cmode` still contains information to get the lowest error of all methods (but that information cannot be recovered). This demonstrates that the modes of a good conditional distribution contain information that can potentially achieve near-zero reconstruction error.
- Much of this information is recovered by `dpmode`, which outperforms or equals any other method, including `mean`, for any mask. Even for the forward mapping, where `mean` is guaranteed to be optimal on the average, `dpmode` still performs as well as the `mean` (it actually outperforms it in table 9.1(top), but this is an isolated instance). Its performance is degraded only slightly for very high amounts of missing data (e.g. mask P3), where the other methods incur huge errors. For regression problems (masks P1 and P2), `dpmode` may perform worse than `mean` in two situations analysed below: nonsmooth density models and oversampled trajectories.
- There is virtually no difference between `meandp` and `dpmode`. This is due to the fact that the training set contained isotropic noise, so that when the conditional distribution is unimodal, it is approximately[4]

[3] It is the best method on the average for an ideally perfect model, but it is possible to find specific instances where it is not, e.g. it is slightly worse than `gmode` for masks P1 and P2 in table 9.1(top) and in fig. 9.18(right) `gmode` is slightly less biased than `mean` at the mapping turns.

[4] It does not have to be symmetric because, for example, the shape of the low-dimensional manifold can produce skewness (see section 9.2.5).

symmetric and its mean and mode nearly coincide. For more complex types of noise `meandp` may improve over `dpmode`.

- Both `grmode` and `sampdp` result most times in wrongly reconstructed trajectories that retrace themselves and contain shortcuts between branches. For `grmode` the reason is the inability to backtrack out of a wrong solution, as discussed in section 7.9.4, although for general missing data patterns its performance is not much worse than that of `dpmode`. For `sampdp` there are two reasons: the inability to find a priori a good value[5] for the number of samples S, so that suboptimal candidate reconstructions are generated and/or correct ones are missed; and the appearance of wrong trajectory reconstructions with low value for the global constraint (see also section 9.2.3). Therefore, despite the computational economy of these approaches, they are not recommended.
- We confirmed that shuffling the trajectory to destroy its continuity resulted in a large increase of the reconstruction error for the methods based on constraint minimisation (`dpmode`, `grmode`, `sampdp`, `meandp`) for all masks (except P1) but did not affect the other methods.

Two general properties arise in common for several methods:

Denoising For all methods based on conditional distributions of a density model (and also for MLPs for masks P1–P2) a noisy trajectory is reconstructed as a smooth trajectory (for mode-based methods, the smoother the density model the smoother the reconstruction, but not so much for the `mean`). This is particularly noticeable in fig. 9.17: compare the original trajectory (top, left) with the reconstructed one by `dpmode` (bottom, right). In fact, a large part of the reconstruction error is due to the noise in the original trajectory, which has been removed from the reconstructed one. The reason is that by reducing a conditional distribution to a point (single pointwise reconstruction) or a point per branch (multiple pointwise reconstruction) all variation is eliminated for the given values of the present variables. This is not to say that the noisy trajectory is reconstructed as the smooth, underlying manifold: bias may exist in the density model (e.g. see fig.9.12(left) and section 9.2.5).

Regression is harder than varying patterns of missing data For methods based on global constraint minimisation, such as `grmode` and in particular `dpmode`, a varying missing data pattern helps to break the ambiguity. The reason is the changing structure in observed space of the candidate reconstructions for varying patterns of missing data (see fig. 9.11). For mask P1 (top, right; assuming several modes, as in the nonsmooth models of section 9.2.3), the candidate reconstructions at point n form a vertical series (parallel to the t_2 axis). As n increases, consecutive series sweep across the (t_1, t_2) space in a nearly rigid motion. Thus, it is possible to have long runs of wrong candidate reconstructions that give a short trajectory segment; these segments may have long jumps where, for example, the conditional distribution becomes unimodal, but on the whole the wrong trajectory can be shorter than the correct one (see, for example, the runs $n = 5$–7 and $n = 9$–13). For mask P2 (bottom, left) the same occurs; the candidate reconstructions at point n form now a horizontal series. But for varying patterns of missing data (bottom, right) the spatial structure of these series typically changes dramatically from n to $n + 1$ (e.g. from horizontal to vertical). Thus, the runs of wrongly reconstructed points are much shorter and when concatenated they give a longer trajectory than the correct one. This is reinforced by having occasionally single candidates, e.g. when all variables are present. This can be seen in tables 9.1, 9.3 and 9.4 for the `dpmode`: large errors, associated with grossly wrong reconstructions, appear only for nonsmooth models or oversampling for the regression-type patterns (masks P1–P2), but never for general ones (P3–P7), even when as many as 76% of the values are missing. The generally unreliable method `grmode` also performs quite well: if it gets into the wrong branch or retraces itself, it is quickly corrected when the pattern of missing data changes.

Thus, the `dpmode` method is very robust for varying patterns of missing data even with not very good density models, oversampling or large amounts of missing data.

9.2.3 Smoothness

In the previous experiments we have used a nearly ideal density model (GTM with $K = 200$): it approximates the true density almost exactly and so any conditional distribution has the right number of modes and at the

[5]To force all mapping branches to be represented, we also tried a very high value $S = 100$. The resulting trajectories were smoother but still wrong.

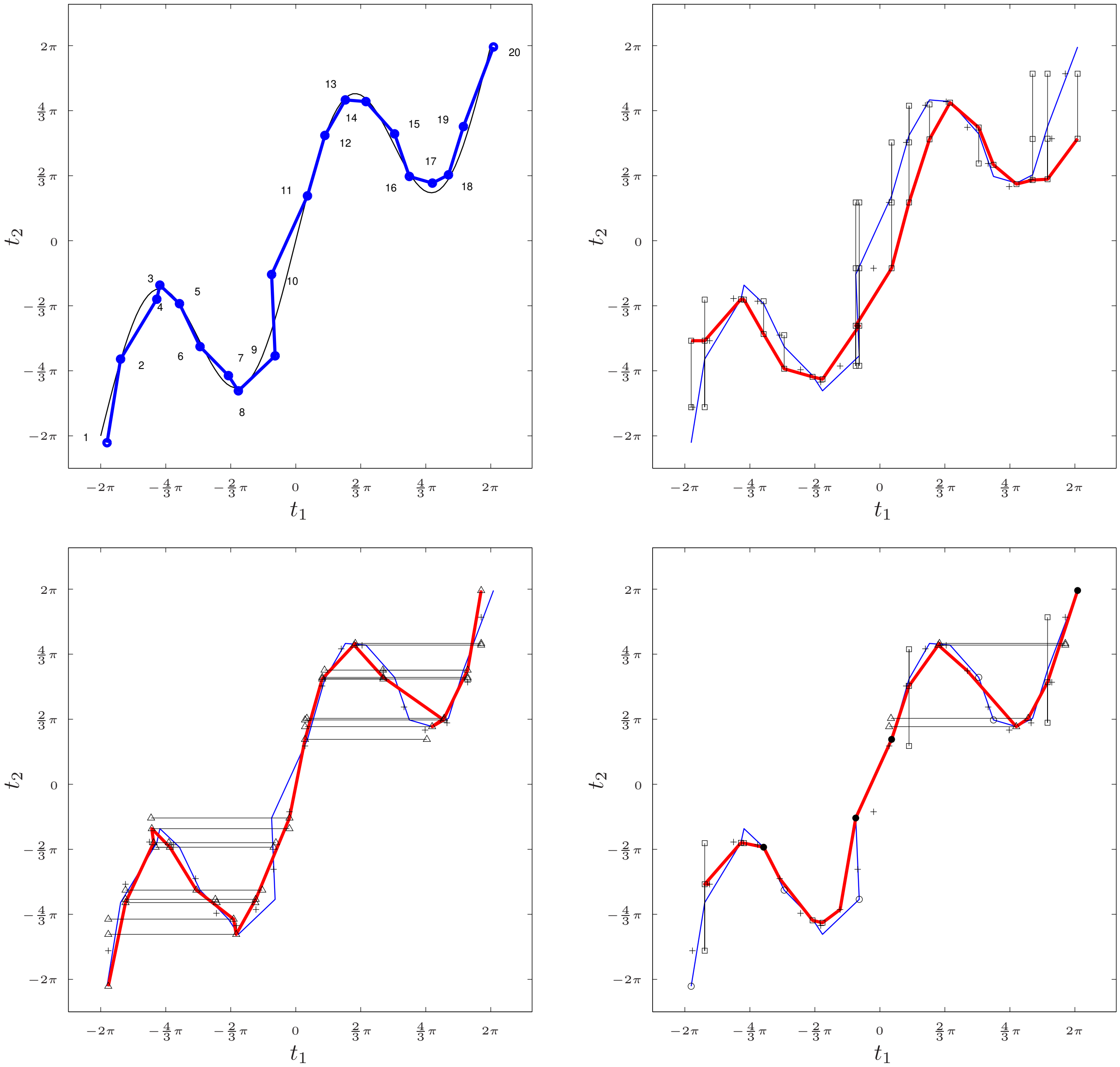


Figure 9.11: Reconstruction results for the toy problem: explicit display of candidate reconstructions for `dpmode` with masks P1, P2 and P4. *Top, left*: original trajectory (thick line), noisy with $N = 20$ points, the same one as that of fig. 9.2(right). The mapping g (thin line) and the point indices $n = 1, \dots, N$ are shown for reference. A low N was chosen to avoid cluttering the graph. For the remaining three graphs, the symbols used are as follows: a vertical series of squares ($\square$) joined by a line representing the modes of $t_2|t_1$ (i.e., t_1 present, t_2 missing); a horizontal series of triangles ($\triangle$) joined by a line representing the modes of $t_1|t_2$ (i.e., t_1 missing, t_2 present); a single filled circle ($\bullet$) representing a known point (both t_1 and t_2 present); a single empty circle ($\circ$) representing a point where both t_1 and t_2 are missing, whose candidate reconstructions are the K centroids of the Gaussian mixture (marked $+$). The reconstructed trajectory (thick line) and original trajectory (thin line) are shown. The model used was $\text{GTM}_{K=20}$ (fig. 9.12(left)). *Top, right*: reconstructed trajectory for mask P1. Observe how, due to spurious modes, the fact that $t_1^{(9)} \approx t_1^{(10)}$ triggers a run of wrongly reconstructed points till $\mathbf{t}^{(14)}$, where there is a single mode. *Bottom, left*: reconstructed trajectory for mask P2. *Bottom, right*: reconstructed trajectory for mask P4. Observe how the fact that for the first point $t_1^{(1)}$ and $t_2^{(1)}$ are both missing results in $\hat{\mathbf{t}}^{(1)}$ being reconstructed as the centroid closest to $\hat{\mathbf{t}}^{(2)}$. The reconstructed trajectory is different from that of `dpmode` in fig. 9.9(right) because there we used $\text{GTM}_{K=200}$.

Average squared error $\frac{1}{N}\sum_{n=1}^{N}\left\|\mathbf{t}^{(n)}-\hat{\mathbf{t}}^{(n)}\right\|^2$

Mask (% missing)	Factor analysis	MLP `mlpbest`	MLP `mlpavg`	MLP `mlpdp`	GTM with $K=200$ `mean`	`gmode`	`rmode`	`cmode`	`grmode`	`dpmode`	`sampdp`	`meandp`
P1(50%)	3.8011	0.0129	0.0129	0.0129	0.0196	0.0120	0.0120	0.0120	0.0120	0.0120	0.2027	0.0196
P2(50%)	4.2702	2.1475	2.1633	2.4746	2.1184	2.0878	7.7086	0.0129	0.7529	0.0129	2.0003	0.0129
P3(76%)	15.5385				14.6874	62.8203	36.1892	0.0069	0.2534	0.1936	5.9703	0.2059
P4(56%)	11.4116				9.7848	31.4508	14.8545	0.0058	0.1512	0.0746	4.6827	0.0867
P5(25%)	2.9049				1.7891	6.5224	3.1629	0.0040	0.0191	0.0066	0.6252	0.0062
P6(50%)	4.1377				0.9555	0.8104	4.7174	0.0122	0.0122	0.0122	0.2423	0.0178
P7(8%)	1.8476				1.5252	5.0694	2.5553	0.0007	0.0050	0.0029	0.0726	0.0027

Value of constraint $\mathscr{C}_1=\sum_{n=1}^{N-1}\left\|\hat{\mathbf{t}}^{(n)}-\hat{\mathbf{t}}^{(n+1)}\right\|$ (original sequence: 29.35)

Mask (% missing)	Factor analysis	MLP `mlpbest`	MLP `mlpavg`	MLP `mlpdp`	GTM with $K=200$ `mean`	`gmode`	`rmode`	`cmode`	`grmode`	`dpmode`	`sampdp`	`meandp`
P1(50%)	15.85	28.00	28.00	28.00	27.42	28.53	28.53	28.53	28.53	28.53	26.11	27.42
P2(50%)	33.53	41.71	39.87	35.42	38.75	44.66	184.38	29.62	33.55	29.62	43.28	29.62
P3(76%)	245.66				260.70	499.68	502.31	29.36	29.85	28.65	159.11	28.54
P4(56%)	233.33				222.44	352.79	312.27	29.90	30.32	28.23	136.56	28.27
P5(25%)	166.73				100.87	157.51	133.56	29.83	30.49	29.82	62.60	29.78
P6(50%)	170.26				77.35	64.16	226.32	30.21	30.21	30.21	41.87	29.71
P7(8%)	95.95				78.32	108.56	90.45	29.40	29.54	29.40	36.46	29.39

Table 9.1: Reconstruction results for the toy problem: average squared error and global constraint value (trajectory length) for a noiseless sequence $\{\mathbf{t}^{(n)}\}_{n=1}^{N}$ with $N = 100$ points that looks like the curve of fig. 9.2(left). $\{\hat{\mathbf{t}}^{(n)}\}_{n=1}^{N}$ is the reconstructed trajectory by the corresponding method. For mask P1, `mlpbest` = `mlpavg` = `mlpdp` since only one MLP was used. For mask P2, `mlpbest` corresponded to the MLP with 15 hidden units. For masks P3–P7, MLPs are not applicable.

Problem type (mask)	Low error $\longrightarrow$ High error
Forward (P1)	cmode = dpmode = mean = mlp = gmode = rmode < fa
Inverse (P2)	cmode $\lesssim$ dpmode < mean = mlp < gmode < fa < rmode
General (P3–P7)	cmode < dpmode $\ll$ mean $\ll$ gmode = rmode = fa

Table 9.2: Reconstruction results for the toy problem: summary comparison of the methods in terms of reconstruction error. This table was obtained from the analysis of reconstruction error tables like 9.1 for many different, random training sets, trajectories and masks (which did not differ much from 9.1).

right locations (but see section 9.2.5). It is clear that a poor density model will degrade the performance the method to some extent, so we need to ensure that we estimate a good enough model by appropriately choosing the model type, number of parameters, training algorithm, etc., just as we would do with any other algorithm.

However, since the modes are sensitive to the shape of the density function, problems may arise even with a density estimate that is qualitatively good: Gaussian mixtures, being a superposition of localised bumps, have a tendency to develop ripple on an otherwise smooth function, as we demonstrate here for the mask P1 (forward mapping).

Figure 9.12(left) shows a GTM model of $K = 20$, which results in a Gaussian mixture of only 20 circular components. The density estimate is worse than that of fig. 9.3(right) in terms of log-likelihood but still represents well the density. What the $K = 20$ model has lost is smoothness: the mixture components are quite separated from each other compared to their widths and do not coalesce enough. This results in wavy conditional distributions having more modes than they should, such as that shown in fig. 9.12(right). Those modes do not necessarily have a low probability value, so they cannot be easily removed by means of a rejection threshold.

How does this affect the trajectory reconstruction? Each, or some, of the true modes along the trajectory has unfolded into a few modes scattered in a small area around the true mode. A very good reconstruction is still possible since some of these modes are very close to the true one. This is evidenced by the low reconstruction error of the `cmode` in table 9.3(top), very similar to that of table 9.1(top), using $K = 200$. The `mean` also achieves a reconstruction error about as low as with $K = 200$, being largely insensitive to the ripple. But the error for `dpmode` is now 1.1226 for $K = 20$ and 0.4049 for $K = 60$, while it was 0.0120 for $K = 200$; observe that the values for $\mathscr{C}_1$ are now smaller. The problem is that this crowd of spurious modes may well allow wrong reconstructed trajectories that have a lower global constraint value (that are shorter) due to *shortcuts* that appear as horizontal and nearly vertical segments in fig. 9.13. The parameter that governs this behaviour is the ratio between the extent of the mode scatter inside a conditional distribution and the sampling period of the trajectory: the larger the scatter, the more likely interference becomes with neighbouring trajectory points. Fig. 9.14 shows the variation of the reconstruction error as a function of the trajectory index n. The large reconstruction errors for `dpmode` (see also fig. 9.13) appear in those areas where the number of modes is large: roughly, the intervals $[-2\pi, -4.5]$, $[-1.5, 1.5]$ and $[4.5, 2\pi]$ along the t_1 axis (compare also with fig. 9.12(left)). The `mean` and `cmode` only show a tiny increase in error in those intervals.

What patterns of missing data are affected? From the reconstruction error for the different masks, table 9.3(top), we can see that the problem with `dpmode` seems confined to the forward mapping (mask P1): the errors for the inverse mapping (mask P2) and for general missing data patterns (masks P3–P7) are barely larger than those of table 9.1(top). This can be explained as follows:

- The likely reason why `dpmode` does not suffer from a nonsmooth model for mask P2 is the particular geometry of the forward mapping chosen: t_2 varies much faster than t_1 almost everywhere[6] and so, when consecutive Gaussian centroids are widely separated from each other, they are nearly stacked parallel to the t_2 axis (e.g. in the interval $[-1.5, 1.5]$ in fig. 9.12(left)). This results in spurious modes in $p(t_2|t_1)$ but not in $p(t_1|t_2)$, as can be observed in fig. 9.15. Therefore, in general the `dpmode` should also suffer from nonsmooth models for mask P2.

[6]Concretely, for $\mathbf{t} = (x, x + 3\sin x)^T$ we have $\left|\frac{dt_1}{dx}\right| = 1$ and $\left|\frac{dt_2}{dx}\right| = |1 + 3\cos x|$ and so t_2 varies faster than t_1 except for $t_1 \in [-\frac{3\pi}{2}, -3.98] \cup [-2.3, -\frac{\pi}{2}] \cup [\frac{\pi}{2}, 2.3] \cup [3.98, \frac{3\pi}{2}]$ where $2.3 = \arccos(-\frac{2}{3})$ and $3.98 = -2.3 + 2\pi$. This is a small region, only 23% of $[-2\pi, 2\pi]$, the domain of t_1. Besides, precisely in those intervals the Gaussian centroids are densely packed because the training set used has more points there (since we sampled t_1 uniformly) and so no spurious modes appear there either.

- For general missing data patterns (masks P3–P7), the explanation is as in the previous section: the subsets of missing variables usually change from point n to point $n+1$ and thus the probability of getting a run of several points whose conditional distributions have spurious modes decreases. That is, $\mathbf{t}^{(n)}_{\mathcal{M}^{(n)}}|\mathbf{t}^{(n)}_{\mathcal{P}^{(n)}}$ may have spurious modes, e.g. if in that region $\mathbf{t}^{(n)}_{\mathcal{M}^{(n)}}$ vary faster than $\mathbf{t}^{(n)}_{\mathcal{P}^{(n)}}$; but $\mathbf{t}^{(n+1)}_{\mathcal{M}^{(n+1)}}|\mathbf{t}^{(n+1)}_{\mathcal{P}^{(n+1)}}$ may not have because $\mathcal{M}^{(n)} \neq \mathcal{M}^{(n+1)}$ most times. Therefore, in general the `dpmode` should not suffer from nonsmooth models for masks P3–P7.

Also, the lack of smoothness does not affect every single conditional distribution (e.g. in fig. 9.12(left), $p(t_2|t_1 = \frac{4}{3}\pi)$ remains unimodal, as it should be). The effect is confined to areas of space where the density is lower (the underlying mapping is more stretched out) and mixture components are more separated from each other with spurious modes arising. In fig. 9.12(left) these areas are the intervals mentioned before.

Do separate, full covariance parameters help? One might think that Gaussian components with separate, full-covariance parameters would adapt better to the true density shape by aligning themselves to the local structure of the low-dimensional manifold and setting their widths independently from each other. However, spurious modes arise again where the components join, as fig. 9.16 demonstrates. A further problem of a different kind altogether is that training Gaussian mixtures with separate covariances for each component is much more difficult than having shared covariances because of the singularities of the log-likelihood surface. Using the Netlab package, we trained Gaussian mixtures with an EM algorithm; the initial values for the centroid locations were obtained with the k-means algorithm. Except for mixtures with very few components ($M < 10$), EM almost always converged to a singularity of the log-likelihood: a component sits on top of a training set point and its covariance matrix goes to zero, increasing the log-likelihood without bounds. Simple heuristics, like resetting to some value the covariance matrix and centroid location of a component if it becomes singular, either did not succeed in avoiding singularities or resulted in very poor density estimates. The estimated models of fig. 9.16 were obtained by using a careful initialisation: the centroids were distributed uniformly along the low-dimensional manifold and the covariance matrices were set to the true noise covariance of the data ($\sigma^2\mathbf{I}$ for $\sigma = 0.2$, fig. 9.2(centre)). We also observed, by visually inspecting the conditional distribution, that there seems to be an optimal number of components that obtains the smoothest model. For our case, it was between $M = 10$ and $M = 20$; for $M = 5$ the conditional distribution was typically bimodal and for $M = 50$ it had many spurious modes. Thus, in practice and for high-dimensional cases, it may be difficult to obtain an accurate, smooth model using a Gaussian mixture with a separate covariance parameter for each component. Still, a possible advantage is that, for the same number of parameters, a mixture of full-covariance Gaussians will have fewer components than a mixture of spherical Gaussians. So, while the resulting pdf may not be smoother, the number of spurious modes—limited by the number of components—may be smaller.

Conclusion If the density model is not smooth, the conditional distribution presents spurious modes which may give rise to wrong solutions of the dynamic programming search. In this case, the reconstruction error with `dpmode` for regression problems (P1, P2) usually exceeds that of the conditional mean. For general patterns of missing data (P3–P7) the error increase is small. The `mean` is barely affected in any case. Removing low-probability modes does not help in general. Section 7.9.1 suggests some possible solutions for the lack-of-smoothness problem.

9.2.4 Over- and undersampling

We experimented with very small and very large values of the sampling rate of the trajectory for method `dpmode`. A very small sampling rate is one close to the Nyquist rate (around $\frac{1}{\pi}$ along the t_1-axis, or $N \approx 4$, in our case); a very large sampling rate is one whose period is much smaller than the noise (normal with $\sigma = 0.2$, or say $N \approx 200$, in our case). That is, we used very small and very large values of N while keeping the start and end points of the trajectory near $[-2\pi, -2\pi]$ and $[2\pi, 2\pi]$, respectively, as in fig. 9.2(right). The results were as follows.

Undersampling For $N = 20$, `dpmode` still reconstructs well the trajectory, but for very small rates, e.g. $N = 10$, `dpmode` starts finding wrong reconstructed trajectories, particularly for the worse GTM models ($K = 20$). This is clearly due to a lack of enough information to reconstruct the trajectory.

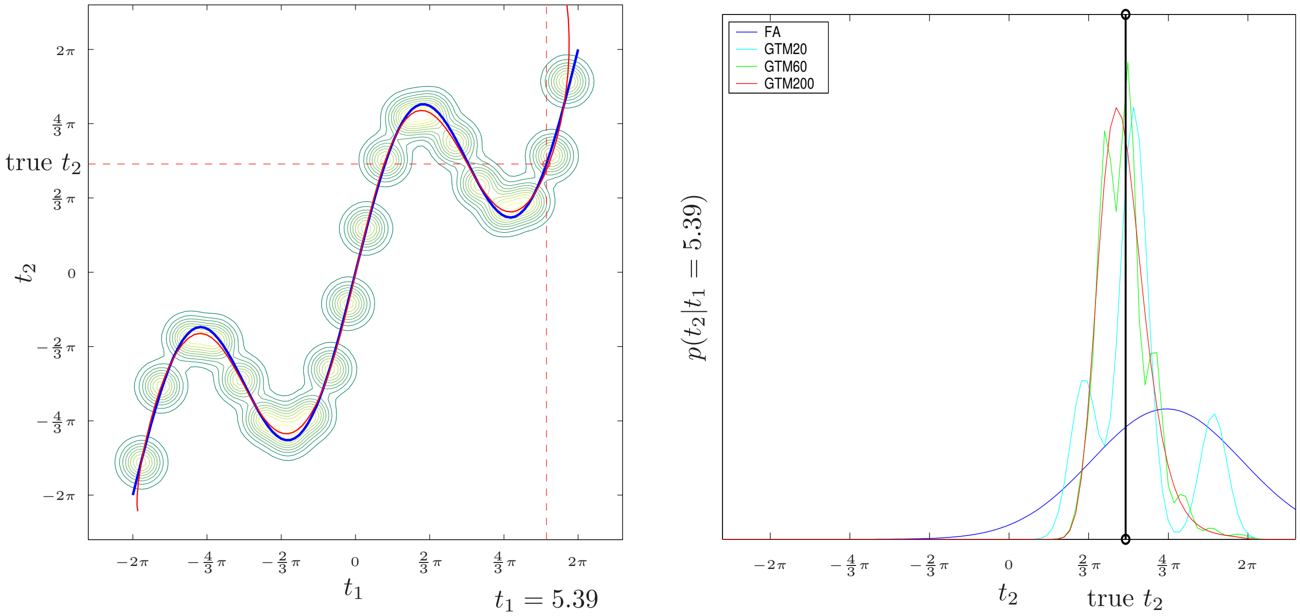


Figure 9.12: Effect on the conditional distribution of a nonsmooth GTM density model. *Left*: GTM model as in fig. 9.3(right) with $L = 1$, $K = 20$, 21 parameters, log-likelihood $= -3\,468$. The mixture components do not interact much at some places. *Right*: conditional distribution $p(t_2|t_1 = 5.39)$ for several models. The conditional distribution should have a unique mode at the location marked; the ideal model $\mathrm{GTM}_{K=200}$ achieves this, but as the number of mixture components K decreases, the unimodal distribution spreads and develops a ripple. This results in more modes than should really exist, scattered around the location of the true mode; while spurious, these modes may have a high probability value. The factor analysis conditional distribution is always smooth and unimodal, but wrong.

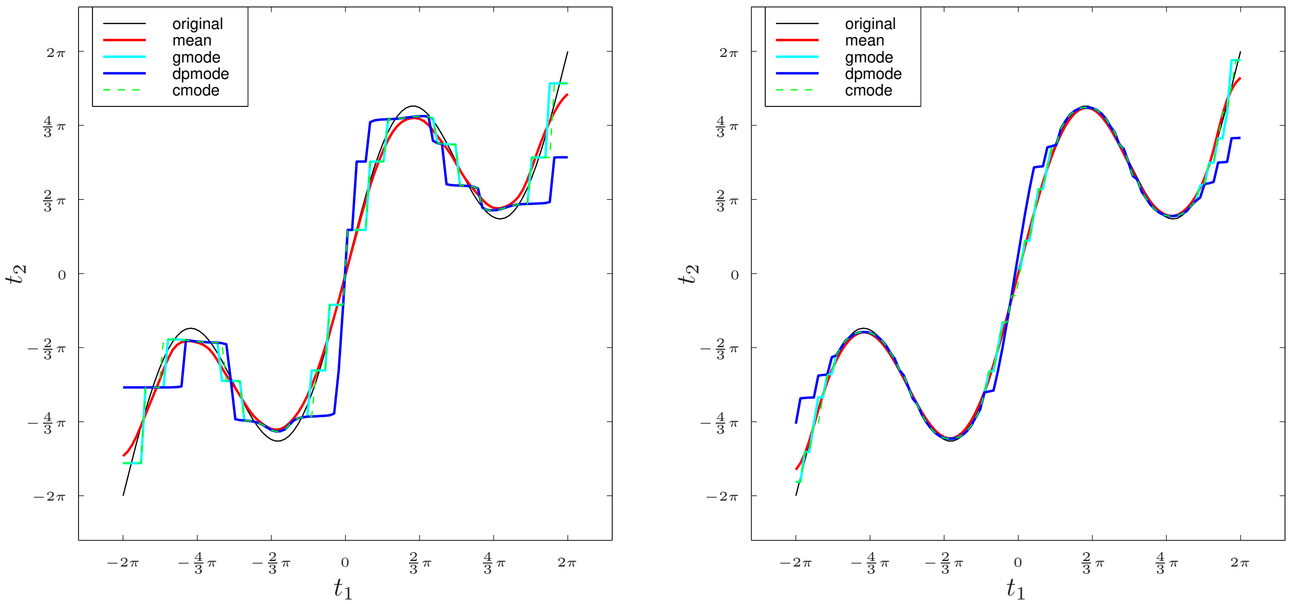


Figure 9.13: Reconstruction results for nonsmooth GTM density models: forward mapping (mask P1). Compare with fig. 9.7(right). *Left*: GTM with $L = 1$, $K = 20$, 21 parameters, log-likelihood $= -3\,468$, various methods. *Right*: GTM with $L = 1$, $K = 60$, 21 parameters, log-likelihood $= -3\,120$, various methods. For low K, the loss of density smoothness in some areas of data space results in quantisation errors. `dpmode` degrades more than the other methods because the increase of spurious modes enables the appearance of short trajectories with larger reconstruction error.

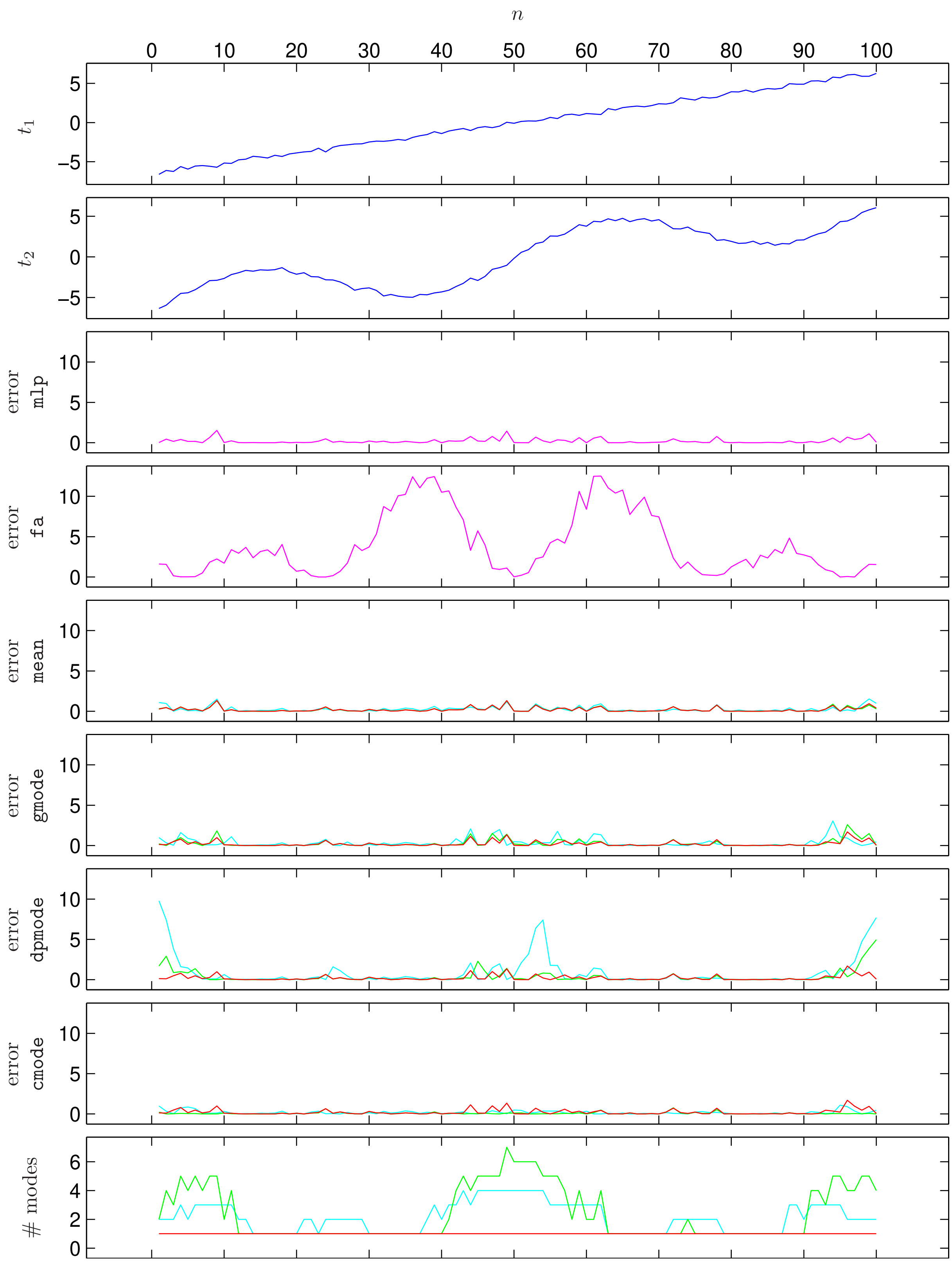


Figure 9.14: Reconstruction error for nonsmooth GTM density models as a function of the sequence index n: forward mapping (mask P1). The data are for a noisy trajectory of $N = 100$ points. *Top two graphs*: $t_1^{(n)}$ and $t_2^{(n)}$ for the original trajectory. *Bottom graph*: number of modes of the conditional distribution $t_2^{(n)}|t_1^{(n)}$. *Rest of graphs*: reconstruction error $\left\|\mathbf{t}^{(n)} - \hat{\mathbf{t}}^{(n)}\right\|^2$ for the respective method. GTM models: $K = 20$ (cyan), $K = 60$ (green) and $K = 200$ (red). The large errors in `dpmode` are confined to specific areas where the number of modes is high for $K = 20$ and 60.

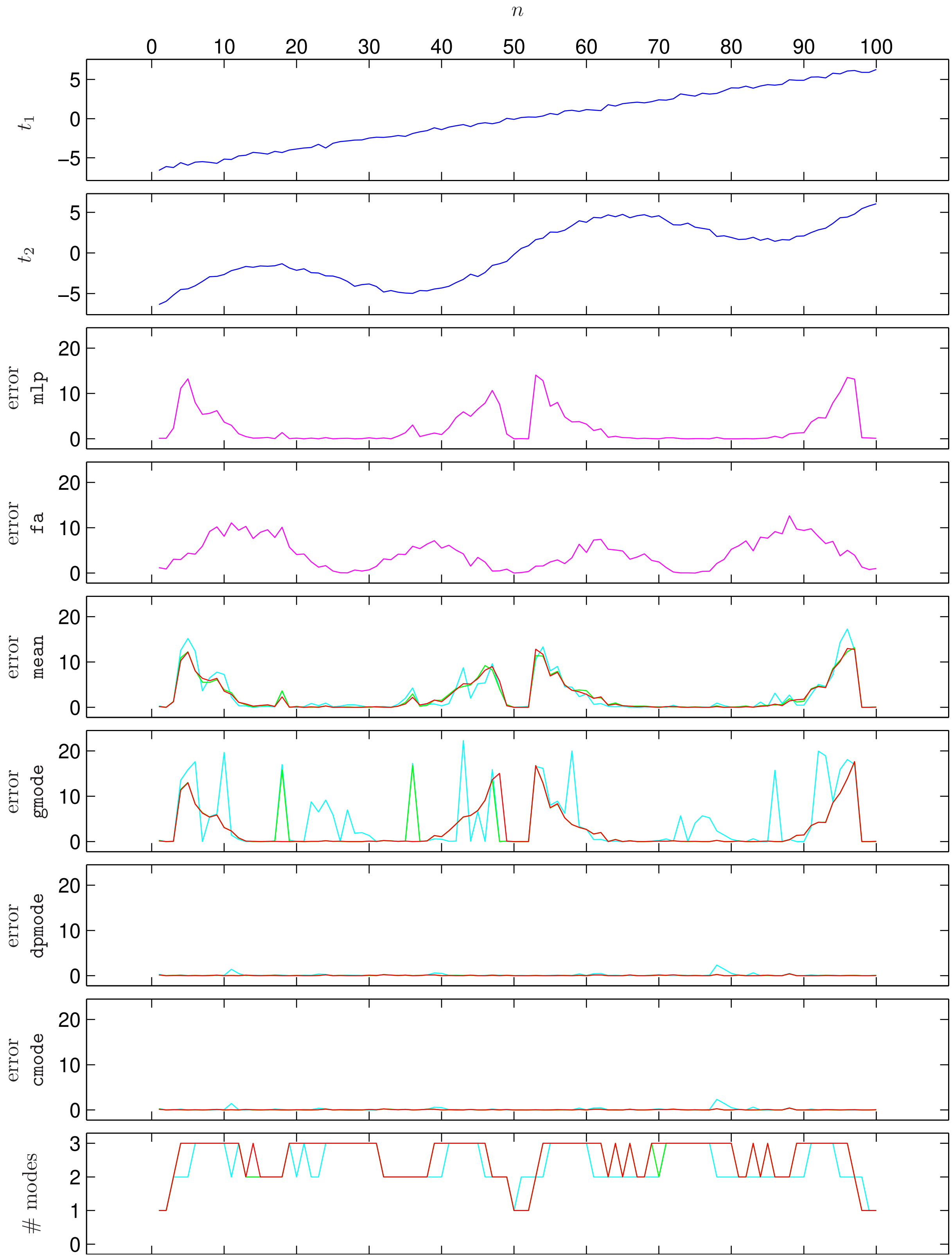


Figure 9.15: Reconstruction error for nonsmooth GTM density models as a function of the sequence index n: inverse mapping (mask P2). The data are for the same trajectory of fig. 9.14, but note the different scale in the vertical axes. Graphs as in fig. 9.14 except for *bottom graph*: number of modes of the conditional distribution $t_1^{(n)}|t_2^{(n)}$. GTM models: $K = 20$ (cyan), $K = 60$ (green) and $K = 200$ (red). The number of modes remains correct (1 to 3 depending on the number of branches), unlike in fig. 9.14 (where there should be always only one, but in some places there were up to 7). This is an artefactual effect due to the idiosyncrasies of the forward mapping and the training set used. In general, the `dpmode` may have large errors for mask P2 in some areas of the space.

Average squared error $\frac{1}{N}\sum_{n=1}^{N}\left\|\mathbf{t}^{(n)}-\hat{\mathbf{t}}^{(n)}\right\|^2$

Mask (% missing)	GTM with $K=20$								GTM with $K=60$							
	`mean`	`gmode`	`rmode`	`cmode`	`grmode`	`dpmode`	`sampdp`	`meandp`	`mean`	`gmode`	`rmode`	`cmode`	`grmode`	`dpmode`	`sampdp`	`meandp`
P1(50%)	0.0956	0.1659	2.0207	0.1609	2.1708	1.1226	0.3732	1.1338	0.0186	0.0261	0.4234	0.0183	0.7655	0.4049	0.2035	0.4037
P2(50%)	2.2376	3.3735	8.1796	0.1048	3.8005	0.1093	2.1427	0.1094	2.1561	2.2564	4.8978	0.0171	0.7595	0.0171	1.3864	0.0171
P3(76%)	14.8596	58.0260	31.1647	0.1566	1.0583	0.2272	8.5859	0.2285	14.7026	62.6840	24.7689	0.0200	0.3549	0.1164	3.2333	0.1250
P4(56%)	9.7731	28.7667	21.0178	0.1168	0.5550	0.1181	2.5967	0.1198	9.7969	31.3769	17.2085	0.0184	0.4607	0.1344	2.3082	0.1354
P5(25%)	1.8337	6.1947	7.9984	0.0510	0.0984	0.0510	0.6293	0.0517	1.7963	6.5038	2.9059	0.0075	0.0530	0.0075	0.3818	0.0078
P6(50%)	1.1561	1.6307	3.3248	0.1261	0.3900	0.1350	0.5803	0.1389	0.9572	0.9752	4.0446	0.0212	0.0997	0.0406	0.1829	0.0408
P7(8%)	1.5348	4.7046	2.4713	0.0124	0.0220	0.0124	0.2547	0.0132	1.5319	5.0580	3.8420	0.0013	0.0041	0.0013	0.2613	0.0011

Value of constraint $\mathscr{C}_1=\sum_{n=1}^{N-1}\left\|\hat{\mathbf{t}}^{(n)}-\hat{\mathbf{t}}^{(n+1)}\right\|$ (original sequence: 29.35)

Mask (% missing)	GTM with $K=20$								GTM with $K=60$							
	`mean`	`gmode`	`rmode`	`cmode`	`grmode`	`dpmode`	`sampdp`	`meandp`	`mean`	`gmode`	`rmode`	`cmode`	`grmode`	`dpmode`	`sampdp`	`meandp`
P1(50%)	25.04	31.05	107.12	31.04	26.86	26.74	27.12	26.29	27.44	29.96	49.32	29.86	26.58	26.34	25.90	26.01
P2(50%)	53.25	111.52	213.24	33.94	35.48	33.93	46.50	33.92	43.02	45.12	179.68	30.02	33.94	30.02	40.29	30.02
P3(76%)	259.56	478.92	555.13	37.73	42.12	33.90	161.56	33.92	261.58	498.99	492.10	29.57	33.29	29.16	140.82	29.07
P4(56%)	221.31	332.63	375.82	36.09	42.12	35.76	97.25	36.02	222.59	352.74	383.20	30.10	32.65	28.38	115.31	28.29
P5(25%)	102.65	157.68	240.67	35.64	37.68	35.64	60.72	35.74	102.16	157.32	149.64	29.96	32.17	29.96	47.03	29.94
P6(50%)	85.09	92.18	177.95	44.19	50.42	43.20	56.61	43.79	79.63	71.42	204.97	31.05	33.02	30.70	34.77	30.42
P7(8%)	79.23	106.05	103.92	31.03	32.40	31.03	46.34	31.26	78.73	108.44	111.49	29.43	29.78	29.43	43.12	29.41

Table 9.3: Reconstruction results for the toy problem with a nonsmooth density model: average squared error and global constraint value (trajectory length) for a noiseless sequence $\{\mathbf{t}^{(n)}\}_{n=1}^{N}$ with $N=100$ points, exactly the same as that of table 9.1 (the values for `fa`, `mlpbest`, `mlpavg` and `mlpdp` are as in that table and are thus omitted). $\{\hat{\mathbf{t}}^{(n)}\}_{n=1}^{N}$ is the reconstructed trajectory by the corresponding method. The results are for two GTM models of $K=20$ and 60, respectively.

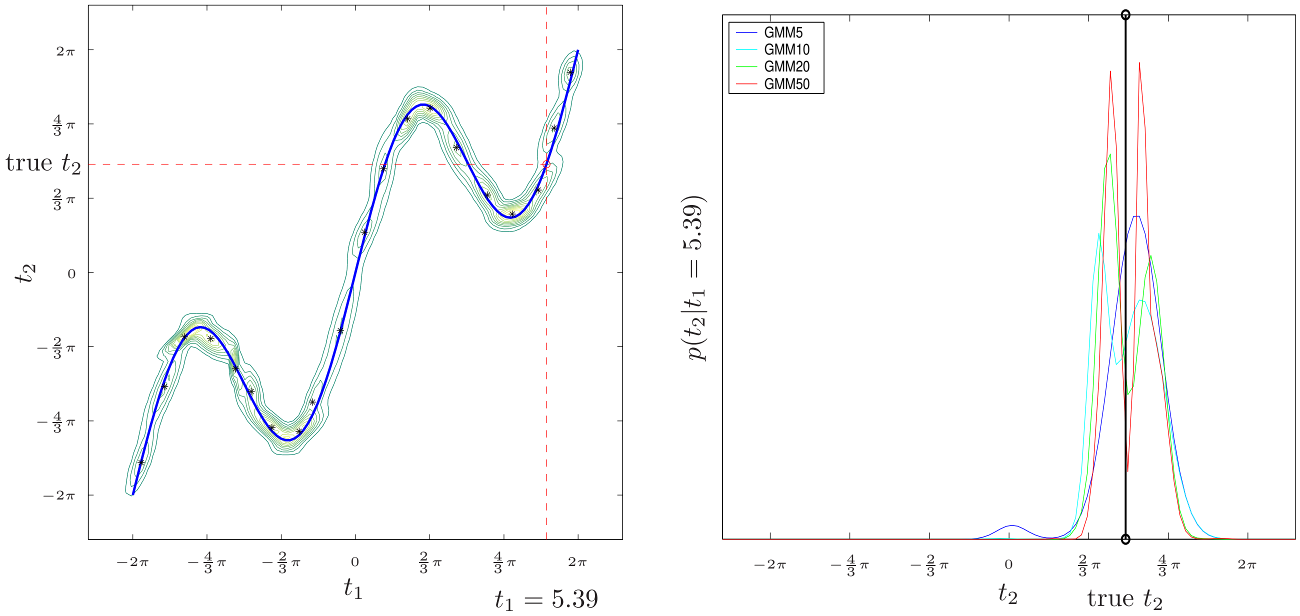


Figure 9.16: Gaussian mixtures with components with separate, full covariance parameters also give nonsmooth densities. Compare with fig. 9.12. *Left*: Gaussian mixture of $M = 20$ components: 119 parameters, log-likelihood $= -3\,053$. *Right*: the conditional distribution $p(t_2|t_1 = 5.39)$ (the same value of t_1 as in fig. 9.12(right)) for various mixtures is multimodal, but it should have a unique mode at the location marked. The results presented correspond to mixtures with different numbers of components: $M = 5$ (29 parameters, log-likelihood $= -3\,404$), $M = 10$ (59 parameters, log-likelihood $= -3\,159$), $M = 20$ (119 parameters, log-likelihood $= -3\,053$) and $M = 50$ (299 parameters, log-likelihood $= -2\,894$). They were trained on the 1 000-point set of fig. 9.2(centre) with the EM algorithm with carefully chosen starting parameters (the density models obtained with a k-means initialisation were quite worse).

Oversampling For very large rates, e.g. $N = 1\,000$, `dpmode` can give wrongly reconstructed trajectories that retrace themselves and have shortcuts for masks P1 and P2 although it still reconstructs well for P3–P7. Table 9.4 gives the reconstruction errors for all GTM models and all masks and fig. 9.17 shows some representative cases:

- For the forward mapping (mask P1), reconstruction remains good for smooth models; but for nonsmooth ones (such as the case $K = 60$ shown in fig. 9.17(top, right)) the existence of spurious modes introduces quantisation errors and allows the appearance of low-$\mathscr{C}_1$ trajectories with characteristic horizontal segments, as in fig. 9.13. That is, a shorter trajectory is obtained by moving along one axis for a while (retracing the horizontal segments several times) and then switching to another segment than by moving in all directions while slowly drifting on. These quantisation errors only occur in the region where spurious modes appear (footnote 6).

- For the inverse mapping (mask P2), `dpmode` breaks down independently of the smoothness of the density model (fig. 9.17(bottom, left)), i.e., even with the ideal model GTM$_{K=200}$, which provides all the right candidate reconstructions and only them. In terms of trajectory length, it pays better to spend more time retracing in one branch and then switching occasionally to another branch than to faithfully remain on the correct branch at each time.

- For general missing data patterns (masks P3–P7), even for high amounts of missing data (76% for P3, fig. 9.17(bottom, right)), `dpmode` reconstructs the trajectory very well independently of the model smoothness. The explanation is again the variation of missing variables subsets from point to point, which makes wrong reconstructions be longer than the correct one.

In summary, oversampling seems: (1) not to affect the `dpmode` for general missing data patterns (for both smooth and nonsmooth density models); (2) to introduce quantisation errors for forward (univalued) mappings but only in some areas, with the overall reconstruction being correct (the smoother the model, the lower the error); and (3) to severely degrade the quality of the reconstruction for inverse (multivalued) mappings due to shortcuts and retracings (for both smooth and nonsmooth density models).

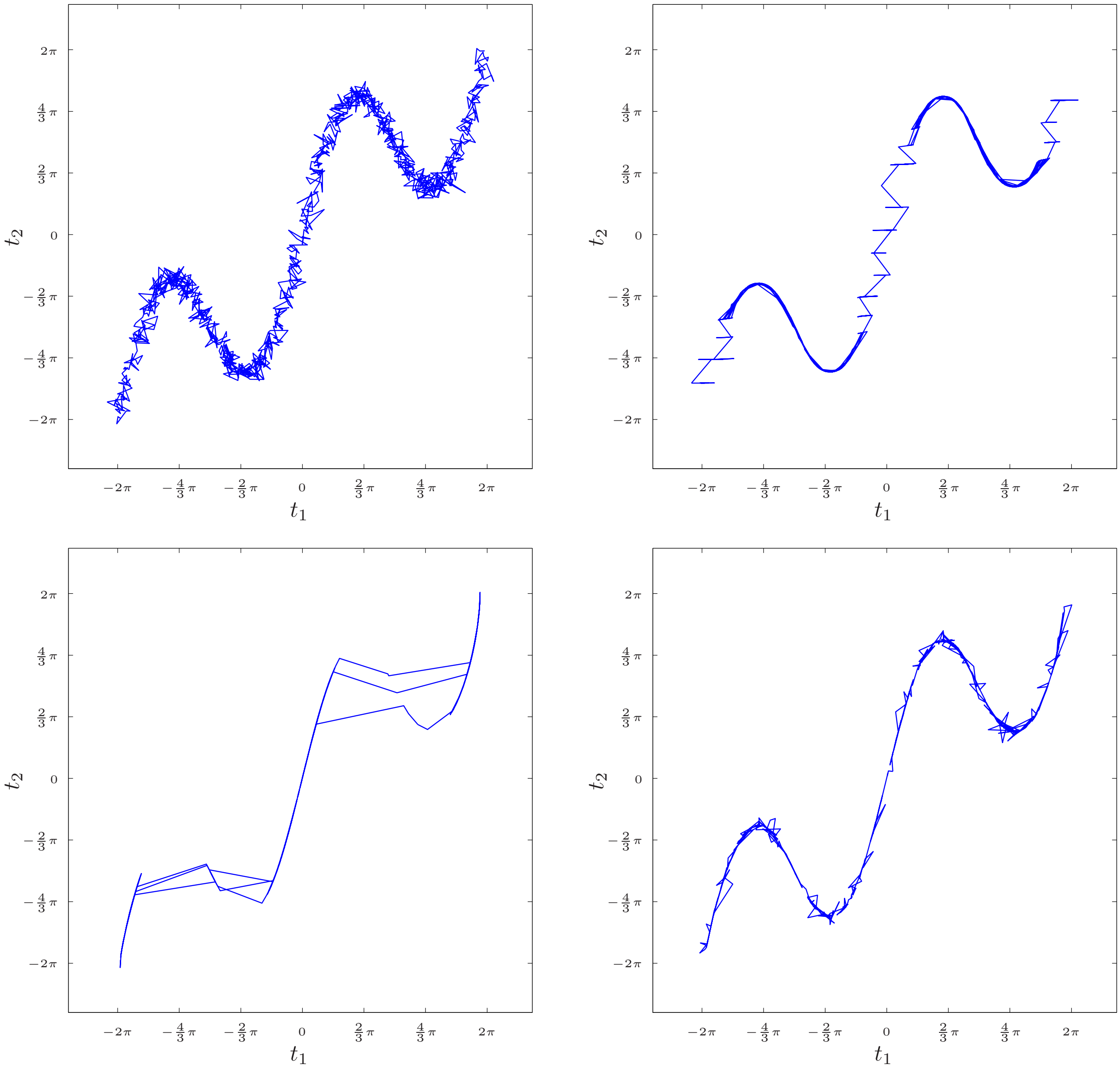


Figure 9.17: Reconstruction results for an oversampled trajectory with `dpmode`. *Top, left*: original trajectory with $N = 1\,000$ points. The trajectory resembles a random walk superimposed on a slow drift along the data manifold. *Top, right*: forward mapping (mask P1) with $\mathrm{GTM}_{K=60}$. The horizontal segments are retraced many times, while the diagonal links between them are typically taken only once. *Bottom, left*: inverse mapping (mask P2) with $\mathrm{GTM}_{K=200}$. The arcs contained in a branch of g^{-1}, such as the one for $t_1 \in [-\frac{\pi}{3}, 0]$ and $t_2 \in [-\frac{\pi}{3} - \frac{3\sqrt{2}}{2}, 0]$, are retraced many times while the shortcuts between them are typically taken only once. *Bottom, right*: 76% missing values at random (mask P3) with $\mathrm{GTM}_{K=200}$. The spike-like structures occur in the few cases when both t_1 and t_2 are present; otherwise, the modes of the conditional distribution always fall on the underlying manifold.

Average squared error $\frac{1}{N}\sum_{n=1}^{N}\left\|\mathbf{t}^{(n)}-\hat{\mathbf{t}}^{(n)}\right\|^2$

Mask (% missing)	GTM with $K=20$		GTM with $K=60$		GTM with $K=200$	
	`mean`	`dpmode`	`mean`	`dpmode`	`mean`	`dpmode`
P1(50%)	0.2673	0.6488	0.2202	0.1918	0.2205	0.2314
P2(50%)	2.5027	5.8372	2.3177	9.8833	2.3179	8.9765
P3(76%)	14.2588	0.2769	14.1821	0.1273	14.1855	0.1772
P4(56%)	7.1915	0.1627	7.1526	0.0820	7.1381	0.1104
P5(25%)	1.9719	0.0884	1.9232	0.0393	1.9286	0.0720
P6(50%)	1.3260	0.1720	1.2067	0.0904	1.2087	0.1605
P7(8%)	0.6926	0.0212	0.6746	0.0087	0.6719	0.0100

Value of constraint $\mathscr{C}_1=\sum_{n=1}^{N-1}\left\|\hat{\mathbf{t}}^{(n)}-\hat{\mathbf{t}}^{(n+1)}\right\|$ (original sequence: 349.73)

Mask (% missing)	GTM with $K=20$		GTM with $K=60$		GTM with $K=200$	
	`mean`	`dpmode`	`mean`	`dpmode`	`mean`	`dpmode`
P1(50%)	430.64	241.28	470.46	312.65	470.32	489.63
P2(50%)	519.35	275.04	389.94	256.96	357.30	253.92
P3(76%)	2357.47	236.92	2361.82	165.31	2360.02	174.98
P4(56%)	2186.05	354.09	2158.10	271.71	2155.54	300.27
P5(25%)	1120.76	400.35	1097.67	330.81	1098.67	373.81
P6(50%)	904.32	390.59	844.79	312.94	839.76	412.54
P7(8%)	418.42	341.72	415.49	337.94	413.13	338.13

Table 9.4: Reconstruction results for the toy problem with an oversampled trajectory: average squared error and global constraint value (trajectory length) for a noisy sequence $\{\mathbf{t}^{(n)}\}_{n=1}^{N}$ with $N=1\,000$ points shown in fig. 9.17(top, left). Only the values for `dpmode` are given, as well as those of `mean` to represent the baseline of the optimal single pointwise reconstruction method. $\{\hat{\mathbf{t}}^{(n)}\}_{n=1}^{N}$ is the reconstructed trajectory by the corresponding method. The results are for three GTM models of $K=20$, 60 and 200, respectively.

The main reason is that the original trajectory is polygonally very long[7] (high $\mathscr{C}_1$) but is not long in terms of actual displacement—being a random walk superimposed on a slow drift, it twists around itself many times in a region of size σ. Thus, if there are multiple pointwise candidate reconstructions, there often exist shorter trajectories containing multiple retracings of a branch segment and infrequent branch switchings. The quality of the density model is not really at fault here: it is a characteristic of the global constraint chosen. Generally speaking, length minimisation algorithms often result in zig-zag, jagged patterns (e.g. the elastic net, section 4.9.2) which may or may not be desirable.

Interestingly, we found that the trajectories were correctly reconstructed if we used the squared Euclidean distance instead of the Euclidean distance in the value of $\mathscr{C}_1$.

The example shows clearly the denoising property of the method: the reconstructed trajectory is more similar to the underlying manifold than to the original, noisy trajectory (the more so the smoother the density model is; compare with the smooth bits in fig. 9.17(top, right) for $K=60$). Note how the average reconstruction error is larger for the noisy trajectory in table 9.4 than the corresponding error for the noiseless trajectory in in table 9.1. A small spike in the tips of the turns is noticeable, as in fig. 9.8(right) or in fig. 9.10(left) (see section 9.2.5).

[7]The 1 000-point trajectory of fig. 9.17(top, left) is 12 times longer than the 100-point one of table 9.1.

9.2.5 Other effects

Reconstruction error in specific places The reconstructed trajectories tend to show a slight error in specific places:

- At the trajectory ends, e.g. in fig. 9.7. This is due to the GTM density estimate being slightly imperfect in the domain boundaries (see fig. 9.3(right)). The MLP estimate is also slightly imperfect there (fig. 9.7(left)).
- At the trajectory turns, e.g. in fig. 9.8(right) for `dpmode`, in fig. 9.10(left) for `grmode` or in fig. 9.13(left) for `mean`. The error consists of cutting short through the turns (for all methods) for mask P1 and, less noticeably, of a spike right at the tip of the turns (for `grmode` and `dpmode`) for mask P2.
  - For `mean` (and also partially for the other methods) this "cutting-short" effect is due, again, to slight imperfections of the GTM density estimate. The Gaussian components interact more strongly in the convex side of the turn, pile up there and bias the mean (see fig. 9.18 and compare with fig. 9.12(left)). The same effect happens when bending a rubber bar: the rubber stretches on the concave side of the bend and compresses and wrinkles on the convex side. This is particularly remarkable with a nonsmooth model such as that of fig. 9.12(left) because the width (variance) of each component is larger: observe how the underlying mapping $\mathbf{f}$ tends to cut short the turns of the forward mapping g. For `grmode` and `dpmode` the "cutting-short" effect is related to the continuity constraint too: cutting through turns of the original trajectory saves trajectory length. This is particularly remarkable in fig. 9.9(right) and fig. 9.13.
  - The spike effect only occurs in multivalued mappings with multiple pointwise reconstruction methods based on all the modes (`dpmode`, `grmode`). The spike is the premature blending of two inverse branches into one branch. As the two branches approach their intersection point, the bumps associated with the two respective modes of the conditional distribution interact and blend into a single unimodal bump before the intersection point (see fig. 9.19).

  The effect size ("cutting-short" bias and spike length) is larger the larger the component variance (σ^2) is; in turn, this is larger the noisier the training set is and the fewer components (K) are used. However, while the "cutting-short" bias would disappear with a perfect density model, the spike would not. Such spikes will always happen in multimodal distributions except for the unrealisable case of perfectly sparse distributions, i.e., delta mixtures (see section 7.12.4 on sparseness).

All variables missing With general missing data patterns, the case of all variables (t_1, t_2) missing at a point n in the sequence results in two different behaviours:

- Single pointwise reconstruction methods prescribe reconstructing them with a fixed value: the mean of the joint density model for `fa` and `mean` and its global mode for `gmode`. This produces large jumps to that fixed value and thus inflates the reconstruction error (fig. 9.9). One could think of improving it by choosing a point near the previous or next point (assuming they can be reconstructed), e.g. by interpolation or simply by replicating that value. In fact this is implicitly a continuity constraint.
- Multiple pointwise reconstruction methods prescribe reconstructing them with all the modes of the joint density model, of which there are 19, 33 and 6 for $K = 20$, 60 and 200, respectively. This adds more flexibility and reduces the reconstruction error, since the jumps are now to one of those modes and are therefore shorter. Even better results are obtained by using all K centroids[8] instead of the modes, particularly for very smooth density models where the components coalesce strongly and decrease the number of modes.

Strictly, though, the case of all variables missing is just a particular case, the most extreme one, of a range of missingness patterns.

9.3 Robot arm inverse kinematics

9.3.1 Introduction to the problem

The inverse kinematics of a robot arm is a prototypical example of a mapping inversion problem, with a well-defined forward mapping and a multivalued inverse mapping (Bernstein, 1967; Asada and Slotine, 1986;

[8] This could also be applied to `mean` or `gmode`.

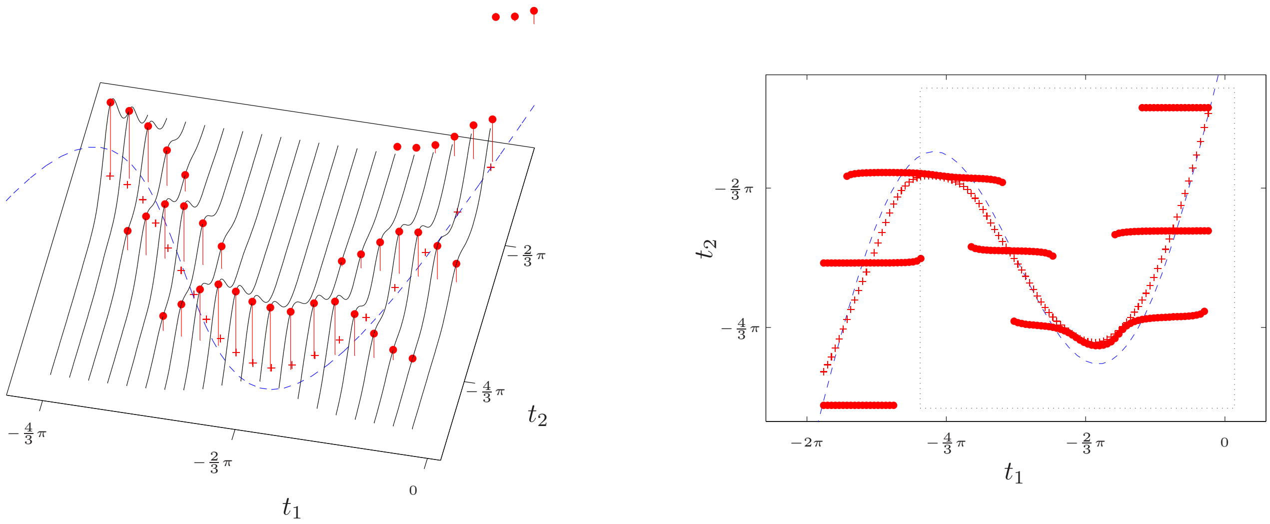


Figure 9.18: Effect of an imperfect density model at trajectory turns, particularly for mask P1: turns are cut short and bias appears. *Left*: conditional distributions $t_2|t_1$ for GTM$_{K=20}$ for several values of t_1 (only the region $[-\frac{4\pi}{3}, -0.25] \times [-5, -1]$ is shown). The Gaussian components interact strongly in the convex side of the mapping (above the curve) and the resulting conditional distribution is skewed towards the convex side. This drives the mean away from the true mapping, inwards of its convex side, thus cutting it short. The mode remains slightly closer to the true mapping because it is less affected by the skewness (see fig. 9.4(right)). The graph shows the mapping g (dashed line), the conditional distributions (thin solid lines) and their means ($+$) and modes ($\bullet$). *Right*: the mean and modes of each $t_2|t_1$ for a larger region (the dotted square marks the previous region) for a much finer sampling of t_1. For Gaussian mixtures, the bias size depends on the component variance; the bias is zero with the true density. Note that the conditional distributions ($p(t_2|t_1)$ for fixed t_1) are not equal to sections of the joint distribution ($p(t_1,t_2)$ for fixed t_1) and so the heights of the curves are not comparable to each other.

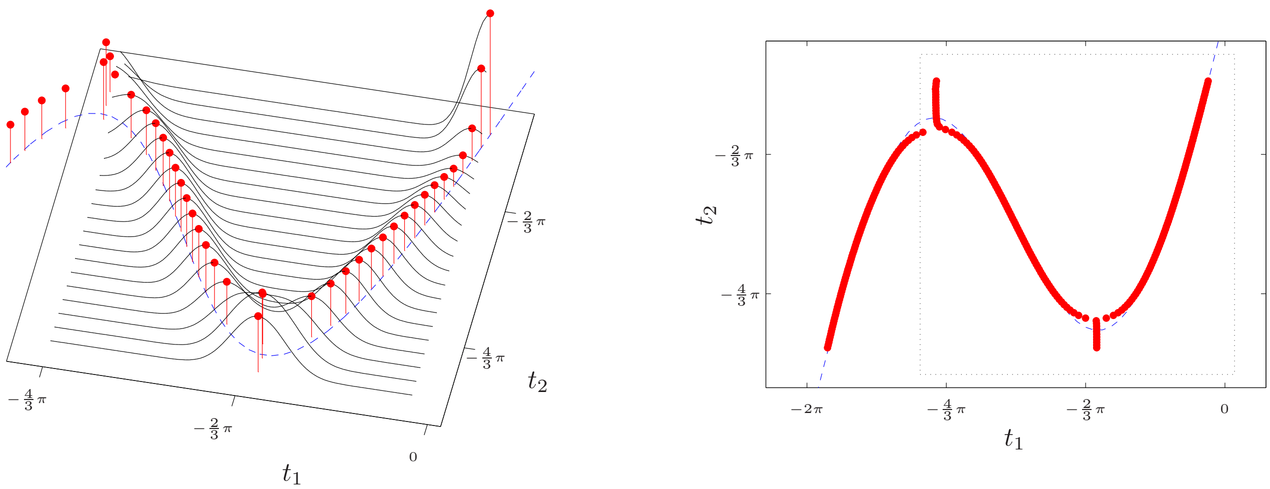


Figure 9.19: Effect of nonsmooth, multimodal conditional distributions for mask P2: spike where inverse branches meet. *Left*: conditional distributions $t_1|t_2$ for GTM$_{K=200}$ for several values of t_2 (the same region is shown as in fig. 9.18). As the two inverse branches approach each other, the two bumps in the conditional distribution coalesce. The conditional distribution becomes unimodal at $t_2 \approx -4.55$ rather than $t_2 = -\arccos\frac{1}{3} - 2\sqrt{2} \approx -4.74$ which is where the branches meet. *Right*: the modes of each $t_1|t_2$ for a much finer sampling of t_2. The length of the spike depends on the noise level of the training data (and on the variance of the density model). For zero noise, all conditional distributions are delta mixtures and the spike disappears; for nonzero noise, the spike appears even with a perfect density model. For nonsmooth models, spurious modes may appear that unfold the ideal branches. Other comments as in fig. 9.18.

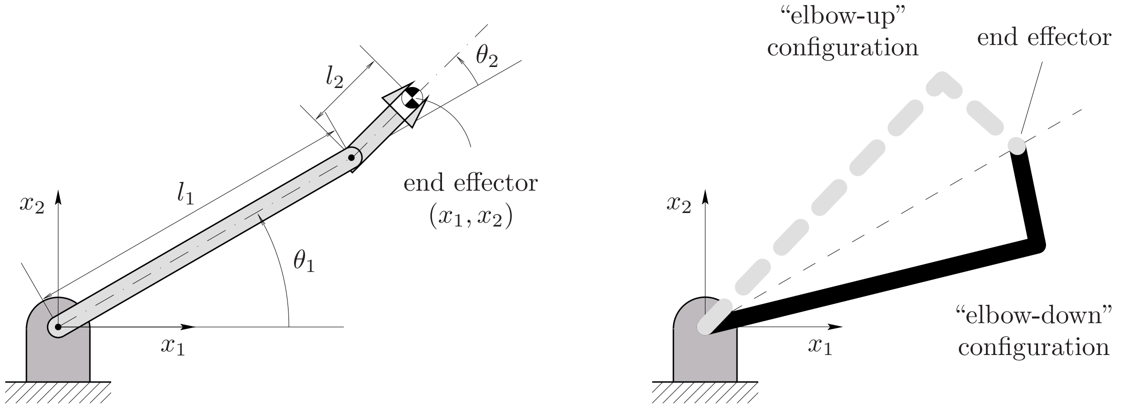


Figure 9.20: *Left*: schematic of a two-link, planar robot arm of joint angles (θ_1, θ_2) and end-effector position (x_1, x_2). *Right*: two different configurations of the joint angles that yield the same end-effector position.

Craig, 1989; Atkeson, 1989). The forward mapping $\mathbf{g}$ gives the position in Cartesian coordinates $\mathbf{x}$ of the end effector (the hand of the robot arm) given the angles $\boldsymbol{\theta}$ at its joints. For the two-joint, planar arm of fig. 9.20, $\boldsymbol{\theta} = \left(\begin{smallmatrix}\theta_1\\ \theta_2\end{smallmatrix}\right) \in \mathcal{A} \xrightarrow{\mathbf{g}} \mathbf{x} = \left(\begin{smallmatrix}x_1\\ x_2\end{smallmatrix}\right) \in \mathcal{C}$ where $\mathcal{A}$ is the *actuator* or *articulator space* and $\mathcal{C} \stackrel{\text{def}}{=} \mathbf{g}(\mathcal{A})$ the *Cartesian*, *task* or *work space* (i.e., the region reachable by the end effector), and always $\dim \mathcal{A} \geq \dim \mathcal{C}$. If $\dim \mathcal{A} > \dim \mathcal{C}$ the manipulator is called *redundant*. Define:

Forward kinematics Transformation from joint[9] angles to end-effector positions. For fig. 9.20 we have:

$$x_1 = l_1 \cos\theta_1 + l_2 \cos(\theta_1 + \theta_2) \tag{9.1a}$$

$$x_2 = l_1 \sin\theta_1 + l_2 \sin(\theta_1 + \theta_2) \tag{9.1b}$$

where the lengths l_1 and l_2 are known and constant. In the human arm case they are not constant and represent muscle lengths, so that the forward kinematics are $(\boldsymbol{\theta}, \mathbf{l}) \xrightarrow{\mathbf{g}} \mathbf{x}$, but we will concentrate on the simple case.

Inverse kinematics Transformation from the desired end-effector position to the corresponding joint angles. For equations (9.1) it can be obtained analytically (Asada and Slotine, 1986; Atkeson, 1989):

$$\theta_2 = \arccos\left(\frac{x_1^2 + x_2^2 - l_1^2 - l_2^2}{2 l_1 l_2}\right) \tag{9.2a}$$

$$\theta_1 = \arctan\left(\frac{x_2}{x_1}\right) - \arctan\left(\frac{l_2 \sin\theta_2}{l_1 + l_2 \cos\theta_2}\right). \tag{9.2b}$$

Thus, there are two solutions ("elbow up" and "elbow down") depending on what quadrant θ_2 is chosen from (fig. 9.20(right)).

Forward dynamics Transformation from the torques applied at the joints to movement (positions, velocities and accelerations of joint angles): $\boldsymbol{\tau} \rightarrow \boldsymbol{\theta}, \dot{\boldsymbol{\theta}}, \ddot{\boldsymbol{\theta}}$.

Inverse dynamics Transformation from a desired pattern of motion to the activation commands necessary to achieve that motion.

Asada and Slotine (1986) and Atkeson (1989) give the equations for the forward and inverse dynamics of the arm of fig. 9.20. The *planning problem*, not considered here, consists of determining a trajectory in the workspace that will achieve some goal (e.g. moving an object from one location to another while avoiding obstacles). The *control problem* consists of specifying the appropriate commands in the actuator space that will produce a desired trajectory in the workspace. Motor learning can be viewed as building an accurate inverse model of the motor apparatus. Biologically, that we can view a target in space, close our eyes and

[9] *Joint* here is a noun meaning the physical junction between two links of the robot arm (thus *joint (angle) variables = angles at the joints*). It should not be confused with the adjective *joint* meaning *conjunct*, which is the sense we have used in chapter 7 (thus *density model of the joint variables = density model of all the variables taken together*, not *density model of the joint angles*).

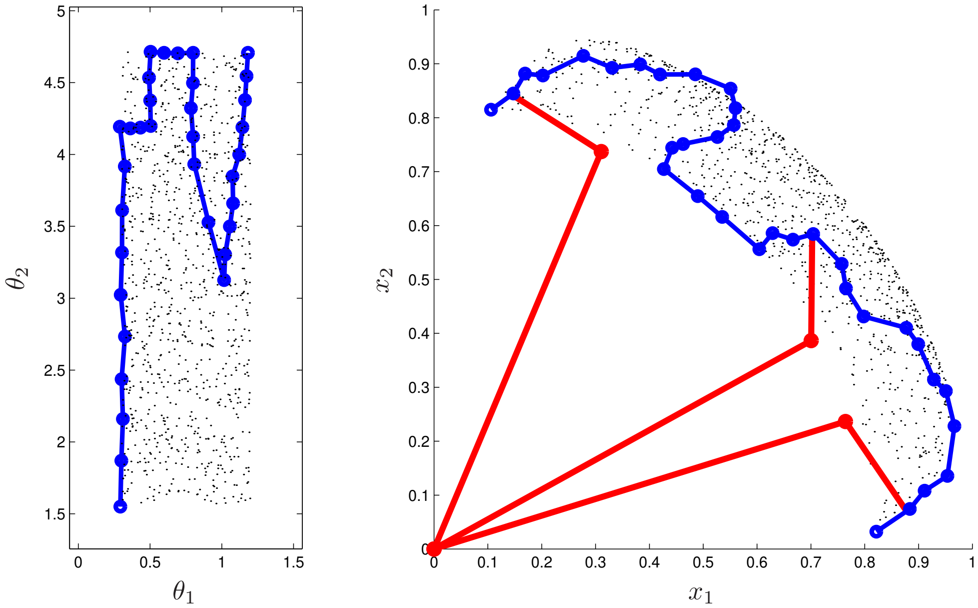


Figure 9.21: Trajectory of the robot arm end effector to be reconstructed. *Left*: trajectory in actuator space (θ_1, θ_2). *Right*: trajectory in work space (x_1, x_2) and three sample robot arm configurations. The training set used is shown in black dots: on the left graph it is a uniform cloud in actuator space; on the right graph it delineates the work space (the region reachable by the end effector).

move our hand to that target suggests that we have an internal representation of the transformation from desired hand positions to joint angles and muscle lengths (Atkeson, 1989). In speech research, internal models for the transformation from desired acoustic signal to vocal tract configuration have also been suggested (see section 10.1.2).

Analytical inversion of the kinematic equations can be done for some simple arms but not in general. Several methods exist (for reviews, see for example Atkeson, 1989 or DeMers and Kreutz-Delgado, 1996 and references therein): standard numerical analysis techniques for solving nonlinear systems of equations, algorithmic methods or neural networks. These methods alone do not work well because the inverse mapping is multivalued. In general, the problem of inverting forward mappings which are continuous, smooth and nonlinear is ill-posed in two senses: globally, because of the existence of multiple solution branches; and (for manipulators with redundant degrees of freedom) locally, because the solution branches are typically manifolds with dimension equal to the number of redundant degrees of freedom (DeMers and Kreutz-Delgado, 1996). When generating commands for a series of points in a desired trajectory, the resulting trajectory must avoid inefficient or impossible movements of the arm. Studies of trajectory formation have considered cost functions such as minimum energy, minimum torque, minimum jerk (third derivative of the position) and minimum acceleration (second derivative) (Nelson, 1983).

9.3.2 Experiments

As a demonstration of our method, we will apply it to the inverse kinematics of the arm of fig. 9.20 and equations (9.1), disregarding the dynamics and considering constant link lengths ($l_1 = 0.8$, $l_2 = 0.2$). This particular problem, or slight variations of it, have been often used in the pattern recognition literature (MacKay, 1992b; Bishop, 1994; Neal, 1996; Cohn, 1996; Rohwer and van der Rest, 1996). We will use a very simple cost function: continuity in the space $(\boldsymbol{\theta}, \mathbf{x})$.

A training set of $N' = 1\,000$ points was generated by sampling uniformly the actuator space $\mathcal{A} = [0.3, 1.2] \times [\frac{\pi}{2}, \frac{3\pi}{2}]$, then applying the forward mapping and finally adding normal spherical noise of standard deviation $\sigma = 0.05$. We trained the following models:

- MLP: one MLP with a single layer of $h = 48$ hidden units (mask P1; 242 parameters) and five MLPs with a single layer of $h = 5, 15, 25, 35, 48$ hidden units, respectively (mask P2; between 27 and 242 parameters).
- Factor analysis: latent space of dimension $L = 2$ (15 parameters).

- GTM: latent space of dimension $L = 2$, 15×15 latent grid and 7×7 RBF grid (201 parameters, $K = 225$ spherical Gaussian components).

For `sampdp`, we generated $S = 6$ samples per conditional distribution.

We then reconstructed the trajectory of fig. 9.21 with $N = 34$ points, which was manually designed and to which small normal noise ($\sigma = 0.01$) was added. The masks were generated as in section 9.1. The reconstruction results, given in table 9.5, show a similar behaviour to that observed in results of the toy experiments: `dpmode` beats the other methods (in particular, the `mean` and the MLPs, both of which perform very similarly) and its performance is often close to the bound of `cmode`, even for high amounts of missing data. The largest error for `dpmode` occurs for the inverse mapping (mask P2; also note the very low $\mathscr{C}_1$-value), which confirms that regression problems, when multivalued, are harder than general missing data patterns. All methods perform equally well in the univalued, forward mapping (mask P1). The number of modes at every point in the trajectory is given in figures 9.22 and 9.23: while it is correct for the forward mapping (which is univalued), there are many modes for the inverse mapping (which is uni- or bivalued). This means that the density model is not smooth, although in this case it does not seem to affect the `dpmode`.

These results demonstrate that, in this problem, the continuity constraint $\mathscr{C}$ alone can allow good reconstruction with `dpmode`. However, since the forward mapping is known, it can perfectly be incorporated in the constraint (section 7.5.3 and term $\mathscr{F}$ in eq. (7.2)) to improve the reconstruction quality.

Average squared error $\frac{1}{N}\sum_{n=1}^{N}\left\|\mathbf{t}^{(n)}-\hat{\mathbf{t}}^{(n)}\right\|^2$

Mask (% missing)	Factor analysis	MLP			GTM with $K = 225$							
		`mlpbest`	`mlpavg`	`mlpdp`	`mean`	`gmode`	`rmode`	`cmode`	`grmode`	`dpmode`	`sampdp`	`meandp`
P1(50%)	0.0130	0.0007	0.0007	0.0007	0.0014	0.0017	0.0017	0.0017	0.0017	0.0017	0.0027	0.0014
P2(50%)	0.7690	0.6371	0.6394	0.6325	0.6767	1.2998	1.4603	0.0057	3.4837	0.3230	0.4602	0.3230
P3(74%)	0.7369				0.7084	1.9592	1.7930	0.0072	0.2395	0.0928	0.1366	0.0928
P4(50%)	0.4136				0.3655	1.1159	1.1873	0.0045	0.4156	0.1092	0.0717	0.1094
P5(25%)	0.2297				0.1486	0.1539	0.2371	0.0029	0.0343	0.0074	0.0147	0.0080
P6(50%)	0.4033				0.3511	0.9107	0.7333	0.0037	0.1979	0.0239	0.1120	0.0236
P7(5%)	0.1964				0.1940	0.4122	0.3314	0.0017	0.0092	0.0023	0.0104	0.0023

Value of constraint $\mathscr{C}_1 = \sum_{n=1}^{N-1}\left\|\hat{\mathbf{t}}^{(n)}-\hat{\mathbf{t}}^{(n+1)}\right\|$ (original sequence: 7.21)

Mask (% missing)	Factor analysis	MLP			GTM with $K = 225$							
		`mlpbest`	`mlpavg`	`mlpdp`	`mean`	`gmode`	`rmode`	`cmode`	`grmode`	`dpmode`	`sampdp`	`meandp`
P1(50%)	7.2	7.2	7.2	7.2	7.3	7.3	7.3	7.3	7.3	7.3	7.2	7.3
P2(50%)	2.9	4.7	4.8	4.5	5.8	19.2	34.9	7.1	4.5	3.5	6.8	3.5
P3(74%)	21.5				21.2	42.3	36.3	7.6	6.8	5.6	10.5	5.6
P4(50%)	25.5				24.1	40.3	43.6	7.3	8.1	6.3	9.2	6.3
P5(25%)	14.4				12.6	12.4	14.5	7.3	7.7	7.0	7.4	7.0
P6(50%)	12.8				11.8	22.2	20.9	7.3	8.1	7.0	8.4	6.9
P7(5%)	12.6				12.7	16.7	15.7	7.4	7.7	7.4	8.2	7.4

Table 9.5: Reconstruction results for the robot arm problem: average squared error and global constraint value (trajectory length) for the noisy trajectory $\{\mathbf{t}^{(n)}\}_{n=1}^{N}$ with $N = 34$ of fig. 9.21. $\{\hat{\mathbf{t}}^{(n)}\}_{n=1}^{N}$ is the reconstructed trajectory by the corresponding method. For mask P1, `mlpbest` = `mlpavg` = `mlpdp` since only one MLP was used. For mask P2, `mlpbest` corresponded to the MLP with 15 hidden units. For masks P3–P7, MLPs are not applicable.

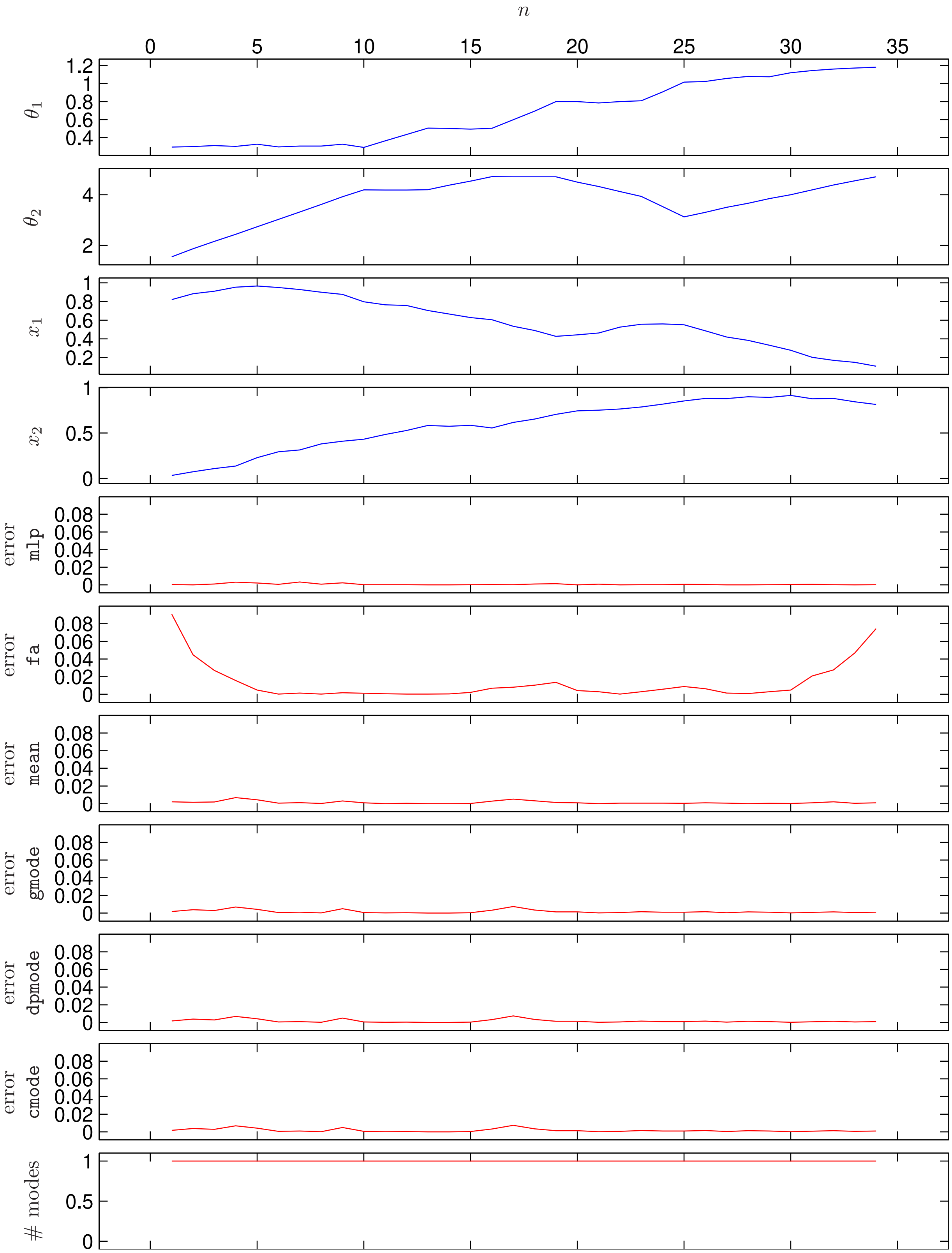


Figure 9.22: Reconstruction error as a function of the sequence index n: forward mapping (mask P1). The data are for the noisy trajectory of $N = 34$ points of fig. 9.21. *Top four graphs*: $t_1^{(n)}$, $t_2^{(n)}$, $t_3^{(n)}$ and $t_4^{(n)}$ ($\theta_1^{(n)}$, $\theta_2^{(n)}$, $x_1^{(n)}$ and $x_2^{(n)}$, respectively) for the original trajectory. *Bottom graph*: number of modes of the conditional distribution $(x_1^{(n)}, x_2^{(n)})|(\theta_1^{(n)}, \theta_2^{(n)})$. *Rest of graphs*: reconstruction error $\left\|\mathbf{t}^{(n)} - \hat{\mathbf{t}}^{(n)}\right\|^2$ for the respective method. The number of modes is one, as it should.

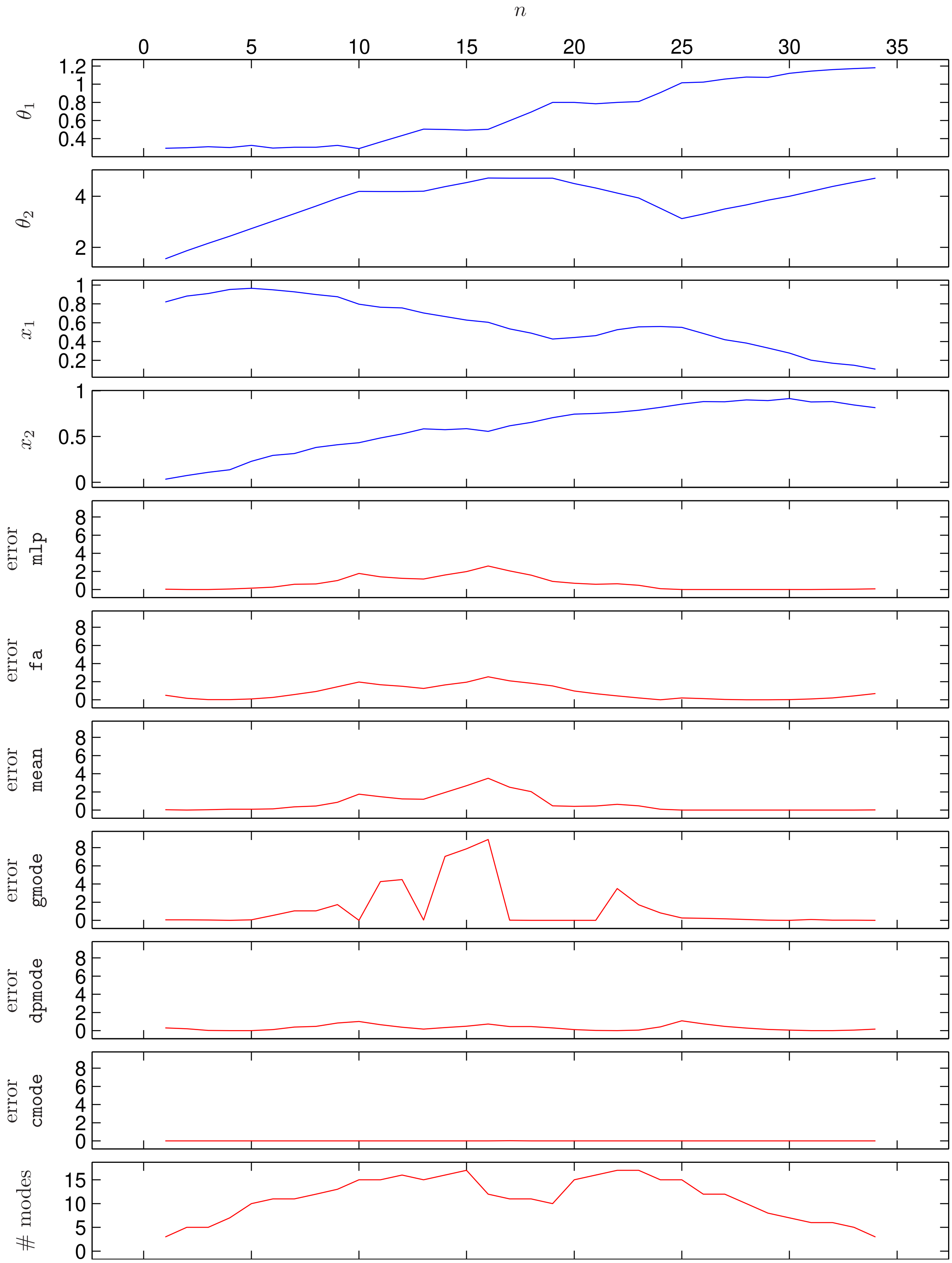


Figure 9.23: Reconstruction error as a function of the sequence index n: inverse mapping (mask P2). The data are for the same trajectory of fig. 9.22, but note the different scale in the vertical axes. Graphs as in fig. 9.22 except for *bottom graph*: number of modes of the conditional distribution $(\theta_1^{(n)}, \theta_2^{(n)})|(x_1^{(n)}, x_2^{(n)})$. The number of modes (which should be at most two, since there are at most two inverse values) is typically very large, which indicates ripple in the conditional distribution.

Chapter 10

Experiments with real-world data: the acoustic-to-articulatory mapping problem

In this chapter, we apply our reconstruction algorithm to a version of the acoustic-to-articulatory mapping, a well-known mapping inversion problem of speech research. It is a complex, high-dimensional task that helps to further understand the performance of the algorithm. Before the description and discussion of the experimental results of section 10.2, we give some background of the problem and its significance for speech perception and automatic speech recognition (ASR) in section 10.1.

10.1 The acoustic-to-articulatory mapping problem

We describe the problem of articulatory inversion, its relation with the motor theory of speech perception, computational approaches for its solution and speech models incorporating articulatory information.

10.1.1 The problem

Broadly speaking, the acoustic-to-articulatory mapping problem consists of determining the vocal tract shape that produced a certain acoustic signal. The forward mapping, from a vocal tract shape or articulatory configuration to the acoustics, is univalued but nonlinear and many-to-one, which makes its inversion difficult (see fig. 10.1). The problem is also further complicated by the fact (among others) that the articulatory and

This chapter is partly based on references Carreira-Perpiñán and Renals (1999); Carreira-Perpiñán (2000b).

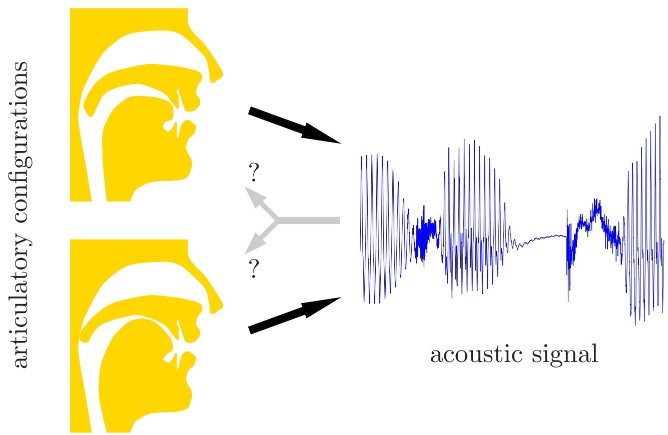


Figure 10.1: The acoustic-to-articulatory mapping problem: one vocal tract configuration produces a unique acoustic signal, but certain acoustic signals may be produced by multiple vocal tract configurations.

acoustic variables are not as clear cut as in, say, the robot arm problem of section 9.3. However, the problem is very important from both engineering and perceptual points of view.

10.1.1.1 Forward mapping: sound propagation in the vocal tract

The vocal tract acts as an acoustic filter that modifies the spectrum of the excitation signal at the glottis. From the point of view of articulatory synthesis one has to model the following elements (Fant, 1970; Flanagan, 1972; Schroeter and Sondhi, 1994):

Geometry of the vocal/nasal tract The vocal tract can be idealised as a straightened, nonuniform acoustic tube extending from the glottis ($x = 0$) to the lips ($x = L$) whose cross-section *area function* $A(x)$ varies continuously but slowly as a function of time. Information about its geometry may be obtained from X-ray measurements.

Wave propagation in the tract It can be described by *Webster's horn equation* (first derived by Bernoulli, Euler and Lagrange in the XVIII century). This is a second-order linear differential equation for the pressure (and volume velocity) as a function of x for a glottal signal and a given $A(x)$. Its solutions are plane waves, i.e., pressure and velocity are constant in a plane perpendicular to the tract axis. The equation is valid as long as the greatest cross-dimension of the tract is appreciably less than a wavelength, which means for frequencies smaller than 4 kHz. Nonlinear effects are important with turbulent flow (high Reynolds or Mach number), which happens through the vocal cords or through narrow constrictions as in fricatives[1]. The equation can be extended to account for effects of energy loss due to viscous friction, thermal conduction and yielding tract walls.

Sound sources and their interaction with the tract This requires a nonlinear model of the glottis (vocal cords).

Therefore the shape of the vocal tract is completely specified at any one time by the area function $A(x)$.

The **direct problem**, i.e., to determine the volume velocity and pressure of the air given $A(x)$ and certain other parameters, can be solved for any given boundary conditions at the lips and glottis. That is, the articulatory configuration plus the glottal source causally determine the acoustics. This allows speech synthesis from articulatory parameters, and is usually straightforward but computationally expensive. The **inverse problem**, i.e., to obtain articulatory information (in particular $A(x)$) from acoustic information extracted from the speech signal, does not have a unique solution: the transfer function of the vocal tract does not uniquely specify the area function[2] and so the acoustic signal at the lips does not either. There are two kinds of nonuniqueness:

- Different tract shapes may have or almost have the same transfer function, thus producing the same acoustic signal for the same given input at the glottis.
- The same acoustic signal may be produced by two different tract shapes with appropriate inputs at the glottis, i.e., changes in the source can compensate for certain changes of the tract transfer function.

Most models address only the former. The only way to deal with the nonuniqueness problem is to use constraints on the area function, particularly temporal continuity.

10.1.1.2 Articulatory variables and their properties

The articulatory configuration or vocal tract shape can be represented in various ways depending on the purpose of the inversion, but in any case it should give a reasonably complete description of the shape of the vocal tract. The area function $A(x)$ is such a complete description but for computational convenience a finite set of articulatory variables[3] is used. This can be achieved by discretising the area function or by using

[1] The main characteristics of speech sounds are (Ladefoged, 2000; Rabiner and Juang, 1993): *voicedness*, where the tract is excited by vibrating vocal cords (e.g. vowels, diphthongs); *frication*, where the tract is excited by turbulence due to flow through a narrow constriction (e.g. /s,z,ʃ,ʒ,f,v/); *plosiveness (stops)*, where the tract is excited by sudden release of pressure (e.g. /p,t,k,b,d,g/); *nasality*, where part or all of the flow is diverted into the nasal tract (e.g. /m,n,ŋ/); *silence*, where there is no excitation (e.g. during stop closures).

[2] The acoustic input impedance of the vocal tract at either end of the tract does uniquely specify the area function (for lossless vocal tracts only). But such information is not easily measurable.

[3] In the acoustic-to-articulatory mapping literature the representation of the vocal tract is usually called articulatory features or parameters or configuration or shape and the representation of the speech signal is usually called acoustic features. We will generally call them articulatory variables and acoustic variables in accordance with the rest of this thesis.

instead the positions of particular articulators or landmarks of articulators, such as the tongue tip, tongue body, jaw, velum, lip opening, etc. Ladefoged (1980) has suggested a set of 16 articulatory parameters which are necessary and sufficient to uniquely characterise all the sounds of every known language, but most studies use fewer than 10 and restrict them to the midsaggittal plane (the plane of fig. 10.1).

The articulators are masses accelerated with finite forces and occupy space, so they are subject to mechanical constraints. Various kinds of constraints have been proposed: static, which discard physically unrealisable articulatory configurations (e.g. 'tongue tip cannot go through the roof of the mouth'); dynamic, in that they change continuously and slowly with the time; and others such as those derived from the economy of skilled movements (Nelson, 1983), e.g. minimal muscle effort or work.

Data for the articulatory variables and their constraints can be derived from measurements with X-rays or EMA (section 7.10.5) or from articulatory models. An **articulatory model** is a geometric model of the vocal tract in terms of several parameters. For example, in the articulatory model of Mermelstein (1973) the parameters are the locations of tongue body centre, velum, tongue tip, jaw, lips and hyoid. The model allows the computation of the area function $A(x)$ that results from given values of the parameters; such area functions can then be used in Webster's horn equation. Articulatory models are aimed at representing mechanical constraints of the vocal tract and so they can be used to generate feasible vocal tract configurations (i.e., not all values of the parameters are allowed). The most favoured models are those of Mermelstein (1973), Coker (1976) and Maeda (1982). They are primarily based on (often two-dimensional) X-ray studies of the vocal tract. For the purpose of articulatory inversion one can use the articulatory parameters directly rather than the area functions.

Besides the squared reconstruction error, or root-mean-square (RMS) error, two other measures of quality of articulatory recovery are often used:

- Pearson product-moment correlation, which quantifies for a given articulator the similarity in shape between two trajectories regardless of magnitude, i.e., whether they rise and fall in synchrony:

$$r \stackrel{\text{def}}{=} \frac{\operatorname{cov}\{a,b\}}{\operatorname{stdev}\{a\}\operatorname{stdev}\{b\}} = \frac{\sum_{n=1}^{N}(a^{(n)}-\overline{a})(b^{(n)}-\overline{b})}{\sqrt{\left(\sum_{n=1}^{N}(a^{(n)}-\overline{a})^2\right)\left(\sum_{n=1}^{N}(b^{(n)}-\overline{b})^2\right)}} \in [-1,1] \tag{10.1}$$

  for two sequences $\{a^{(n)}\}_{n=1}^{N}$ and $\{b^{(n)}\}_{n=1}^{N}$ of means $\overline{a}$ and $\overline{b}$, respectively.

- In the context of automatic speech recognition (ASR), some measure of articulatory gesture or phoneme recognition, such as a phone classification score.

10.1.1.3 Acoustic variables and their properties

The raw acoustic signal as a function of time is not convenient because of its high sampling rate (normally around 20 kHz) and variability (due to inter- and intraspeaker variability, noise and coarticulation). Depending on the problem, other representations (Rabiner and Juang, 1993; Gold and Morgan, 2000) are used that result in a vector time series with a rate of the order of 100 Hz (closer to that of the articulators). In decreasing order of closeness to the vocal tract shape:

Formants The formants are the resonances of the vocal tract and are therefore very closely related to its shape, changing slowly with time and showing relatively simple phonemic transitions. Besides, they are quite robust to noise. However, they cannot be generally used: they are not always visible in the spectrum (e.g. when a narrow constriction decouples the rear cavity, as in fricatives) and they are difficult to extract reliably. The formants have been often used in studies of the acoustic-to-articulatory mapping for vowels.

Linear predictive coding (LPC) performs spectral analysis on speech frames with an all-pole filter. It provides a good approximation to the vocal tract spectral envelope for voiced speech and achieves a reasonable source-vocal tract separation. It is less effective for unvoiced and transient regions of speech. Other variations of LPC are *line spectral frequencies* (LSF) and *line spectral pairs* (LSP).

Filter banks The speech signal is passed through a bank of several independent but overlapping bandpass filters collectively spanning the frequency range of interest. Thus, the output of each filter is a short-time spectral representation of the signal at the filter's centre frequency at the current time frame.

Auditory-based cepstral representations A smoothed short-term spectrum is derived from a filter bank that has been designed according to some model of the auditory system. The features are decorrelated with a linear transformation which also separates out pitch, spectral detail and spectral tilt. The most common variants are the *mel cepstrum*[4] (MFCC) and *perceptual linear prediction* (PLP) (Hermansky, 1990), which provide very similar features; a more recent proposal are *modulation-filtered spectrogram* (MSG) features (Kingsbury et al., 1998), developed for speech recognition robust to acoustic interference such as additive noise and reverberation. In addition to the cepstral coefficients, estimates of their velocities and accelerations are often used to account for dynamic features of speech. Most current ASR systems use this representation. However, cepstra are sensitive to noise (because of the logarithmic compression and subsequent spread over all features by the linear transformation), to coarticulation and speaker-dependent; and they present complex transitions and discontinuities where the excitation changes.

Thus, while the raw acoustic waveform is continuous in time, the acoustic variables generally are not, even for representations like LSPs which are closer to the formants. And articulatory trajectories that differ only slightly can result in very different acoustic utterances. Deng et al. (1997) mention a good, well-known example: stop epenthesis after nasals. This occurs when an extra silence and burst are introduced in the acoustic signal due to variation in timing between adjacent velic and oral closures. For example, the realisation of the word "princess" as [printses] or [prinses] depends on whether the velum is raised before or after the release of the alveolar stop, and the amount of desynchronisation varies continuously (as observed in articulatory data). Accounting for this in the acoustic domain is far more difficult than in the articulatory domain, requiring either to assume an extra stop phoneme for the word in question or to extend the acoustic model of the nasal to include the epenthetic stop.

In many of the computational approaches for the acoustic-to-articulatory mapping problem, an **acoustic distance** is required, i.e., a distance between acoustic vectors. Many such distances have been proposed in the speech literature, often quite complex to make them more relevant to human perception (distortion measures). For filterbank and cepstral vectors, an L_1, L_2 or covariance weighted spectral difference is often used; for LPC coefficients, the likelihood ratio distance is more appropriate (Rabiner and Juang, 1993, chapter 4). Likewise, an **articulatory distance** is required, and several have been proposed, with the Euclidean one being often used.

10.1.1.4 Example of the nonuniqueness

Many examples of the nonuniqueness of the acoustic-to-articulatory mapping have been given in the literature, resulting from both articulatory models and human experiments (Schroeter and Sondhi, 1994). A familiar demonstration are ventriloquists, who can produce intelligible speech without moving the lips. Another often-cited example is that of the approximant consonant /ɹ/ of American English (the 'r' as in 'beaker', 'perk', 'rod' or 'street') (Westbury et al., 1998; Espy-Wilson et al., 2000). Speakers of rhotic dialects of American English use many different articulatory configurations for /ɹ/, which are all acoustically characterised by an extremely low frequency of the third formant (often close to that of the second formant). These configurations expose an ante/sub-lingual cavity and involve three constrictions: in the pharynx, along the palate and at the lips. The configurations differ most in the palatal constriction and have traditionally been divided into contrasting categories of *retroflex* (tongue tip raised, tongue dorsum lowered) and *bunched* (tongue dorsum raised, tongue tip lowered), but there really seems to exist a continuum between them. These different configurations occur both within and across speakers: some speakers may use one type of configuration exclusively while others switch between two or three different types in different phonetic contexts and according to prosodic variables.

Computational models have also confirmed the nonuniqueness of the acoustic-to-articulatory mapping. Atal et al. (1978) found articulatory regions (fibers) that map onto a single acoustic point by linearising the forward mapping in a small neighbourhood and extending it in small steps. They found that many sounds can be produced by many different vocal tract shapes.

Finally, from a theoretical standpoint, the nonuniqueness appears for lossless vocal tracts with fixed boundary conditions: the area functions $A(x)$ and $1/A(L-x)$ (where L is the vocal tract length) produce the same transfer function. For lossy vocal tracts, it is not clear whether the nonuniqueness remains, but practically more than one area function produce very similar transfer functions.

[4] The (complex) cepstrum of a signal is the Fourier transform of the log of the signal spectrum (which is itself the Fourier transform of the signal).

10.1.1.5 Coarticulation

Coarticulation broadly refers to the fact that a phonological segment is not realised identically in all contexts, but often apparently varies to become more like an adjacent or nearby segment (Hardcastle and Hewlett, 1999). It can be anticipatory (a later segment influences an earlier one) or carryover (vice versa). For example, the English phoneme /k/ is articulated further forward on the palate before a front vowel ([k̟iː] 'key') and further back before a back vowel ([k̠ɔː] 'caw'); and will have a lip position influenced by the following vowel (e.g. rounded in [k̠ʷɔː] 'caw'). As another example, in velopharyngeal coarticulation nasality spreads from a consonant to a neighbouring vowel: compare the /a/ in /as/ and /an/.

The reason for coarticulation is that the vocal tract cannot move from one target configuration to the next one instantaneously, so instead of keeping each phoneme as an invariant articulation and then slowly moving to the next, the articulators follow a faster, more graceful trajectory. The higher the coarticulation the more fluent the sequence and the more difficult it is to isolate individual phonemes. The same happens in handwriting.

Coarticulation is thought to have advantages for perception: spreading the effect of a phoneme to a larger interval makes it more likely to be spotted and several phonemes may be processed in parallel. Thus, coarticulation could be an adaptation of the human communication system to maximise the transmission rate at its bottleneck: the slow-moving articulators. However, coarticulation makes each acoustic unit depend heavily on its context, which makes speech recognition difficult. Dealing with coarticulation should be much easier in its native, articulatory domain than in the acoustic one.

There is a parallelism in terms of planning between an utterance and a robot arm trajectory in that targets must be met: in the utterance, these are acoustic targets given by the phonemic transcription of the utterance, while in the robot arm these are physical locations through which the end-effector must pass (perhaps avoiding obstacles). The targets are given in the acoustic or work space and result in corresponding targets in the articulatory space. However, in speech the articulatory targets do not necessarily have to be fully realised for speech to be intelligible, particularly in fast speech styles, as shown by coarticulation.

10.1.1.6 Critical vs non-critical articulators ("don't care" values)

The concept of critical articulators refers to the fact that, for a given production, the movement of a small subset of articulators is crucial, while the movement of the rest is not. For example, the acoustics, particularly the formants, are more sensitive to place and degree of constriction than to the rest of the area function. One reason for this is that the coefficients of Webster's horn equation are functions of the logarithm of the area function. Recasens' work on coarticulation in Catalan (reviewed in Hardcastle and Hewlett, 1999, chapters 2 and 4) showed that the more an articulator is involved in producing a consonant, the less susceptible it is to coarticulatory influences from adjacent vowels.

Papcun et al. (1992) demonstrated empirically that critical articulators are less susceptible to coarticulation and have a greater range of variation than noncritical articulators, which are freer to vary or play along. They used real articulatory data (recorded with an X-ray microbeam) and trained a neural net to learn the mapping from acoustics[5] (bark scaled FFT bins) to articulators' positions (lower lip, tongue tip and tongue dorsum) for the English stop consonants /p,b,t,d,k,g/. The critical articulators were the lower lip for bilabials (/p,b/), the tongue tip for alveolars (/t,d/) and the tongue dorsum for velars (/k,g/). By comparing the articulatory trajectories inferred by the neural net with the original ones, they found good correlation ($r \approx 0.9$) for the critical articulators of each consonant type and bad correlation ($r = 0.19$ to 0.78) for the noncritical ones, but higher RMS error for the critical than for the noncritical ones. They hypothesise that intra-speaker variability of non-critical articulators could be caused by principled differences (such as differences in phrasal stress) or considered as noise; while inter-speaker variability could result from the fact that each speaker has acquired idiosyncratic patterns of noncritical articulator movement but shares patterns of critical articulator movement with other speakers.

Another case of "don't care" values occurs during silence intervals in the speech (e.g. during stop closures), in which the output spectrum is similar to that of the background noise and does not contain any information regarding the shape of the vocal tract.

"Don't care" values also occur in the robot arm problem (section 9.3). For example, Jordan (1990) considers the case of a robot with two arms: if at some moment there is only one target to manipulate, the arm which is not manipulating the target is free to move; but it may move towards a future target to anticipate a movement and so make the transition to the next configuration easier (faster, requiring less energy, etc.). Coarticulation

[5] Rather than using as input the acoustic vector of a single frame, they used a sequence of 25 frames (around 200 ms), which they called *context frame*.

is the analog case in speech production. The same phenomena should occur if the vocal tract is extended to include facial features (section 7.10.4).

10.1.1.7 Significance for speech processing

The constraints of the speech production system (such as slow, continuous dynamics, coarticulation, identification of critical articulators and speaker-dependent characteristics such as vocal tract length) can be better represented by an articulatory representation of speech than by an acoustic one. An articulatory representation should be useful for:

- Speech recognition: traditional speech recognisers have problems with nasals, voiced and unvoiced stops and voiced and unvoiced fricatives, for which spectrograms are very similar, discriminatory features concentrate on very few transitionary frames and the spectrum between transitionary frames is ambiguous or useless. They also have problems with the effects of prosody and coarticulation, which they partially alleviate by using context-sensitive models (e.g. diphones), but at the cost of using many parameters.

  The addition of articulatory features to acoustic features in HMMs has been shown to increase recognition performance over acoustic features alone (Zlokarnik, 1995a). We review some more sophisticated models using production information in section 10.1.4.

- Coding and text-to-speech synthesis: an articulatory representation has lower requirements than an acoustic one for transmission rate and storage because of the slow dynamics of the articulators. Articulatory trajectories are also closer to a phonetic transcription.

- Visualisation of vocal tract features and training aids for the deaf, etc. (as in chapter 5 with the EPG data).

In summary, some aspects of speech processing should be simpler in the articulatory domain. Since usually only the acoustic signal is readily available, articulatory inversion becomes necessary.

10.1.2 The motor theory of speech perception

Irrespectively of any mathematical or engineering approach, how the human brain may perform articulatory inversion is unknown[6]. We briefly review the well-known motor theory of speech and note an interpretation in terms of latent variables.

The basic motivation for speech theory is that people can both perceive speech and produce speech. It seems unparsimonious to assume that the speaker-listener employs two entirely separate processes, one for encoding language and the other for decoding it. A simpler assumption is that there is only one process with appropriate links between sensory and motor components. Speech is then assumed to be perceived by processes that are involved not only in auditory perception but also in speech production.

The motor theory (Liberman et al., 1967; Liberman and Mattingly, 1985) was originally proposed to try to account for perceptual invariance in the face of highly variable, context-dependent, acoustic cues. Experiments show that there is typically lack of correspondence between acoustic cue and perceived phoneme, which rules out the use of acoustic cues as perceptual primitives. For example, coarticulation effects, the fact that any particular acoustic segment will likely to be cueing more than one phoneme at a time. In several cases it appears that perception mirrors articulation more closely than sound. This supports the assumption that the listener uses the inconstant sound as a basis for finding his way back to the articulatory gestures that produced it.

The motor theory makes two basic claims: (1) the existence of an invariant motor code of phonetic gestures shared by speech perception and production; and (2) the existence of an innate, specialised module in the brain responsible for the translation between phonetic gestures and acoustic patterns. Let us examine these claims in more detail.

The existence of a motor code The objects of speech perception are the intended phonetic gestures of the speaker, represented in the brain as invariant motor commands that call for movements of the articulators through certain linguistically significant configurations. These gestural commands are the physical reality underlying the traditional phonetic notions (such as tongue backing, lip rounding or jaw raising) that provide the

[6] Likewise, how the brain solves other motor problems, like arm control, is the subject of active research (for a recent review see Wolpert and Ghahramani, 2000). Concepts similar to those proposed by the motor theory appear there too, such as postulated motor primitives or forward and inverse internal models.

basis for phonetic categories. They are the elementary events of speech production and perception. Phonetic segments are simply groups of one or more of these elementary events; thus [b] consists of a labial stop gesture and [m] of that same gesture combined with a velum-lowering gesture. Phonologically, of course, the gestures themselves must be viewed as groups of features, such as labial, stop or nasal, but these features are attributes of the gestural events, not events as such. To perceive an utterance, then, is to perceive a specific pattern of intended gestures (more or less altered due to coarticulation and other effects).

As a detailed example of how a phoneme would be broken down, from a production point of view, into a sequence of possibly overlapping subphonemic elements, consider the articulation of /b/ (Liberman et al., 1967):

1. Closing and opening the upper vocal tract in such a way as to produce the manner feature characteristic of the stop consonants.
2. Closing and opening the vocal tract specifically at the lips, thus producing the placed feature termed bilabiality.
3. Closing the velum to provide the feature of orality.
4. Starting vocal fold vibration simultaneously with the opening of the lips, appropriately to the feature of voicing.

Other phonemes could be described using these and other gestures:

/p/ has features 1, 2 and 3 but differs in 4 in that vocal fold vibration begins some 50 or 60 milliseconds after opening the lips.

/m/ has features 1, 2 and 4 but differs in 3 in that the velum hangs open to produce the feature of nasality.

/d/ has features 1, 3 and 4 but differs in place of articulation.

Therefore, a phonetic, or motor, gesture can be defined as a class of movements by one or more articulators that results in a particular, linguistically significant deformation over time of the vocal tract. A gesture may be effected by several articulators for several reasons:

- A gesture may require the collaboration of several articulators. For example, lip rounding is a collaboration of the lower lip, the upper lip and the jaw.
- A single articulator may participate in the execution of two different gestures at the same time. For example, the lips may be simultaneously rounding and closing in the production of a labial stop followed by a rounded vowel, as in [bu].
- Prosody effects. For example, producing a stressed syllable requires a greater displacement of some or all of the active articulators than when producing an unstressed one.
- Linguistically irrelevant factors, notably speaking rate, affect the trajectory and phasing of the component movements.

The existence of a specialised module for the interface between speech perception and speech production The existence of a motor code implies the existence of an intimate link between speech perception and speech production. In the motor theory, this link is innate, not learned, and is implemented by a specialised module of the brain. Thus, perception of the gestures occurs in a specialised mode different from the auditory mode.

Computation of the phonetic gestures from the acoustic signal by a cognitive process does not seem reasonable. This justifies the need in the motor theory for a modular account of linguistic perception and the assumption of the existence of a special-purpose computational device that relates gestural properties to acoustic patterns. The conversion from acoustic signal to gesture (i.e., a form of articulatory inversion) is done automatically, so that listeners perceive phonetic structures without mediation by, or translation from, the auditory appearances that the sounds might, on purely psychoacoustic grounds, be expected to have.

The motor theory assumes that adaptations of the motor system for controlling the organs of the vocal tract took precedence in the evolution of speech over the development of a perceiving system. These adaptations made possible not only to produce phonetic gestures, but also to coarticulate them so that they could be produced rapidly. A perceiving system, specialised to take account of the complex acoustic consequences, developed concomitantly.

As biological basis for this specialised module, the motor theory proposes the existence of several neural networks—those that supply control signals to the articulators and those that process incoming neural patterns from the ear—with overlapping activity, so that information is correlated by these networks and passed through them in either direction.

10.1.2.1 Problems of the motor theory

The motor theory looks for a hidden representation of the speech message in terms of articulatory gestures, with information flowing in both directions (articulators → acoustics and acoustics → articulators), passing through the gestural representation. However, the idea of a gestural representation runs into a number of problems. A major shortcoming of the theory is the difficulty of rigorously defining, in physical terms, a particular gesture, due to the complications posed by coarticulation and other factors. This makes the motor gestures hardly more satisfactory as perceptual primitives than the acoustic cues. Further, categorising one group of the infinite number of possible articulatory movements as lip rounding and another as lip closure is entirely a priori. Besides, experiments in language acquisition in newborns have showed that structures of speech perception occur well before those of production.

Thus, while the mere ability of humans to listen and speak suggests that some sort of representation of the speech message must exist in the brain, it does not seem plausible that this representation is in terms of articulatory gestures, as the motor theory assumes. Several recent papers debating whether speech is controlled by auditory-acoustic goals or by articulatory goals appear in the *Journal of the Acoustic Society of America* 99(3):1680–1741 (March 1996); a summary is given by McGowan and Faber (1996).

There exist other feature-based theories, e.g. in phonology. In the theory of articulatory phonology of Browman and Goldstein (1992), the basic units of phonological contrast are called gestures and are also abstract characterisations of articulatory events, each with an intrinsic duration. Utterances are modelled as organised patterns of gestures, called constellations, in which gestural units may overlap in time. Thus, utterances differ from one another in the particular set of gestures they use or in how those gestures are organised, and the same gesture may have different acoustic consequences, depending on other concurrent gestures. The patterns of overlapping organisation can be used to account for several types of phonological variation, including coarticulation. Again, a listener must have a mechanism to recover the underlying gestures from the varying acoustics.

10.1.2.2 A latent-variable view

Regardless of its biological validity, the motor theory and other feature-based theories like that of Browman and Goldstein (1992) are computationally attractive and have been used as the motivation for several ASR approaches, some of which we have described in this thesis (e.g. Papcun et al., 1992; Erler and Freeman, 1996; Richards and Bridle, 1999). We point out here that the motor theory can be formulated as a latent variable model. Let us imagine the existence of a more abstract representation, neither expressed in terms of acoustic cues nor of articulatory gestures, and whose biological basis would reside in neural networks with bidirectional links with both the auditory and the articulatory systems. This could be implemented with a latent variable model trained in an unsupervised way with both acoustic and articulatory data (the latent variables would not necessarily be interpretable as a neural code); and the links between all three domains (acoustic, articulatory and the hidden representation) would be implemented by conditional probability rules.

10.1.3 Computational approaches

Webster's horn equation allows the computation of the acoustic signal resulting from a given area function for some glottal excitation, i.e., the articulatory-to-acoustic (forward) mapping, but not its inverse. Decades of research have been dedicated to computing this inversion. Levinson and Schmidt (1983) started their paper saying that

> The direct determination from a speech signal of the corresponding articulatory parameters, such as area functions or other representations of vocal tract shape, is a long standing problem in speech research.

Almost 20 years later, the problem of articulatory inversion is still unresolved, particularly for unvoiced sounds. Much of the work on it has been reviewed by Schroeter and Sondhi (1994), so we restrict ourselves here to the most important approaches as well as some of the more recent ones. Further review material can be found in some of the references cited in this section.

Many articulatory inversion methods can be used with the *analysis-by-synthesis* procedure, which is an optimisation closed loop in which the spectrum of the synthesised speech is compared to the real one at consecutive speech frames. For each frame, an optimisation procedure tries to minimise an acoustic distance between the two signals. In other words: take a starting articulator configuration $\mathbf{x}_0$; compute the forward mapping $\mathbf{g}(\mathbf{x}_0)$; backpropagate the error (acoustic distance) between the original speech $\mathbf{y}$ and the synthesised speech $\mathbf{g}(\mathbf{x}_0)$ to obtain an improved articulatory vector $\mathbf{x}_1$; iterate till convergence. The optimisation is initialised with an articulatory vector $\mathbf{x}_0$ obtained by some articulatory inversion method (e.g. from a codebook), which should be good to avoid local minima of the distance—which, in fact, is the major problem of this framework. Also, if the starting articulatory vector was good enough, one could avoid the optimisation loop altogether. Analysis-by-synthesis methods usually include as main parts an articulatory model (which describes the geometry of the oral cavity), an articulatory synthesiser (vocal tract model that simulates the physics of sound generation in that cavity), an optimisation algorithm with an error measure, and a spectral estimation algorithm (acoustic variables).

Three approaches to the acoustic-to-articulatory mapping are particularly important:

Dynamic programming search in a large articulatory codebook The use of articulatory codebooks was introduced by Larar, Schroeter, and Sondhi (1988) and its search by dynamic programming by Schroeter and Sondhi (1989). Earlier work by Atal et al. (1978) contains precursory ideas for codebooks. An articulatory codebook is a fixed, very large ($M > 100\,000$ entries) table of M vocal tract shapes (obtained either from measurements with X-ray, EMA, etc. or by sampling an articulatory model) and their respective acoustic output. It disregards glottal excitation. The whole codebook is scanned at every speech frame and the optimal path found by dynamic programming[7] to minimise a cost function of the form (a particular case of eq. (7.2)):

$$\underbrace{\lambda \sum_{n=1}^{N-1} \left\| \hat{\mathbf{x}}^{(n+1)} - \hat{\mathbf{x}}^{(n)} \right\|^2}_{\mathscr{C}} + \underbrace{\sum_{n=1}^{N} \left\| \hat{\mathbf{y}}^{(n)} - \mathbf{y}^{(n)} \right\|^2}_{\mathscr{F}} \tag{10.2}$$

where $\mathbf{y}$ represent original acoustic vectors, $(\hat{\mathbf{x}}, \hat{\mathbf{y}})$ represent codebook vectors (with $\hat{\mathbf{y}} \stackrel{\text{def}}{=} \mathbf{g}(\hat{\mathbf{x}})$), $\mathbf{g}$ is the known[8] articulatory-to-acoustic mapping, the $\mathscr{C}$ term is the continuity constraint (applied solely to the articulatory variables) and the $\mathscr{F}$ term is the forward mapping constraint (applied to all variables, but the articulatory ones cancel out). Thus, the $\mathscr{C}$ term is based on an articulatory distance (typically Euclidean) while the $\mathscr{F}$ term is based on an acoustic distance (of which many variations exist in the speech literature; section 10.1.1.3). Another constraint that has often been used for the articulators (e.g. Shirai and Kobayashi, 1986; Sorokin, 1992; Yehia and Itakura, 1996) is muscle effort, given by a quadratic function of the displacement of the articulators (see section 7.5):

$$\sum_{d=1}^{D_1} c_d (x_d^{(n)} - \overline{x}_d)^2 \tag{10.3}$$

where $\overline{x}_d$ is the equilibrium position of the dth articulator and c_d is a coefficient of tissue elasticity. It can be extended to a Mahalanobis distance or some other distance between the articulatory vector $\mathbf{x}^{(n)}$ and the constant equilibrium vector $\overline{\mathbf{x}}$. Determination of the acoustic-articulatory cost weight λ has been tried in various heuristic ways (Schroeter and Sondhi, 1994).

Although there are heuristic techniques to speed up the search (such as selecting at each frame only the M' best acoustic fits, with M' of the order of $1\,000$), since M is so large, the dynamic programming search is very slow and must be limited to around $N = 20$ frames ($= 200$ ms of speech for a frame shift of 10 ms), having a computational complexity of $\mathcal{O}(NM^2)$.

[7] This use of dynamic programming must be distinguished from the *dynamic time warping* algorithm (Rabiner and Juang, 1993, pp. 200–240), in which dynamic programming and various forms of constraints are applied in pattern comparison for speech recognition purely in the acoustic domain. There, the goal is to align in time and normalise the distance between *two* speech patterns (sequences of spectral vectors) of possibly different durations (number of vectors in the sequence). The constraints try to ensure that there is a proper time alignment by avoiding time reversal (monotonicity constraint) and obtaining a smooth alignment path between the starting point and the end point, both of which are fixed and given by the sequences' length (continuity, slope, endpoint and global path constraints).

[8] "Known" means that either it can be derived analytically from a physical model or it can be reliably approximated from data for the inputs and outputs.

Dynamic programming search of codebooks is currently the most accurate method of articulatory inversion. Its disadvantages are the large size of the codebook, which consequently takes a large storage and results in a slow search (of the order of 300× slower than real time for a 100 000-entry codebook in a 40-MFLOP computer; Schroeter and Sondhi, 1994); and the difficulty of constructing a good codebook. Generating the codebook is a careful process that requires:

- A method to obtain training vectors that adequately span both the articulatory and the acoustic spaces. This can be done in two ways:
  - From measurements (e.g. X-rays): the difficulty is to thoroughly span the articulatory space, since such measurements are a finite collection of one-dimensional trajectories designed in the acoustic space.
  - By finely sampling an articulatory model: this needs to eliminate unfeasible shapes. Another way is to interpolate along an articulatory trajectory determined by some predefined, "root" shapes, but this usually leaves areas of the articulatory space uncovered.
- A method to cluster the training vectors. Quantisation is attained by a clustering algorithm, which can be complicated and time-consuming depending on the acoustic distance used. For example, an intermediate point between two sample points may be associated with an unfeasible articulatory configuration and so standard k-means does not work.
- A definition of distances in both spaces, articulatory and acoustic.

Global parametric mappings with more or less sophisticated architectures, such as polynomials, radial basis functions or neural networks (trained with codebook vectors or measured data). Neural networks are faster and more compact than codebooks, but produce worse mappings, which is not surprising in view of the nonuniqueness of the mapping (section 7.3.4). An example using neural networks (though not the first one) was that of Papcun et al. (1992), described in section 10.1.1.6.

Methods based on local acoustic-to-articulatory mappings These methods try to split the acoustic space into regions such that each region maps one-to-one to a corresponding region in the articulatory space (branch determination step). Inside each region, a function approximator is used, trained in a supervised way using the data (or codebook) vectors that fall in the region. In particular, Rahim et al. (1993) (building on unpublished work by Parthasarathy and Sondhi described by Schroeter and Sondhi, 1994) cluster the training data as described in section 7.11.2.1 resulting in $N_{\mathbf{y}} = 32$ acoustic clusters each containing $N_{\mathbf{x}} = 4$ articulatory subclusters. They then fit a different MLP with 26 hidden units in each of the $N_{\mathbf{x}}N_{\mathbf{y}} = 128$ clusters. At the dynamic programming search stage (which is carried out every 15 pitch periods), first the centroids of the $N_{\mathbf{x}} = 4$ clusters are searched for the one closest to a given acoustic vector; then the $N_{\mathbf{y}} = 32$ mappings for the selected acoustic cluster are used to compute $N_{\mathbf{y}}$ mapped articulatory vectors, which are declared as possible candidates. Rahim et al. claim that the ensemble is 20 times more efficient than a codebook method both in memory and lookup time with results of similar quality. An extra advantage over the codebook is that, after having been trained using the codebook, the MLPs can be bootstrapped from natural speech. The disadvantages of this method were mentioned in section 7.11.2.1:

- There is no guarantee that the local mappings are one-to-one inside every region, and determining the regions is difficult in high dimensions.
- $N_{\mathbf{x}}$ and $N_{\mathbf{y}}$ are ad-hoc parameters (e.g. given an acoustic vector, why should there be precisely $N_{\mathbf{y}}$ candidates?).
- Training the MLP ensemble is difficult. Rahim et al. try several heuristic approaches to determine which MLP to adjust for a given speech frame.

Other methods have been recently proposed. In the analysis-by-synthesis technique of McGowan (1994) the articulatory model is the ASY articulatory synthesiser from the Haskins Laboratories (Rubin et al., 1981), which is based on the articulatory model of Mermelstein (1973). The acoustics are represented by the first three formants. A *task dynamics model* (Saltzman and Kelso, 1987) is added to further constrain the articulatory model: rather than articulatory trajectories, the task dynamics model uses "tract variables," which are variables describing constriction locations and degrees (that are more relevant than other parts of the vocal tract for determining the acoustics). The complexity of the task dynamics model makes difficult the use of derivative-based optimisation methods such as gradient descent. Thus, optimisation is performed by a genetic algorithm (Goldberg, 1989) with fitness = 1/error, where the error is the sum of squared errors for the first three formants. A disadvantage of genetic algorithms is that the parameters for optimisation must be coded into finite length strings (*chromosomes*) and this discretises the parameter space (to 6 bits per variable in his case). Using simulated data for /əbæ/ and /ədæ/ the method recovers most parts of an original articulatory trajectory but has trouble in obtaining precise timing, which is imputed to the lack of additional acoustic information, such as the RMS amplitude.

The approach of Sorokin and collaborators (see Sorokin, 1992; Sorokin et al., 2000 and references therein) is also an analysis-by-synthesis technique, where the inversion is considered from the point of view of standard regularisation theory (Tikhonov and Arsenin, 1977) for ill-posed problems (see chapter 6). In regularisation theory, the ill-posedness of an inverse problem is broken by the use of constraints. Sorokin et al. (2000) propose several types of constraints, most of which have been implicitly or explicitly used earlier in the acoustic-to-articulatory mapping problem; these include bounds on muscle forces, articulatory parameters and area functions, mutual dependence of the articulatory parameters (i.e., low intrinsic dimensionality) or complexity of planning and programming motor commands. The Tikhonov regularisation framework results in a cost function of the form acoustic fitness plus articulatory constraints, just as in eq. (10.2), and so the approach does not differ much from dynamic programming search of articulatory codebooks. Sorokin et al. have proposed articulatory models for vowels and fricatives that take into account a condition of non-turbulent air flow, given by a threshold for the Reynolds number, and minimise muscle effort as in eq. (10.3); and specific optimisation algorithms, since they use the uniform (L_∞) norm as acoustic distance between the formants, which is not differentiable.

Yehia and Itakura (1996) consider a simplified version of the acoustic-to-articulatory mapping problem where the acoustics are given by the first M formant frequencies and the articulator configurations by the first $2M$ coefficients of the Fourier cosine series expansion of the log-area function, for a known vocal tract length. Work by Mermelstein (1967) and Schroeder (1967) showed that a one-to-one relationship holds approximately between the first M odd Fourier coefficients of the area function and the first M formants and set to zero the M even Fourier coefficients, which are undetermined. Yehia and Itakura apply a quadratic cost function to the area function (representing minimal effort, like Sorokin, 1992) to obtain those M even coefficients; this is done at each frame, using the Newton-Raphson method, and no continuity constraint is used. Their results using a small corpus of French vowels are just barely better than those of Mermelstein (1967). This is not surprising since this approach simply converts the one-to-many mapping into a one-to-one mapping without even the guarantee that a solution branch is selected.

Based on the idea that speech production realises articulatory targets while subject to coarticulation, Blackburn and Young (2000) propose a method to produce a smooth articulatory trajectory given a time-aligned phonetic string. The trajectory is obtained from the requirement that it passes through *soft* regions of articulatory space given by each phoneme but keeping articulatory effort low. The soft regions are obtained by replacing the distribution of articulatory positions at the midpoint of a given phoneme (obtained from X-ray data) with an independent Gaussian distribution for each articulator. This is a variation of earlier work by Keating, who used *hard* windows (i.e., uniform distributions rather than Gaussian). The requirement that articulatory effort be low is attained by allowing the articulatory trajectory to under- or overshoot the midpoints (articulatory targets), resulting in a trajectory that is a smoothed version of the polygonal line joining the midpoints. This is in effect a coarticulation model, similar in its goal to that of Bakis (1993) (see section 10.1.4). The RMS errors for the recovered articulators' positions were around 35% of their standard deviation on the average. The selection of soft regions can be seen as a conditional mean method and one would expect it to perform similarly to an MLP with errorbars. Thus, for multimodal distributions, known to exist for the articulatory-to-acoustic mapping, the mean of the Gaussian may lie in an incorrect articulatory value.

Summarising, state-of-the-art articulatory inversion in terms of accuracy is obtained by dynamic programming search of a large, carefully constructed articulatory codebook, at an enormous computational cost in storage and time. A carefully prepared ensemble of neural networks approaches codebook performance and is fast.

10.1.4 Speech recognition models that incorporate production information

Hidden Markov models and variants of them[9] are currently the unrivalled method for automatic speech recognition. Like neural networks, HMMs are complex generic statistical models that could be used for the description of many physical phenomena because the strong assumptions that they make can usually be overcome by hav-

[9] In particular, hybrid recognisers based on neural nets and HMMs, which share the advantages of the two frameworks and often deliver superior performance (Bourlard and Morgan, 1994).

ing a large number of parameters and of training data. But there is a limit to what models based on acoustic information alone can do. The performance of HMMs degrades dramatically when the speech style changes, the speaker changes or there is noise[10], all of which occur in spontaneous speech in natural environments. Several people (e.g. Rose et al., 1996, Deng et al., 1997 and Deng, 1998) have suggested the addition of e.g. linguistic or production information to acoustic models. In particular, the advantages of articulatory representations discussed earlier and the availability of articulatory data from X-ray and EMA measurements have recently led to several models that incorporate articulatory constraints (not necessarily approaching the articulatory inversion). We briefly review some here.

Zlokarnik (1995a) has provided empirical evidence that straightforward addition of articulatory information to an acoustic HMM improves recognition performance. Simultaneously recorded acoustic and articulatory data (the positions of several articulators, recorded by EMA) were combined to make up an acoustic-articulatory feature vector on a speaker-dependent isolated word recognition task with an HMM. Compared with a purely acoustic HMM, using acoustic and articulatory data both for training and testing reduced the error rate by 60%; and using articulatory measurements only during the training and implementing an acoustic-to-articulatory mapping with an MLP during the testing phase, the error rate could be reduced by a relative percentage of 18% to 25%. In another experiment, Zlokarnik (1995b) showed that of the first three time derivatives (velocities, accelerations and jerks) of the articulators' positions, accelerations perform best for ASR. Although this is surprising, since in the acoustic domain, acceleration features perform worse than static features, it confirms the importance of the role of articulatory forces in speech production.

Other variations of HMMs that use articulatory information in more sophisticated ways (e.g. forbidding transitions) were discussed in section 7.11.5. But, given the continuous nature of the temporal variation of the articulators, it seems more natural to use models of the style of Kalman filters rather than HMMs.

Bakis (1993) (see also Bakis, 1991) proposed a generic speech production model that can be seen as an acoustic recognition model, such as an HMM, augmented by an analysis-by-synthesis technique. Given an acoustic waveform to be recognised, the acoustic model proposes a hypothesis, i.e., a phoneme transcription. An abstract, deterministic articulatory model then takes as input this transcription and synthesises acoustic features that can be compared with similar features computed from the actual speech. The abstract articulatory model works as follows. First, the phonetic string is transformed into an *idealised target path* in a multidimensional Euclidean space via a table lookup; this path is piecewise constant, with abrupt transitions at phoneme boundaries. Then, this path is transformed into a *realised articulatory path* in the same multidimensional Euclidean space via convolution with a FIR filter; this path is a smoothed version of the target path and results in bounded first and second derivatives of the articulators' motion (and in correspondingly bounded forces), thus modelling coarticulation. Finally, acoustic vectors are generated from the realised articulatory path via a neural network in the form of MFCCs or any other suitable acoustic representation. Therefore, the details of the vocal tract model are left to be determined empirically from data and only the general properties are specified: coarticulation is implemented by an empirical FIR filter (with memory) rather than described in terms of masses, forces and viscous damping; and the articulatory-to-acoustic mapping is implemented by an empirical neural network (memoryless nonlinear function) rather than derived from Webster's horn equation. The components of both the acoustic model (HMM) and the articulatory model (lookup table, filter, neural net) are parametric. The parameters are adjusted from prior knowledge and empirical information to minimise the mean square error of the acoustic vectors (using conjugate gradients, the gradient being computed by the chain rule). Further prior knowledge can be included as penalty terms on the parameters in the objective function. The time-aligned phonetic transcription is given as input at training time, while at recognition time it is proposed as a hypothesis to be tested.

In this model, then, the abstract space consists of a finite collection of targets—basically, an articulatory codebook—that acts as a scaffolding on which to create smooth trajectories. An important problem is thus how to select the dimensionality of the articulatory space and the number of phonetic targets, which must be given by the user. Presumably the number of phonetic targets is related to the number of different phonemes in the language or training set under consideration, but it needs not be necessarily equal (e.g. consider the case of diphthongs and allophones). Determining the dimensionality of the articulatory space is probably a similar problem to that of determining the map dimensionality in multidimensional scaling (section 4.10.1.1).

Bakis seems never to have implemented this interesting model in practice. Recently, Richards and Bridle (1999) have implemented essentially the same model, with two minor variations: the abstract articulatory model (which they rename *hidden dynamic model*) and the acoustic model are not trained jointly, but become

[10]In section 7.10.6 we described some research on occluded speech recognition based purely on acoustic models. Also, several techniques have been devised in the speech recognition literature that partially alleviate the problem of speaker adaptation, e.g. by maximum a-posteriori estimation (Gauvain and Lee, 1994) or maximum likelihood linear regression (Leggeter and Woodland, 1995).

separate entities, with the articulatory model being used to rescore N-best lists of hypotheses; and the realised articulatory path is obtained via a second-order symmetrical (forward-backward) low-pass filter with one time constant per articulatory dimension and phonetic target. This filter, which is a simple form of Kalman smoother, controls how much to undershoot a target: the larger the time constant, the more undershooting and smoothing; in the zero limit, the transitions are discontinuous and there is no smoothing. The filter is symmetrical so that the centre of transitions occurs at phone boundaries. Thus, the articulatory model remains deterministic and does not deal with time alignment (unknown time scales in phone durations). Richards and Bridle also show the necessity of a nonlinear articulatory-to-acoustic mapping: if a linear one is used instead, the model fails to produce continuous transitions. In an evaluation of the approach with a conversational speech recognition task with the Switchboard corpus (Picone et al., 1999), improvements in terms of word error rate compared to a standard acoustic HMM only occur if, as well as the most likely hypotheses, the reference transcription is given (which is unavailable in practice). This proves that the articulatory model has information not in the acoustic HMM, although it remains to be seen how to actually use it.

Deng (1998) has proposed a stochastic approach combining the two contrasting aspects of speech: phonological (characterised by the discrete nature of phonemes) and phonetic (characterised by the continuous nature of the vocal tract). The basic structure of the model is the same as that of Bakis (1993): the phonemic string is realised in a continuous, dynamical system and nonlinearly mapped onto the acoustic, observable features. Specifically, the model consists of the following levels: (1) a language model which provides the probability for an arbitrary word sequence $p(W = w_1 \cdots w_N)$; (2) a phonological or pronunciation model based, rather than on phones (as most speech recognisers are), on overlapping features (Browman and Goldstein, 1992), which provides the probability $p(F|W)$ for a feature-overlapping pattern F of an entire utterance given its word sequence; and (3) a phonetic model which provides the probability $p(O|F, W)$ of an observed acoustic trajectory O, based on the task dynamics model of Saltzman and Kelso (1987), which is implemented with a smooth linear dynamical system (with memory) and a nonlinear articulatory-to-acoustic mapping (memoryless, such as an MLP or RBF net). Consequently, inference about the word transcription W given the acoustics O is done via Bayes' rule as in HMMs. But here the dependence of the acoustic sequence on the word sequence is more complex, involving the intermediate stage of continuous variables at the phonetic level where the articulatory constraints are applied. This complex stochastic model, containing nonlinear functions and dynamical systems, seems to be trainable for maximum likelihood given observed acoustic data by a generalised EM algorithm—a surprising fact in view of the intractability that is invariably associated with the marginalisation of complex distributions. An evaluation of a version of this model with the mentioned Switchboard corpus (Picone et al., 1999) gave very similar results to those of the hidden dynamic model implemented by Richards and Bridle (1999).

King and Wrench (1999) and Frankel et al. (2000) have modelled the articulatory trajectories with a linear dynamical model (of 4 to 13 dimensions for the hidden space) and implemented the inversion mapping with a neural network similar to that of Papcun et al. (1992) but with the addition of recurrence via context units in a hidden layer, which results in smoother trajectories. They have used TIMIT sentences with acoustic and EMA data from the MOCHA database (section 7.10.5) in a recognition task. The results using acoustics plus recovered articulators' positions were considerably worse than using acoustics plus the real articulatory data or just using acoustic HMMs. One possible reason they adduce for this is that the segmentation based on acoustic information (data forced-aligned with an HMM, which assumes that state and phone boundaries are strictly synchronised with articulatory events) differs from the segmentation based on articulator positions: they observed a slight asynchrony between changes in articulatory gestures and HMM-produced phone boundaries.

In summary, while the articulatory trajectories contain information that can be used to improve automatic speech recognition, the integration of production and acoustic models has so far not attained this goal.

10.2 Experiments with electropalatographic and acoustic data

At the time when this research was being carried out, we did not have access to articulatory data that appropriately represented the vocal tract, either synthetic (from an articulatory model) or measured (with X-ray or EMA). Instead, we used the electropalatographic (EPG) data from the ACCOR database, as in chapter 5, together with the simultaneously recorded acoustic waveform. The EPG characterises well the pattern of tongue-palate contact but is an incomplete representation of the vocal tract, and so many phonemes are indistinguishable in the EPG. For example, in fig. 5.6, the EPG labelled /æ/ can result from many other vowels (and even from silence intervals), while those of /g/ and /k/ or /t/ and /d/ are almost interchangeable. Conversely, from the nonuniqueness of the acoustic-to-articulatory mapping it is also reasonable to assume that in certain cases one phoneme may be produced with more than one different EPG. Consequently, our

"EPG-to-acoustic" mapping is many-to-many.

The EPG data present two further disadvantages:

- While the articulatory configurations are continuous in time, the EPG not always are[11]: the tongue may make or lose contact with the palate in many points at almost the same time, flipping the value of many components of the EPG vector between consecutive frames; and undersampling of the EPGs occasionally happens (e.g. fig 5.20). However, the EPG sequence generally shows a slow variation, as can be confirmed visually—e.g. fig. 5.2—or numerically—e.g. for the utterance of fig. 10.2, the maximal number of flipped components between consecutive frames is 9, occurring just once; and only for 3% (15 frames) more than 5 are flipped.
- Computationally, the EPG vectors have a high dimensionality, 62 in our case. A good vocal tract representation can be obtained with fewer than 20 dimensions (corresponding to the two-dimensional positions of 10 articulators). This results in a very high amount of missing data for the mapping acoustic $\rightarrow$ EPG (84%).

10.2.1 Experiment setup

Training data The ACCOR database (appendix B) contains data for the acoustic waveform (sampled at 20 kHz with 16 bits per sample) and the EPG frames (sampled at 200 Hz with 62 bits per sample) for several utterances and speakers. We extracted 12th-order perceptual linear prediction (PLP) coefficients (Hermansky, 1990) from the waveform (plus the log-gain, which we use in some experiments) with the RASTA program (Hermansky and Morgan, 1994); we used a window size of 25.6 ms and a step of 5 ms to match the EPG sampling rate. Thus, out data variables are:

- Articulatory variables: 62-dimensional real EPG vector (a small amount of random noise was added to the original, binary EPGs to ensure that the data set was full-rank and so avoid potentially singular covariance matrices in factor analysis), rate 200 Hz.
- Acoustic variables: 12-dimensional real PLP vector, rate 200 Hz.

The concatenation of these two vectors gives us a 74-dimensional data vector $\mathbf{t} \in \mathcal{T}$. We used 10 shuffled sentences (1, 3–5 and 9–14) for training, resulting in 8 270 data vectors, and the rest (2 and 6–8) for testing, all for speaker RK. In another set of experiments, the silence intervals at the beginning and end (but not inside[12]) of each sentence were manually removed. This resulted in 5 544 data vectors for training.

Figure 10.2 shows the acoustic waveform, log-energy (or log-gain) and PLP coefficients (1 to 12) for one utterance. The silence intervals are clearly visible in the waveform and energy graphs. The PLP coefficients are quite jagged, especially those of high order: the continuity properties of the articulators are largely lost in them.

The masks used are as in chapter 9 (fig. 9.1): P1 (P2) means EPG (PLP) present and P3–P7 are general missing data patterns.

Models We trained MLPs as universal approximators, factor analysis as linear baseline and GTM as non-linear latent variable model providing a multimodal density model:

- MLP: for each of masks P1 and P2, five MLPs with a single layer of $h = 5, 15, 25, 35, 48$ hidden units, respectively (between 387 and 3 612 parameters).
- Factor analysis: latent space of dimension $L = 9$ (789 parameters).
- GTM: latent space of dimension $L = 2$, 30×30 latent grid and 7×7 RBF grid (3 701 parameters, $K = 900$ spherical Gaussian components). Another set of experiments was run with a 20×20 latent grid (same number of parameters, $K = 400$).

The latent variables are then an abstract representation of the underlying low-dimensional manifold of the combined EPG-PLP data (fig. 10.3).

The methods used (`mean`, `dpmode`, `mlpavg`, etc.) are as in chapter 9. For `sampdp` we used $S = 10$ samples. In the continuity constraint $\mathscr{C}_1$ for `dpmode` and `grmode` we used the Euclidean distance in data space as well

[11] As in chapter 5, we consider the EPGs as real, rather than binary, vectors and define continuity in terms of the Euclidean distance (which is equivalent to the Hamming distance for binary vectors).

[12] Short silence intervals inside the utterance (such as interword pauses) could be removed by thresholding the waveform energy.

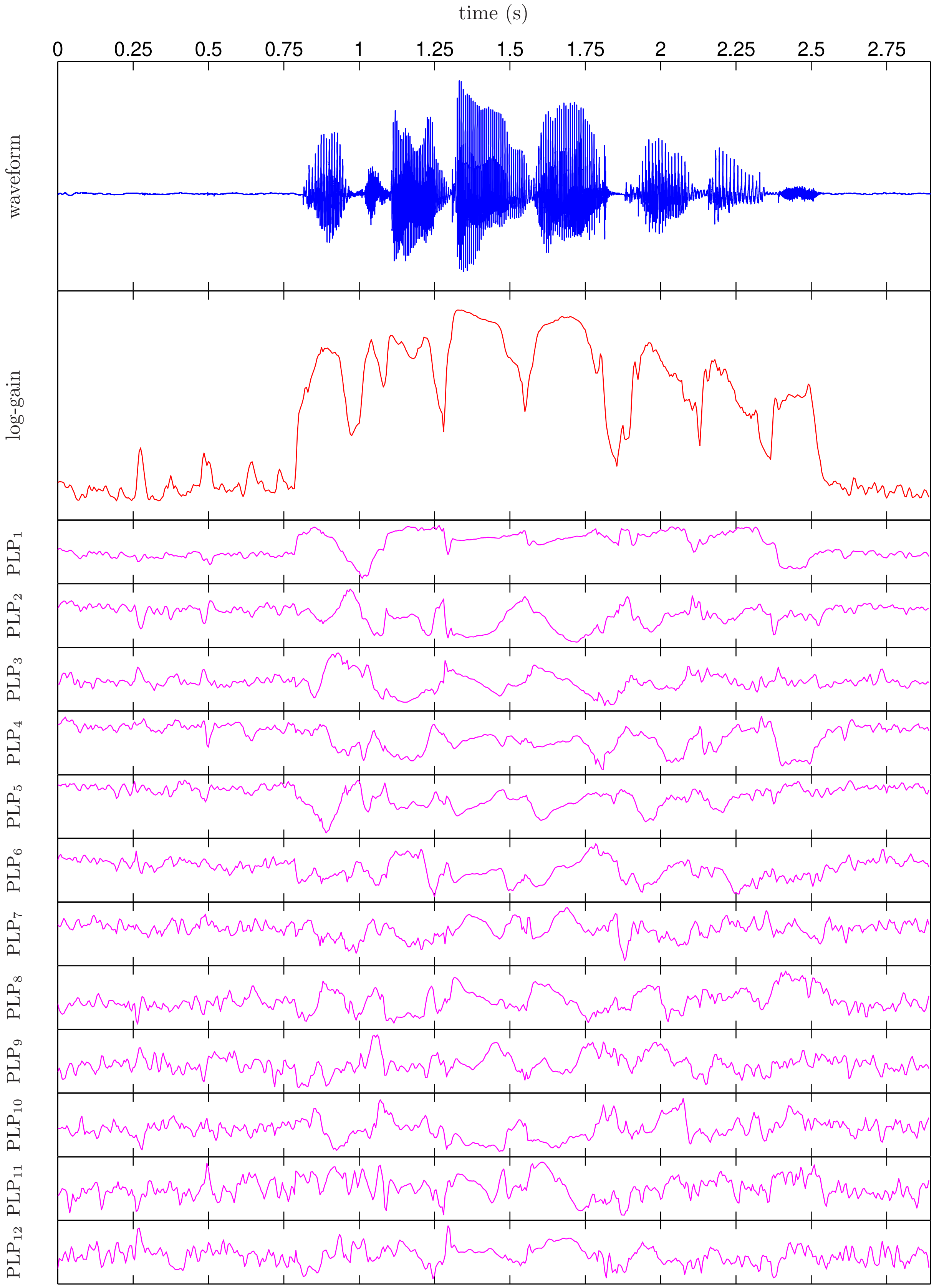


Figure 10.2: Temporal plots of the waveform, log-gain and PLP coefficients for the utterance “We tore down the outbuildings” ($N = 579$ points including the silence intervals at the beginning and end of the utterance, $N = 343$ without them). The units in the vertical axes are unimportant and thus omitted. Note the silence intervals in the waveform and log-energy plots and the jagged aspect of the PLP curves.

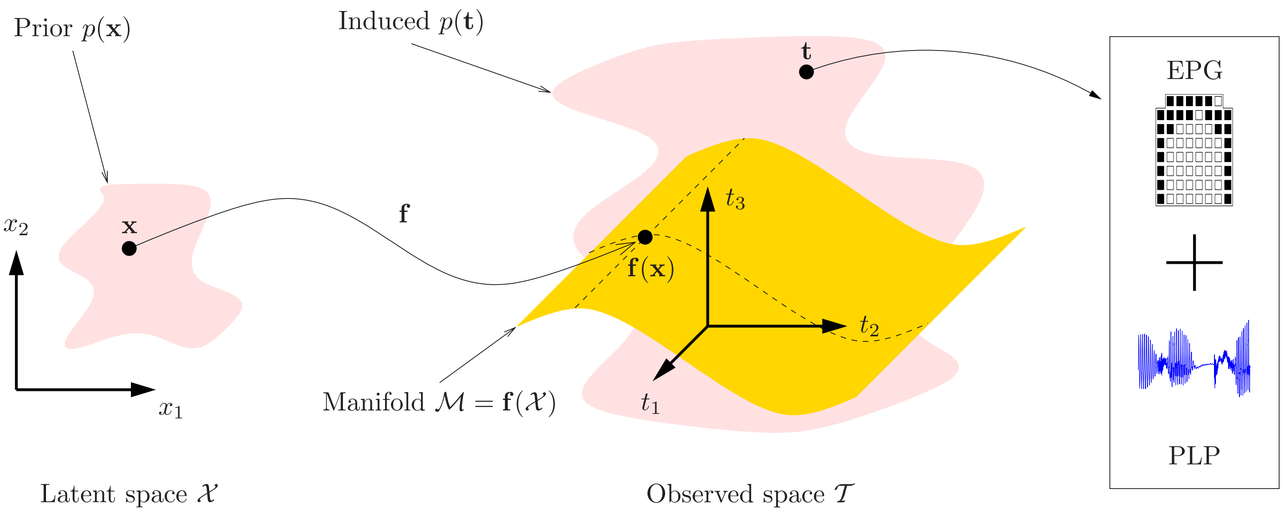


Figure 10.3: Schematic of a latent variable model where the observed data consists of EPG patterns and PLP coefficients; compare with fig. 2.6.

as various weighted Euclidean distances described later; we did not use a constraint by forward articulatory-to-acoustic mapping (resulting in the acoustic fitness term $\mathscr{F}$ in eq. (10.2)) because we do not have such a mapping: both the EPG $\rightarrow$ PLP and PLP $\rightarrow$ EPG mappings are multivalued.

Most approaches to the acoustic-to-articulatory mapping problem use sophisticated processing of the acoustic data, tailored acoustic distances and rules to apply specific treatments to certain situations (e.g. silence frames, fricatives) and very often are restricted to specific articulators or specific sound types (e.g. excluding unvoiced frames or considering only vowels) rather than spontaneous speech. Our experimental setup is very simple, partly because this is only a preliminary study and the data used, being an incomplete representation of the vocal tract, does not warrant such an effort; and partly because we believe that the approach should be equally applicable to all sound types—except silence intervals, where the vocal tract shape is fully unconstrained (although there may be some anticipatory movement).

Before describing the reconstruction results, we demonstrate multiple pointwise reconstruction.

10.2.2 Multivalued reconstruction in observed space

In fig. 10.4 we reconstruct parts of the EPG frame given other parts of it using the modes of the conditional distribution for the GTM model ($K = 400$). Note how the reconstructed pattern is slightly different when the left half is given than when the right half is given, revealing asymmetry in the tongue movement. When the bottom half is given, the distribution for the top half happens to be multimodal, with several patterns (corresponding to vowels and alveolars) becoming possible. This example demonstrates that reconstruction via the conditional distribution can be applied to any combination of present and missing variables; if an observed variable is neither in the present nor in the missing subset, it is integrated over, as the PLP variables are in this example.

10.2.3 Reconstruction results

The reconstruction results were similar for all the test (and also training) utterances. We give detailed tables and graphs only for the utterance “We tore down the outbuildings” (number 6), which is a typical representative. From the results for all utterances, we can draw the following conclusions:

- `fa` performs almost as well as the `mean`, independently of the mask: the error of `fa` was lower than that of the `mean` 35% of the times and was usually not much higher the rest of the times.
- MLP methods: `mlpbest` tended to perform better than `mlpavg` and this better than `mlpdp`, but there was little difference in reconstruction error. For mask P2 the MLPs performed always quite worse than `cmode`; for mask P1, both `cmode` and the MLPs performed very similarly. For both P1 and P2, the MLPs performed nearly always better than the `mean`.

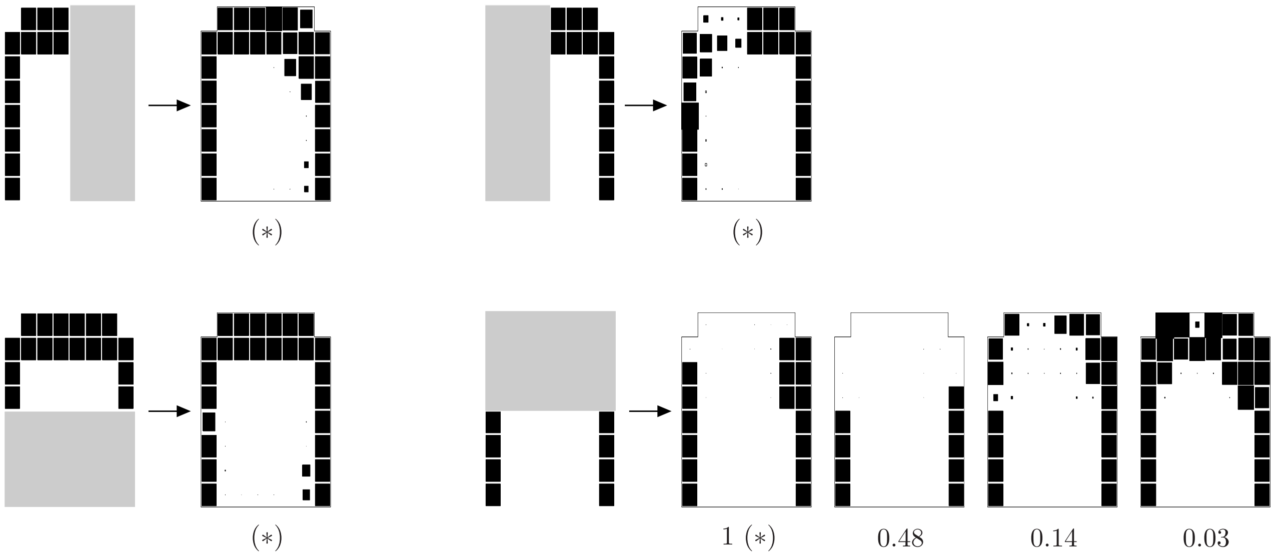


Figure 10.4: Reconstruction of single EPG frames. We use the GTM model ($K = 400$) to compute the distribution of the EPG part greyed out (the missing values) conditional on the EPG part which is not greyed out (the present values). Its modes are given to the right of the arrow, labelled with their normalised probability if there is more than one mode. In the case where the top half was given, there were additional modes of very small normalised probability value. In all four cases, the mean (marked $*$) coincided approximately with one of the modes. Note the left-right asymmetry in most of the reconstructed EPGs, even when the present part was symmetric.

- `gmode` was almost always worse than `dpmode` but better than `rmode` (around 20% lower reconstruction error).
- `cmode` was always much better than `mean` and `fa`.
- `dpmode` was almost always worse, and considerably so, than `mean` for mask P2; very similar for P1; and always better, and considerably so, for P3–P7, being only second to its bound `cmode`.
- `sampdp` was nearly always quite worse than `dpmode`; `meandp` was nearly always equal to `dpmode`.
- `grmode` was almost always worse than `dpmode`, but only slightly so (around 10% higher reconstruction error).

In summary, `dpmode` is always the best method with random missing data patterns (P3–P7) but has problems with some of the regression problems, particularly P2 (PLP $\rightarrow$ EPG). For this mask, the number of modes per point can be very high (a significant fraction of the number of components K, see fig. 10.6; this also increases enormously the reconstruction time, see later) which indicates that the GTM model is not smooth. While the error of `dpmode` is sensitive to the number of components K (see below), the fact that the MLPs performed better than the `mean` and that even `fa` was almost as good as the `mean` suggests that the GTM model estimate used is quite poor. This is probably due to the low latent dimensionality assumed ($L = 2$, for computational reasons), which is certainly quite lower than that of the speech signal (usually considered around 5). Although even a poor model like this still contains information useful for reconstruction, as evidenced by the low error of `cmode`, the presence of lots of spurious modes makes it impossible to use. Observe in table 10.1 the very short $\mathscr{C}_1$-value for `dpmode` (85.84) in mask P2 compared to that of the original utterance (255.75).

The results also confirm that, in any case, the methods `gmode`, `rmode`, `grmode` and `sampdp` perform badly.

Further experiments We performed further experiments with the following results:

- Rounding the reconstructed EPG values to $\{0, 1\}$ always increases considerably the error (unsurprisingly).
- Using a GTM model with fewer components ($K = 400$) generally increases the error of most mode-based strategies (particularly `cmode`) but does not affect much the `mean` and `gmode`. It does not seem to affect positively or negatively to `sampdp` and `rmode`, which perform as badly as with $K = 900$. This confirms that the mode-based schemes are more sensitive to the goodness of the model than the `mean`.

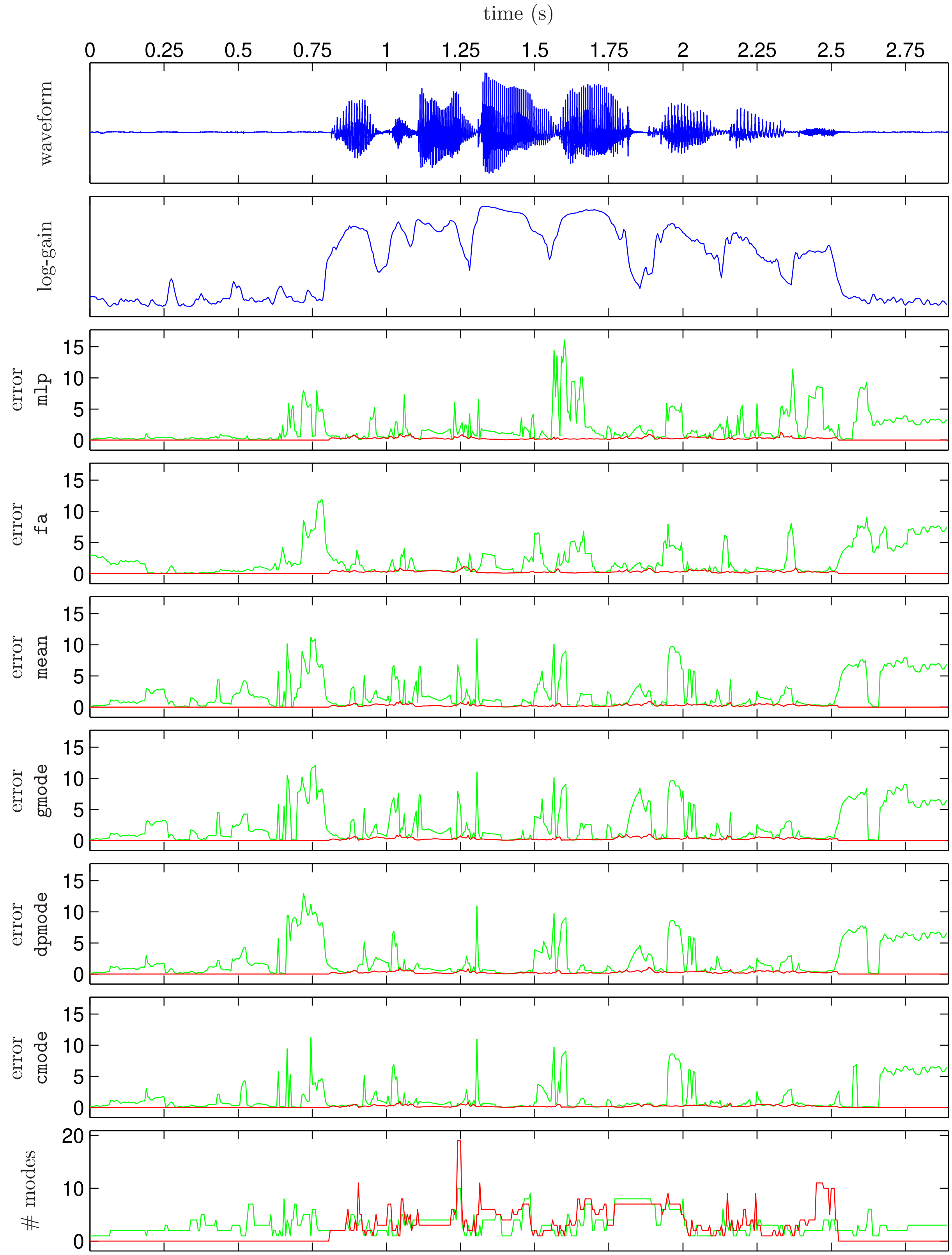


Figure 10.5: Reconstruction error as a function of the time (or sequence index n): mapping EPG $\rightarrow$ PLP (mask P1). Utterance "We tore down the outbuildings" ($N = 579$ points) reconstructed by models trained without (red) and with silence and log-gain (green). *Top two graphs*: waveform and log-gain. *Bottom graph*: number of modes of the conditional distribution $t_{\mathrm{PLP}}^{(n)}|t_{\mathrm{EPG}}^{(n)}$. *Rest of graphs*: reconstruction error $\left\|\mathbf{t}^{(n)} - \hat{\mathbf{t}}^{(n)}\right\|^2$ for the respective method. The GTM model has $K = 900$.

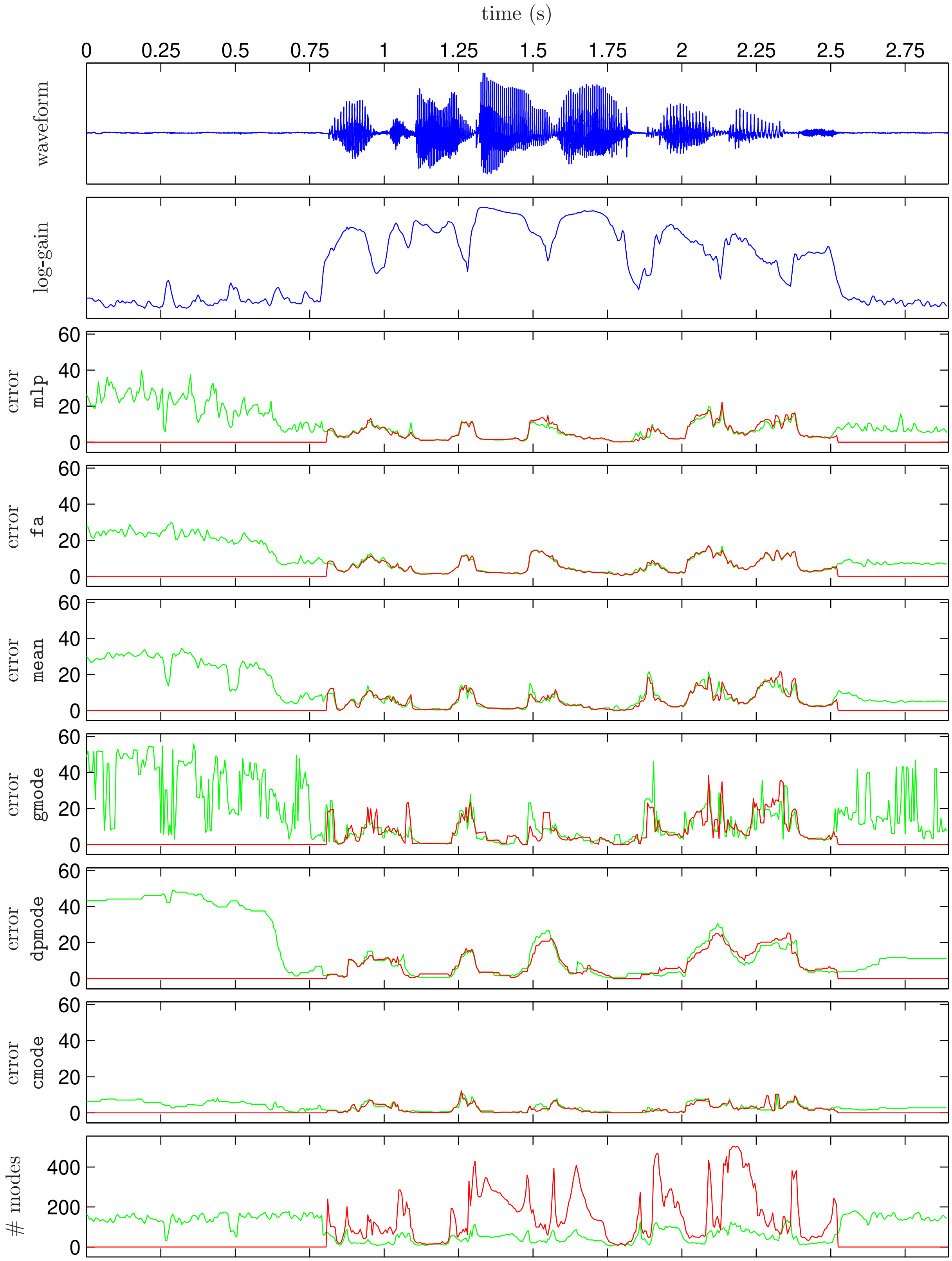


Figure 10.6: Reconstruction error as a function of the time (or sequence index n): mapping PLP → EPG (mask P2). Utterance "We tore down the outbuildings" ($N = 579$ points) reconstructed by models trained without (red) and with silence and log-gain (green). All graphs and models as in fig. 10.5 except for *bottom graph*: number of modes of the conditional distribution $t^{(n)}_{\mathrm{EPG}}|t^{(n)}_{\mathrm{PLP}}$. Note the different scales in the vertical axes; and how, for all methods, the reconstruction errors usually match the silence intervals (at the beginning, end or inside the utterance).

Average squared error $\frac{1}{N}\sum_{n=1}^{N}\left\|\mathbf{t}^{(n)}-\hat{\mathbf{t}}^{(n)}\right\|^2$

Mask	Factor	MLP			GTM with $K = 900$							
(% missing)	analysis	`mlpbest`	`mlpavg`	`mlpdp`	`mean`	`gmode`	`rmode`	`cmode`	`grmode`	`dpmode`	`sampdp`	`meandp`
P1(16%)	0.3185	0.3063	0.3102	0.2977	0.2843	0.3040	0.2983	0.2399	0.2784	0.2821	0.4726	0.2821
P2(84%)	5.7824	5.6305	5.4522	5.6407	5.5576	7.8321	9.2272	2.4032	9.3540	8.4136	8.9014	8.4136
P3(75%)	2.9885				3.1947	3.6480	4.0173	2.2315	2.7525	2.4717	3.8426	2.4717
P4(51%)	1.7364				1.8605	1.9608	2.1005	1.5297	1.6673	1.6100	2.1844	1.6098
P5(25%)	0.8000				0.8151	0.8498	0.8832	0.6948	0.7436	0.7178	0.8843	0.7178
P6(51%)	3.1221				3.0183	4.2263	5.0106	1.4511	1.8598	1.6953	3.5229	1.6953
P7(7%)	0.3889				0.4398	0.5347	0.5477	0.2100	0.2434	0.2208	0.4307	0.2207

Value of constraint $\mathscr{C}_1 = \sum_{n=1}^{N-1}\left\|\hat{\mathbf{t}}^{(n)}-\hat{\mathbf{t}}^{(n+1)}\right\|$ (original sequence: 255.75)

Mask	Factor	MLP			GTM with $K = 900$							
(% missing)	analysis	`mlpbest`	`mlpavg`	`mlpdp`	`mean`	`gmode`	`rmode`	`cmode`	`grmode`	`dpmode`	`sampdp`	`meandp`
P1(16%)	228.68	231.17	231.40	230.23	231.63	232.53	258.75	234.31	230.41	230.03	323.65	230.03
P2(84%)	125.47	122.53	152.82	120.42	155.89	363.73	825.86	210.18	120.01	85.84	614.36	85.84
P3(75%)	513.14				562.12	632.38	701.99	428.48	426.06	389.70	691.67	389.73
P4(51%)	478.28				490.53	511.08	543.52	437.50	438.50	424.78	588.05	424.75
P5(25%)	428.04				421.18	429.22	443.11	390.51	390.65	386.46	458.10	386.48
P6(51%)	501.34				490.74	609.49	801.95	380.64	361.81	346.91	643.00	346.91
P7(7%)	270.11				272.98	282.74	304.86	262.21	261.42	259.42	303.01	259.40

Table 10.1: Reconstruction results for the EPG-PLP mapping problem: average squared error and global constraint value (trajectory length) for the utterance "We tore down the outbuildings" with $N = 343$ points ($N = 579$ including silence), without start and end silence intervals and without log-gain. $\{\hat{\mathbf{t}}^{(n)}\}_{n=1}^{N}$ is the reconstructed utterance by the corresponding method. For masks P1 and P2, `mlpbest` corresponded to the MLP with 5 hidden units. For masks P3–P7, MLPs are not applicable.

Problem type (mask)	Low error $\longrightarrow$ High error
EPG $\to$ PLP (P1)	`cmode` $\approx$ `mlp` $<$ `dpmode` $\lesssim$ `mean` $\lesssim$ `gmode` $\lesssim$ `fa` $<$ `rmode`
PLP $\to$ EPG (P2)	`cmode` $\ll$ `mlp` $<$ `mean` $\lesssim$ `fa` $<$ `dpmode` $<$ `gmode` $<$ `rmode`
General (P3–P7)	`cmode` $<$ `dpmode` $\ll$ `mean` $\lesssim$ `fa` $\approx$ `gmode` $<$ `rmode`

Table 10.2: Reconstruction results for the EPG-to-PLP mapping problem: summary comparison of the methods in terms of reconstruction error. Compare with table 9.2.

- Removing the gain from the density model and the start and end silence intervals from the data reduces the error of all methods considerably (compare the curves in figures 10.5 and 10.6 for reconstruction including silence and log-gain with those not including them). The reason is that the gain is independent of the EPG variables, reflecting only the amplitude of glottal excitation; and that during silent intervals (when the glottal excitation is zero) the EPG vectors are unconstrained by the acoustics. This is equally applicable to any other articulatory variables.

  From figures 10.5 and 10.6 we can observe that the effects of removing the silence intervals are:

  – To reduce the reconstruction error for P1 (by half), although not for P2.

  – To increase the number of modes, due to the Gaussian components that were located on areas of space corresponding to silence.

  Figure 10.6 shows that, after the silence intervals at the beginning and end have been removed, large errors mainly occur at the short silence intervals inside the utterance. Such intervals can be detected by thresholding the gain and eliminated from the reconstruction process (but we did not do this).

  Also, notice in fig. 10.6 how the reconstruction error curves for both models (trained with / without silence and log-gain) nearly coincide for all methods during the non-silence intervals. The explanation is as follows. For the models trained with the log-gain and with silence, the Gaussian components basically go to one of two areas of data space, corresponding to silence and non-silence intervals, respectively; while for the models trained without the log-gain and without silence, all Gaussian components model only non-silence intervals. Given the gain and the PLP coefficients, the models trained with the log-gain and with silence can reconstruct the EPG vectors in non-silence intervals practically as well as the models trained without the log-gain and without silence. That is, in the latter case the mapping is PLP $\to$ EPG while in the former case it is (PLP, gain) $\to$ EPG, and the gain is acting as a discriminative variable. Thus, both models perform similarly in non-silence intervals. This does not happen in fig. 10.5 because there only the EPG variables are present, not the gain. That is, the models trained with the log-gain and with silence cannot know anymore, given the EPGs alone, whether the interval corresponds to silence or not.

- Using various weighted Euclidean distances (e.g. zeroing the weights of the PLP or the EPG variables) did not alter the errors in general (less than 5%). Using a Mahalanobis distance based on the covariance matrix (either that of the whole training set for all utterances or a separate one for each utterance), which is equivalent to sphering the data, did not improve the results either[13]. One would have expected that using zero weight for the PLP vectors (i.e., a purely articulatory distance), which are not very continuous, would have reduced the error. Therefore, the fact that the results with `dpmode` are practically insensitive to the distance used again indicates that the proliferation of modes caused by the nonsmooth GTM model makes it impossible to select the correct ones.

10.2.4 Computational performance

Table 10.3 and figure 10.7 give the reconstruction time for a 1.73-second utterance (that of fig. 10.2, without start or end silence intervals) for the methods `mean`, `grmode` and `dpmode`. The bottleneck of the mode-based methods is the mode-finding procedure run at every frame, which takes longer the more missing variables there are (particularly when the EPG variables are missing). The longest times correspond to P2, where the 62 EPG variables are always missing; for a real acoustic-to-articulatory mapping problem, the time would be far

[13]Neither scaling nor sphering solve in general the problem of using different units for each variable (fig. 2.8). This needs a diagonal GTM model (section 2.12.4), which accounts for different scales along each axis in the noise locally (although such scales are constant over the data space).

Reconstruction time (seconds)

Mask (% missing)	mean	threshold = $\frac{1}{100}$ grmode	dpmode	threshold = $\frac{1}{20}$ grmode	dpmode	threshold = $\frac{1}{10}$ grmode	dpmode	threshold = $\frac{1}{5}$ grmode	dpmode
P1(16%)	1.83	1.84	1.89	1.58	1.62	1.51	1.52	1.44	1.46
P2(84%)	1.45	523.58	527.38	202.53	204.50	112.06	112.68	54.76	53.78
P3(75%)	1.45	57.01	57.23	37.22	37.50	29.76	29.88	21.99	23.03
P4(51%)	1.59	6.13	6.20	4.42	4.47	3.68	3.72	2.84	2.87
P5(25%)	1.74	2.21	2.24	1.77	1.82	1.60	1.62	1.45	1.47
P6(51%)	1.61	301.77	300.53	117.08	120.66	66.18	66.60	31.11	31.25
P7(7%)	0.42	71.95	72.45	69.35	67.66	64.74	66.66	62.34	62.48

Reconstruction error $\frac{1}{N}\sum_{n=1}^{N}\left\|\mathbf{t}^{(n)}-\hat{\mathbf{t}}^{(n)}\right\|^2$

Mask (% missing)	mean	threshold = $\frac{1}{100}$ grmode	dpmode	threshold = $\frac{1}{20}$ grmode	dpmode	threshold = $\frac{1}{10}$ grmode	dpmode	threshold = $\frac{1}{5}$ grmode	dpmode
P1(16%)	0.2919	0.2951	0.2935	0.2906	0.2929	0.2895	0.2941	0.2903	0.2960
P2(84%)	5.8094	9.9276	9.9251	10.1307	9.5446	9.5212	8.8867	8.1065	8.1206
P3(75%)	3.1482	2.7991	2.6490	2.8705	2.7278	2.8989	2.7895	2.9353	2.8415
P4(51%)	1.9147	1.7337	1.6942	1.7943	1.7679	1.8210	1.8048	1.8711	1.8500
P5(25%)	0.8694	0.7840	0.7635	0.8130	0.8008	0.8288	0.8227	0.8414	0.8359
P6(51%)	3.1562	1.8389	1.7431	2.0661	1.9883	2.4008	2.5410	2.7907	2.9043
P7(7%)	0.4238	0.2788	0.2432	0.3102	0.2655	0.2997	0.2653	0.2696	0.2835

Table 10.3: Reconstruction time and error for the EPG-PLP mapping problem for the utterance "We tore down the outbuildings" with $N = 343$ points (1.73 seconds), without start and end silence intervals and without log-gain, with model $\mathrm{GTM}_{K=400}$. The values are given for methods `mean`, `grmode` and `dpmode`, for various values of the threshold for removal of low-probability components; the higher the threshold, the larger the error but the less computing time. The computer used was a Compaq XP1000 workstation with an Alpha 21264 CPU at 667 MHz and 768 megabytes of RAM memory. For the $\mathrm{GTM}_{K=900}$ model, the reconstruction times were generally 2–6 times longer and the errors 5% lower on the average (using $\mathrm{GTM}_{K=400}$ as baseline).

shorter (roughly comparable to those of P4 or P5), since only around 20 variables or fewer would be missing. The extra time spent by the `dpmode` in the dynamic programming search over that of the `grmode` is negligible. For small amounts of missing data, the `dpmode` takes little longer than the `mean`.

The reconstruction times can be considerably reduced (down to one tenth for P2 and P6) with a mild increase in reconstruction error (less than 17%) by raising the threshold for removal of low-probability components (section 8.2.3): the default threshold of $\frac{1}{100}$ means that all components of the conditional distribution whose posterior probability is less than $\frac{1}{100}$th of the largest posterior probability are removed. An appropriate value of this threshold should be carefully determined for the application under consideration. Note how, contrarily to all other masks, for P2 the error consistently *decreases* when raising the threshold! This indicates a very large number of spurious, low-height modes caused by a nonsmooth model.

10.2.5 Discussion

We have presented experiments for a mapping inversion problem similar to that of the acoustic-to-articulatory mapping, using EPG data for the articulatory variables and PLP coefficients for the acoustic ones, with a continuity constraint based on the Euclidean distance for both EPG and PLP variables and no further preprocessing. The results for general missing data patterns have been similar to those of the previous chapter, with the `dpmode` method attaining the lowest reconstruction error of all methods. A major difference has occurred for the mapping PLP $\rightarrow$ EPG, where the `dpmode` has performed worse than `mean`, `mlp` and `fa`, due to several factors: a poor, nonsmooth density model derived from a GTM model with too low dimensionality of the latent space and using a spherical rather than diagonal noise model (partial improvements should be expected if using a mixture of diagonal Gaussians); a very high dimensionality of the missing variables (62), far higher than that of the real acoustic-to-articulatory mapping problem; and the fact that the EPGs

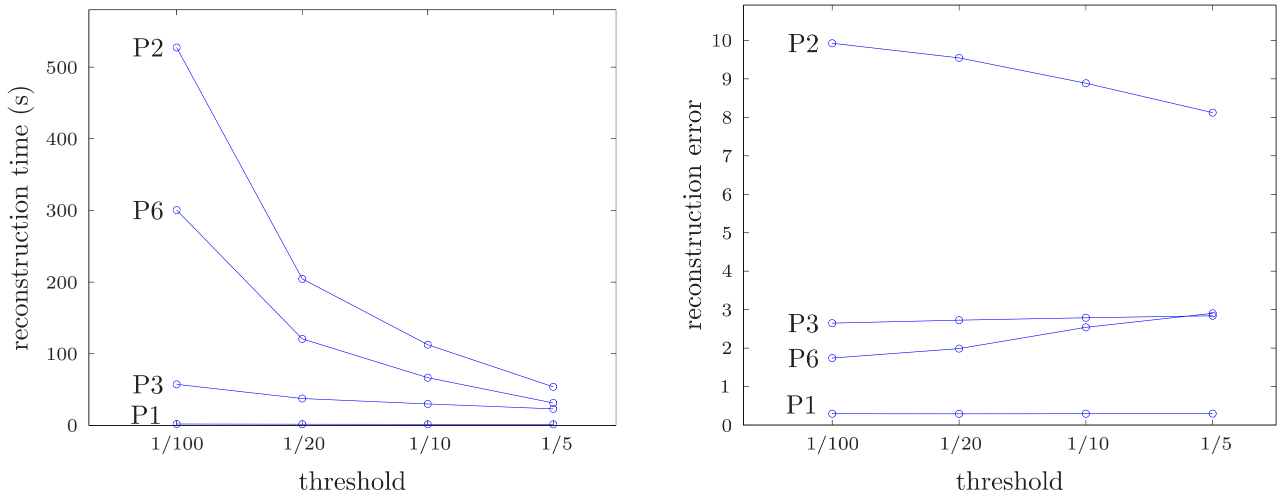


Figure 10.7: Reconstruction time and error for the EPG-PLP mapping problem for the same utterance as in table 10.3. For clarity and to highlight the dependence of the reconstruction time (left graph) and reconstruction error (right graph) on the threshold for removal of low-probability components, only the curves for method `dpmode` and masks P1–P3 and P6 are shown. When raising the threshold, the reconstruction time decreases considerably, while the reconstruction error increases only slightly or, for mask P2, even decreases.

are largely undetermined by the acoustics for many phonemes. These results are not representative of what should be expected for a genuine acoustic-to-articulatory mapping, which we leave for future research using measured articulatory data that has become recently available (section 7.10.5). The reconstruction times for the different masks give an indication of the performance as a function of the amount of missing variables for a high-dimensional, heavily unconstrained problem.

More generally in the scope of the acoustic-to-articulatory mapping problem and of speech research, we highlight the following properties of our method:

- The method does not break the nonuniqueness of the mapping by learning one branch of it or an average of the branches: it directly addresses the nonuniqueness by representing all branches and selecting the appropriate one only at reconstruction time.

- Both acoustic and articulatory data are used for training, but only the acoustics are necessary for testing. This allows the combination of the method with acoustic models for ASR, e.g. by using an articulatory-acoustic HMM, where the articulatory features are filled in by our method given the acoustic ones.

- A well known problem with codebook-based methods for the acoustic-to-articulatory mapping is that the codebook vectors must be feasible articulatory configurations. In principle, the same problem can happen in our approach: a mode of the conditional distribution might correspond to an unfeasible articulatory configuration. This would be a symptom of a poor density model that assigns high probability mass to areas that should be devoid of data: perhaps due to a bad local minimum or to under- or overfitting. Whether this is a serious problem can only be ascertained in a practical implementation with articulatory data.

- The relative freedom that characterises noncritical articulators to adopt different configurations should be accounted for by our method as a multiplicity of modes in the corresponding conditional distribution. If the noncritical articulators are really widely unconstrained for a given phonetic context then recovering the actual configuration is not important, but if they are constrained to some extent by the anticipation of a future, critical stage so that the configuration is determined by the context, then the global constraint (continuity, muscle effort, etc.) may select the correct mode.

- Speech research has concentrated so far on one of two situations: either missing articulatory variables and present acoustic variables (articulatory inversion, section 10.1) or partially present acoustic variables disregarding articulatory variables (recognition of occluded speech, section 7.10.6). The combination of both ("robust articulatory inversion" or even "robust ASR with articulatory information") has not been investigated. For the purpose of reconstruction, our method can integrate both and any other combination seamlessly without any modification.

- If using a latent variable model for the combined articulatory and acoustic data, it may be argued that, since the articulatory variables (causally) determine the acoustic ones, one could use the articulatory variables as latent variables; the mapping $\mathbf{f}$ from latent to observed space would then be interpretable as the articulatory-to-acoustic mapping. The reason for not doing this is that the chosen articulatory variables will also have a low intrinsic dimensionality resulting from the dependence of each articulator on the others. A unified representation of the articulatory-acoustic data is obtained by using a common set of abstract latent variables—which might be interpretable in terms of articulatory gestures after all.

Chapter 11

Conclusions

We conclude the thesis with some comments about our general approach, a discussion of the contributions of the thesis and a suggestion for future work.

11.1 General approach of the thesis

Pattern recognition This is a thesis on pattern recognition: the aim of the methods proposed or reviewed in it is to find structure or information about a system by learning a flexible model from data observed in that system, with or without knowledge of the domain, by using generic properties that most physical systems have (e.g. dependence between variables, such as continuity). Such algorithms may not tell about the true, underlying nature of the system, but they can simulate the observed behaviour—just as we can very accurately simulate one period of the function $\sin x$ with a fifth-degree polynomial or the planets' trajectories with a system of epicycles (as in Ptolemy's geocentric model).

This does not mean that we reject domain knowledge. On the contrary, its judicious inclusion is desirable (although sometimes a pattern recognition algorithm may work better in practice than a physically-derived one). It does not mean either that the abstract representation of a pattern recognition model (e.g. its parameter estimates) are never interpretable in physical terms. For example, if carefully designed to avoid non-identifiability (e.g. by regularising it) a latent variable model can indeed be interpretable.

Probabilistic models Most of the models discussed in this thesis, in particular latent variable models, are probabilistic (also called stochastic or generative), i.e., they define a probability density for the data. Probabilistic models have well known advantages over non-probabilistic ones:

- There is a well-defined objective function, the likelihood, which allows the development of principled estimation techniques that optimise the objective function (such as gradient methods or, better still, EM algorithms); and to compare the method with other probabilistic methods and to construct statistical tests. Ideally, the likelihood should be combined with a prior distribution on the parameters for Bayesian model comparison.
- To combine several probabilistic methods in a mixture, which is again a probabilistic model and has the power of building a complex global model out of simple local ones.
- To predict any variable(s) as a function of any other variable(s) by using conditional probabilities. That is, arbitrary classification, regression and missing data reconstruction (see chapter 7).

Therefore, probabilistic models are perhaps the most principled way to deal with the stochastic variability that appears in nearly all practical applications. However, they have disadvantages with respect to non-probabilistic models, mainly that it is harder to obtain a density model than a function and that the computations are usually more complex.

Bayesian analysis One step further in probabilistic modelling is Bayesian analysis, where probability distributions are defined not just for the latent and observed variables but also for the parameters. Inference is then carried out by Bayes' theorem as usual but requires integrating over the parameters' distributions. Perhaps the greatest benefit of Bayesian methods is in model selection. However, their full application is very complicated theoretically and computationally—one just needs to see how complex it can get for the simple case of a univariate Gaussian mixture using a hierarchical prior model (Richardson and Green, 1997). The problem is in the marginalisation of complicated multivariate densities, just as in section 2.4 but now including the parameters as well. Besides, approximate Bayesian schemes are of debatable efficacy when compared with other model selection methods in real applications. Thus, while we subscribe in spirit to the Bayesian framework, we have refrained from using it in this thesis because of these difficulties: our models use point estimates for the parameters obtained by maximum likelihood (possibly with regularisation terms) and the only random variables are the latent and observed variables.

11.2 Contributions of this thesis

The main contributions of this thesis are:

- Some extensions to the theory of latent variable models (chapter 2):
  - The investigation of the definition and properties of the dimensionality reduction and reconstruction mappings in continuous latent variable models and mixtures of them: continuity and why the dimensionality reduction mapping is not the inverse of the mapping from latent to observed space.
  - Some theoretical results for continuous latent variable models regarding identifiability, independence relations and entropy.
  - The extension of the GTM model to a diagonal noise model and the derivation of an associated EM algorithm for maximum likelihood estimation (section 2.12.4). A diagonal noise model is useful when the observed variables have different local scales of dispersion, which cannot be accounted for by sphering the data. At the time of this writing, a Matlab implementation is not yet available.
- Two contributions to Bernoulli mixtures (chapter 3):
  - The proof that the log-likelihood surface has no singularities of infinite value, which implies that the EM algorithm for Bernoulli mixtures (known since Wolfe, 1970) converges to proper maxima from almost every starting point. The existence of infinite-value or boundary singularities in other mixtures makes estimation difficult in practice, as is well known, and we have given empirical examples for mixtures of factor analysers (section 5.4.4.1) and Gaussian mixtures (section 9.2.3).
  - Empirical evidence and some theoretical discussion that, while Bernoulli mixtures are non-identifiable in theory (and so different values for the parameters can give rise to exactly the same distribution) they are usually identifiable in practice. This restores confidence in interpreting parameters estimated from samples for this type of mixtures. We also remind the fact that, for models which are identifiable in theory (such as Gaussian mixtures), practical identifiability may not always hold.
- The application of latent variable models to electropalatographic (EPG) data, specifically the following results (chapter 5):
  - That linear dimensionality reduction methods (factor analysis, PCA) can be used to adaptively derive EPG indices. Some of these indices roughly coincide with previously proposed, non-adaptive EPG indices for data reduction, but others are new and better reflect the pattern of tongue-palate contact of the speaker under consideration.
  - That a two-dimensional, nonlinear model (GTM) performs better than any of the other models (which use higher dimensions) in terms of log-likelihood, reconstruction error and visualisation. We propose the two-dimensional representation of GTM as a map of EPG categories that can be used to investigate articulatory dynamics (e.g. transition between categories, which can result in undersampling discontinuities) or in speech therapy and language learning.
  - That the intrinsic dimensionality of the EPG data is probably lower than previous estimates that set it to 5 or more dimensions, but that the intrinsic manifold is nonlinear. This is relevant for articulatory phonetics and the acoustic-to-articulatory mapping problem.
  - That continuous latent variable models perform well with EPG data despite the binary character of the latter.

- The definition of the problem of missing data reconstruction for sequential data and the proposal, evaluation and probabilistic interpretation of an algorithm for its solution (chapter 7). As far as we know, the definition of the general problem is new; in fact, we have not yet found an actual practical instance of interest of it. Similar but not identical problems have been approached in the past by others: regression and classification with missing inputs (e.g. recognition of occluded speech) and inference with missing data. We have also suggested and partially investigated the application of the method to some specific practical problems. A particularly important novel feature of our method is the emphasis in preserving the nonuniqueness of the plausible values of a missing variable (rather than ignoring it) and its implementation as the modes of a conditional distribution.

- The application of the missing data reconstruction method to a toy problem, a robot arm inverse kinematics problem and an EPG-acoustic mapping problem (chapters 9 and 10). Given that these problems are either of very small size or of low practical interest, they must be regarded as evaluations of the method, rather than solutions of real-world problems.

- The introduction of the concept of sparseness of a distribution (sections 7.3.1, 7.12.4 and 8.5), of particular interest for conditional distributions, to express the degree to which the conditioned-on variables determine the conditioned variables; the investigation of conditions that a quantitative measure of sparseness should satisfy; the proposal of entropy as an acceptable, though not perfect, sparseness measure; and the derivation of lower and upper bounds for the entropy of a Gaussian mixture (which is not analytically computable).

- Several results about the modes of Gaussian mixtures (chapter 8), as follows. The proposal and partial proof of a conjecture for a type of finite mixtures of Gaussian distributions: that the number of modes is upper bounded by the number of components in the mixture and that the modes are contained in the convex hull of the component centroids. The development and analysis of two iterative algorithms for exhaustive mode finding in Gaussian mixtures inspired by that conjecture: one based on gradient-quadratic search and the other one on fixed-point search, both of proven efficacy (from toy problems that can be visualised to large problems of 900 components and 75 dimensions). The derivation of approximate error bars (confidence intervals) for the location of each mode.

  Apart from their use in our reconstruction method, the mode-finding algorithms are generally applicable to other problems, such as clustering (e.g. determination of subclustering within galaxy systems) or Bayesian analysis (scrutiny of the posterior modes).

- Some of the review material is of interest by its novelty, extension or because it straddles different fields:

  - Chapter 2 is an up-to-date, unifying review of continuous latent variable models: the definition in terms of prior distribution in latent space, mapping from latent to observed space and noise model allows a clear comparison of all known types of continuous latent variable models. The only existing treatments of latent variable models have been given in the statistics literature; these include specialised books on factor analysis (for example, Harman, 1967; McDonald, 1985; Basilevsky, 1994), specialised books on latent variable models (for example, Lazarsfeld and Henry, 1968; Everitt, 1984; Bartholomew, 1987; Berkane, 1997) or chapters of books on multivariate analysis (for example, Mardia et al., 1979; Krzanowski, 1988; Morrison, 1990). Such literature deals with both discrete and continuous observed and latent variables, but the treatment of the case where both observed and latent variables are continuous is always restricted to linear-normal methods (factor analysis, non-probabilistic PCA) and with an emphasis on interpretability rather than dimensionality reduction. Our review fills this gap by dealing with recent models developed in the statistical learning literature (GTM, independent component analysis, independent factor analysis) and by emphasising the dimensionality reduction ability of continuous latent variable models and mixtures of them.
  - Chapter 4 is quite comprehensive, bringing together aspects usually found in different fields, such as the geometry of high-dimensional Euclidean spaces or space-filling curves. The discussion of multidimensional scaling methods, based on either straight-line or geodetic distances, is particularly extensive. The chapter also introduces an idea for using discrete variables for dimensionality reduction.
  - In chapter 6, our goal is not to give a comprehensive treatment of inverse problem theory (which is well covered in, e.g. Tarantola, 1987; Parker, 1994; Engl et al., 1996; Scales and Smith, 1998; Snieder and Trampert, 1999) but to introduce it to the pattern recognition community, where it is

often taken to mean mapping inversion; to show when an inverse problem reduces to a mapping inversion problem (namely, when we have locally independent inverse problems); and to show the links between latent variable models and Bayesian inverse problem theory.

  – The review of the acoustic-to-articulatory mapping problem of section 10.1 updates that of Schroeter and Sondhi (1994). Of particular interest is the survey of recent speech models that incorporate production information. We also give a view of the motor theory in terms of latent variables.

- The Matlab implementation of several of the algorithms and tools (see appendix C for details).

11.3 Directions for further work

- Specifically for latent variable model theory:
  – Development of nonlinear models with locally varying noise model, i.e., where the parameters of the noise model depend on the latent variables. This is desirable because using a fixed noise model results in an intermediate covariance that does not fit the data well in some areas of data space. However, this extension is difficult analytically and besides would suffer from two computational problems: the appearance of infinite-value singularities in the log-likelihood surface, to which the EM algorithm can be easily attracted; and a larger demand for training data, particularly in models that sample the latent space (such as GTM).
  – The development of latent variable models for periodic variables. This is desirable because modelling a periodic observed variable with a nonperiodic latent variable results in a discontinuous dimensionality reduction mapping. An application of this would be a dimension-reduction model of maps of mammal visual cortex (Swindale, 1996), where one of the "observed" variables, the orientation of a line in the visual field, is periodic in $[0, \pi)$. GTM can be extended to deal with periodic variables (Chris Williams, pers. comm.).
- Integration of latent variable models, which have the ability for dimensionality reduction and missing data reconstruction, with sequence or time series models. This is something we have not actually done in this thesis: our application of latent variable models to sequential data—dimensionality reduction of EPG data in chapter 5 and missing data reconstruction of various types of trajectories in chapters 9 and 10—has not modelled its temporal evolution. Such integration would be particularly useful in automatic speech recognition, which can be improved by reconstructing the articulatory configuration and adding it to the acoustics.
- Development of practical, reliable algorithms for model selection regarding the dimensionality of the latent space—a problem related to the estimation of the intrinsic dimensionality of a data set. The problem, which is not new, is difficult for several reasons:
  – If using the simplest technique of model selection, trial-and-error to find the best dimensionality, the computation time is high and the curse of the dimensionality limits the maximal dimensionality that can be implemented (depending on each method). But, more importantly, this technique is not well grounded—it amounts to inspecting the dependence on the number of dimensions used of the likelihood, reconstruction error, stress or some other fitness measure—and can result in unreliable estimates of the intrinsic dimensionality, more so if the model is not suitable for the data.
  – None of the more sophisticated techniques, such as cross-validation, minimum description length, minimum message length or Bayesian analysis, is generally successful. The Bayesian framework is the most principled one, but it is analytically intractable for any model of moderate complexity. Practical implementation of Bayesian techniques is a subject of active research in statistics and pattern recognition.
  – It is dependent on prior information that we often do not really know how to express, e.g. how much of the variation is noise and how much intrinsic? Do we need a two-dimensional linear model or a one-dimensional nonlinear one? This is further complicated by our lack of intuition of the geometry of high-dimensional Euclidean spaces.
- Development of continuous latent variable models (and generally density models and mapping approximators) whose computational complexity is affected as little as possible by the curse of the dimensionality while keeping a high degree of flexibility. As the previous suggestion, this is neither a new nor an easy

problem, but it is certainly of crucial practical relevance. On the other hand, as processing power increases, we may be able to work comfortably with intrinsic dimensions up to a value good enough for many applications. For example, 3 should be enough for a moderately complex robot arm, while speech may require more than 5.

- Our method for missing data reconstruction is very general and so could have many practical applications. We have described some in the thesis in some detail: decoding of neural population activity for hippocampal place cells; wind field retrieval from scatterometer data; inverse kinematics and dynamics of a redundant manipulator; the acoustic-to-articulatory mapping; audiovisual mappings for speech recognition; and recognition of occluded speech.

  We have also hinted at several extensions of the missing data reconstruction algorithm in chapter 7:

  - Multidimensional constraints, e.g. for data with continuity over a field rather than along a curve.
  - Use of different constraint types, e.g. smoothness, forward mapping or quadratic potential energy.
  - The use of bump finding instead of mode finding, which could be much faster and robust. One difficulty is that, unlike modes, bumps are not well defined.
  - The estimation of smooth density models whose conditional distributions do not contain spurious modes.
  - The treatment of unbounded horizon problems, where the sequence is infinite.
  - The treatment of discontinuities in the data, either intrinsic or due to undersampling.

Appendix A

Mathematical formulae

This appendix contains a collection of well-known mathematical results, stated without demonstration, that are used throughout the thesis. The reader is referred to standard texts on calculus, algebra, probability theory and information theory for proofs and background material.

A.1 Matrix identities

Definition A.1.1 (Penrose). The *pseudoinverse* matrix (or generalised inverse, or Moore-Penrose inverse) of a $p \times q$ matrix $\mathbf{A}$ is the $q \times p$ matrix $\mathbf{A}^+$ that verifies the following conditions:

$$\mathbf{A}\mathbf{A}^+ \text{ and } \mathbf{A}^+\mathbf{A} \text{ are symmetric} \tag{A.1a}$$

$$\mathbf{A}\mathbf{A}^+\mathbf{A} = \mathbf{A} \tag{A.1b}$$

$$\mathbf{A}^+\mathbf{A}\mathbf{A}^+ = \mathbf{A}^+ \tag{A.1c}$$

It can be proven that $\mathbf{A}^+$ exists and is unique for any matrix (Bjerhammar, 1973).

Theorem A.1.1 (Partitioned matrices). *If* $\mathbf{A} = \left(\begin{smallmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{A}_{21} & \mathbf{A}_{22} \end{smallmatrix}\right)$ *is a matrix partitioned in blocks, then:*

(i) $|\mathbf{A}| = |\mathbf{A}_{11}|\,|\mathbf{A}_{22} - \mathbf{A}_{21}\mathbf{A}_{11}^{-1}\mathbf{A}_{12}| = |\mathbf{A}_{22}|\,|\mathbf{A}_{11} - \mathbf{A}_{12}\mathbf{A}_{22}^{-1}\mathbf{A}_{21}|$.

(ii) If all the necessary inverses exist, the inverse $\mathbf{A} = \left(\begin{smallmatrix} \mathbf{A}^{11} & \mathbf{A}^{12} \\ \mathbf{A}^{21} & \mathbf{A}^{22} \end{smallmatrix}\right)$ *is given by*

$$\mathbf{A}^{11} = (\mathbf{A}_{11} - \mathbf{A}_{12}\mathbf{A}_{22}^{-1}\mathbf{A}_{21})^{-1} \qquad \mathbf{A}^{12} = -\mathbf{A}^{11}\mathbf{A}_{12}\mathbf{A}_{22}^{-1} = -\mathbf{A}_{11}^{-1}\mathbf{A}_{12}\mathbf{A}^{22}$$
$$\mathbf{A}^{22} = (\mathbf{A}_{22} - \mathbf{A}_{21}\mathbf{A}_{11}^{-1}\mathbf{A}_{12})^{-1} \qquad \mathbf{A}^{21} = -\mathbf{A}^{22}\mathbf{A}_{21}\mathbf{A}_{11}^{-1} = -\mathbf{A}_{22}^{-1}\mathbf{A}_{21}\mathbf{A}^{11}.$$

Theorem A.1.2 (Sums of products). *Given* $\mathbf{A}_{p\times p}$, $\mathbf{B}_{p\times q}$, $\mathbf{C}_{q\times q}$ *and* $\mathbf{D}_{q\times p}$, *if all the necessary inverses exist:*

$$(\mathbf{A} + \mathbf{BCD})^{-1} = \mathbf{A}^{-1} - \mathbf{A}^{-1}\mathbf{B}(\mathbf{C}^{-1} + \mathbf{D}\mathbf{A}^{-1}\mathbf{B})^{-1}\mathbf{D}\mathbf{A}^{-1} \tag{A.2}$$

$$|\mathbf{A} + \mathbf{BD}| = |\mathbf{A}|\,|\mathbf{I}_p + \mathbf{A}^{-1}\mathbf{BD}| = |\mathbf{A}|\,|\mathbf{I}_q + \mathbf{D}\mathbf{A}^{-1}\mathbf{B}|. \tag{A.3}$$

Eq. (A.2) *is the* Sherman-Morrison-Woodbury (SMW) formula.

For proofs see e.g. Golub and van Loan (1996) or simply confirm them by substitution.

Theorem A.1.3 (Matrix differentiation). *Let* $\mathbf{A} = (a_{ij})$ *be a square matrix and* $\mathbf{A}^{-1} \stackrel{\text{def}}{=} (a_{ij}^{-1})$ *its inverse:*

$$\frac{\partial}{\partial a_{ij}} \ln|\mathbf{A}| = a_{ji}^{-1} \qquad \frac{\partial}{\partial a_{ij}} a_{lm}^{-1} = -a_{li}^{-1}a_{jm}^{-1} \qquad \frac{\partial f}{\partial a_{ij}^{-1}} = -\sum_{l,m} a_{jm}\left(\frac{\partial f}{\partial a_{lm}}\right) a_{li}. \tag{A.4}$$

To prove the previous identities, use the cofactors. For example, $|\mathbf{A}| = \sum_j a_{ij} \operatorname{cof} a_{ij}$ for any row i. Then:

$$\frac{\partial}{\partial a_{ij}} \ln|\mathbf{A}| = \frac{\operatorname{cof} a_{ij}}{|\mathbf{A}|} \Rightarrow \frac{\partial}{\partial \mathbf{A}} \ln|\mathbf{A}| = \frac{(\operatorname{adj}\mathbf{A})^T}{|\mathbf{A}|} = (\mathbf{A}^T)^{-1}.$$

A.2 Moments and cumulants of a distribution

The moments and cumulants of the density $p(x)$ of a random variable x are defined as follows:

$$\text{Moment of order } m\text{: } \mu_m(p) = \mathrm{E}_{p(x)}\left\{x^m\right\} = \frac{d^m}{ds^m}\mathrm{E}_{p(x)}\left\{e^{sx}\right\}\Big|_{s=0}.$$
$$\text{Cumulant of order } m\text{: } k_m(p) = \frac{d^m}{i^m ds^m}\ln \mathrm{E}_{p(x)}\left\{e^{isx}\right\}\Big|_{s=0}, \text{ with } i = \sqrt{-1}.$$

The cumulant of order 4 is called the *kurtosis* of the distribution[1],

$$k_4 = \mu_4 - 4\mu_3\mu_1 - 3\mu_2^2 + 12\mu_1^2\mu_2 - 6\mu_1^4,$$

and indicates how peaked and heavy-tailed the distribution is. For symmetric zero-mean distributions μ_2 is the variance and $k_4 = \mu_4 - 3\mu_2^2$ (sometimes written as $k_4 = \frac{\mu_4}{\mu_2^2} - 3$). We define the following for symmetric and unimodal distributions:

- A *mesokurtic* distribution has zero kurtosis, like the normal.
- A *supergaussian* or *leptokurtic* distribution has positive kurtosis, that is, its peak is higher and its tails heavier than those of the normal distribution of the same variance.
- A *subgaussian* or *platykurtic* distribution has negative kurtosis, that is, its peak is lower and its tails lighter than those of the normal distribution of the same variance.

A.3 Properties of the normal distribution

Theorem A.3.1. *Let* $\mathbf{x} \sim \mathcal{N}_p(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. *Then:*

(i) $\mathbf{A}_{q\times p} \Rightarrow \mathbf{A}\mathbf{x} + \mathbf{b} \sim \mathcal{N}_q(\mathbf{A}\boldsymbol{\mu} + \mathbf{b}, \mathbf{A}\boldsymbol{\Sigma}\mathbf{A}^T)$.

(ii) $\boldsymbol{\Sigma}^{-1/2}(\mathbf{x} - \boldsymbol{\mu}) \sim \mathcal{N}_p(\mathbf{0}, \mathbf{I})$.

(iii) $\boldsymbol{\Sigma} = \sigma^2\mathbf{I}$ *and* $\mathbf{A}\mathbf{A}^T = \mathbf{I} \Rightarrow \mathbf{A}\mathbf{x} \sim \mathcal{N}_q(\mathbf{A}\boldsymbol{\mu}, \sigma^2\mathbf{I})$.

(iv) Partitioning $\mathbf{x} = \left(\begin{smallmatrix}\mathbf{x}_1\\ \mathbf{x}_2\end{smallmatrix}\right)$ *as* $p = r + s$ *and correspondingly* $\boldsymbol{\mu} = \left(\begin{smallmatrix}\boldsymbol{\mu}_1\\ \boldsymbol{\mu}_2\end{smallmatrix}\right)$ *and* $\boldsymbol{\Sigma} = \left(\begin{smallmatrix}\boldsymbol{\Sigma}_{11} & \boldsymbol{\Sigma}_{12}\\ \boldsymbol{\Sigma}_{21} & \boldsymbol{\Sigma}_{22}\end{smallmatrix}\right)$, *then:*

$$\mathbf{x}_1 \sim \mathcal{N}_r(\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_{11}) \qquad \mathbf{x}_2|\mathbf{x}_1 \sim \mathcal{N}_s\left(\boldsymbol{\mu}_2 + \boldsymbol{\Sigma}_{21}\,\boldsymbol{\Sigma}_{11}^{-1}(\mathbf{x}_1 - \boldsymbol{\mu}_1), \boldsymbol{\Sigma}_{22} - \boldsymbol{\Sigma}_{21}\,\boldsymbol{\Sigma}_{11}^{-1}\boldsymbol{\Sigma}_{12}\right).$$

For proofs see e.g. Mardia et al. (1979).

A.4 Information theory properties

For proofs and a thorough treatment of information theory see e.g. Cover and Thomas (1991).

Definition A.4.1. Given discrete random variables X, Y and Z, the following functionals, depending only on the distribution of the associated variable(s), are defined (in all cases the expectations are taken with respect to the joint distribution of all variables involved, unless indicated):

- Entropy $H(X) = \mathrm{E}\left\{\ln\frac{1}{p(X)}\right\}$: a measure of the uncertainty in a random variable, or a measure of the amount of information required on the average to describe it.
- Conditional entropy $H(X|Y) = \mathrm{E}\left\{\ln\frac{1}{p(X|Y)}\right\}$.
- Joint entropy $H(X,Y) = \mathrm{E}\left\{\ln\frac{1}{p(X,Y)}\right\}$.
- Kullback-Leibler distance, or directed divergence, or relative entropy $\mathrm{D}\left(p\|q\right) = \mathrm{E}_p\left\{\ln\frac{p(X)}{q(X)}\right\}$: a measure of the distance between two distributions p and q, or a measure of the inefficiency (in bits of a code if the logarithm is in base 2) of assuming that the distribution is q when it is p.

[1]Sometimes the *standardised kurtosis* is employed, obtained by dividing k_4 by the squared variance σ^4, and is often expressed as $\frac{\mathrm{E}_{p(s)}\left\{(s-\mu)^4\right\}}{\sigma^4} - 3$.

- Conditional relative entropy $\mathrm{D}\left(p(Y|X)\|q(Y|X)\right) = \mathrm{E}_p\left\{\ln\frac{p(Y|X)}{q(Y|X)}\right\}$.
- Mutual information $I(X;Y) = \mathrm{E}\left\{\ln\frac{p(X,Y)}{p(X)p(Y)}\right\}$: a measure of the amount of information that one random variable contains about another one, or of the reduction in the uncertainty of one random variable due to the knowledge of the other.
- Conditional mutual information $I(X;Y|Z) = \mathrm{E}\left\{\ln\frac{p(X,Y|Z)}{p(X|Z)p(Y|Z)}\right\}$.

Theorem A.4.1. *Properties of the entropy, Kullback-Leibler distance and mutual information:*

(i) $H(X) \geq 0$.

(ii) $H(Y|X) \neq H(X|Y)$ *in general.*

(iii) $H(X,Y) = H(X) + H(Y|X)$ *and* $H(X,Y|Z) = H(X|Z) + H(Y|X,Z)$.

(iv) $H(X|Y) \leq H(X)$ *with equality if and only if* X *and* Y *are independent: conditioning reduces entropy. Note that this is only true in the average, since for a particular value* y *of* Y *it can happen that* $H(X|Y=y)$ *be greater, equal or smaller than* $H(X)$.

(v) $H(X,Y) \leq H(X) + H(Y)$ *with equality if and only if* X *and* Y *are independent: independence bound on entropy.*

(vi) $I(X;Y) \geq 0$ *with equality if and only if* X *and* Y *are independent.*

(vii) $I(X;Y|Z) \geq 0$ *with equality if and only if* X *and* Y *are independent given* Z.

(viii) $I(X;Y) = I(Y;X)$*:* X *says as much about* Y *as* Y *says about* X.

(ix) $I(X;Y) = H(X) - H(X|Y)$*: reduction in the uncertainty of* X *due to the knowledge of* Y.

(x) $I(X;Y) = H(X) + H(Y) - H(X,Y)$.

(xi) $I(X;X) = H(X)$*: entropy = self-information.*

A.5 The Jacobian

For a function $\mathbf{f}: \mathcal{U} \to \mathbb{R}^D$ with $\mathcal{U}$ open in $\mathbb{R}^L$, the derivative $\mathbf{f}' = \frac{\partial \mathbf{f}}{\partial \mathbf{x}}$ is a linear mapping from $\mathbb{R}^L$ to $\mathbb{R}^D$. Its matrix of partial derivatives $\mathbf{J}(\mathbf{x}_0) \stackrel{\text{def}}{=} \left(\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right)\big|_{\mathbf{x}_0} = (j_{dl})$ with $j_{dl} = \frac{\partial f_d}{\partial x_l}\big|_{\mathbf{x}_0}$ is called *Jacobian matrix* or simply *Jacobian* and its determinant $J(\mathbf{x}_0)$ is called *Jacobian determinant* or, confusingly enough, *Jacobian* as well:

$$J: \begin{array}{ccc} \mathbb{R}^L & \longrightarrow & \mathbb{R} \\ \mathbf{x}_0 & \mapsto & J(\mathbf{x}_0) = |\mathbf{J}(\mathbf{x}_0)|. \end{array}$$

When we use the term *Jacobian* in this work, the context will make clear what meaning we refer to.

A point $\mathbf{x}$ is called *singular* if the Jacobian of $\mathbf{f}$ at $\mathbf{x}$ is singular (or equivalently if the derivative $\mathbf{f}'$ at $\mathbf{x}$ is not surjective). Otherwise it is called *regular*.

If we assume probability distributions on $\mathbb{R}^L$ and $\mathbb{R}^D$, then the transformation $\mathbf{x} \xrightarrow{\mathbf{f}} \mathbf{t}$ distorts the space $\mathbb{R}^L$, but the probability of a hyperrectangle is conserved: $p(\mathbf{x})d\mathbf{x} = p(\mathbf{t})d\mathbf{t}$. Thus $p(\mathbf{t}) = p(\mathbf{f}(\mathbf{x})) = p(\mathbf{x})\,|\mathbf{J}(\mathbf{x})|^{-1}$ where $\mathbf{J}(\mathbf{x})$ is the Jacobian of $\mathbf{f}$ at $\mathbf{x}$. This gives a recipe to compute the density induced by a transformation.

A.6 The inverse function theorem

Intuitively, the inverse function theorem allows to solve the equation $\mathbf{f}(\mathbf{x}) = \mathbf{t}$ locally and besides gives $\mathbf{x}$ as a smooth function of $\mathbf{t}$.

Theorem A.6.1 (Inverse function theorem). *Let* $\mathcal{U}$ *be an open subset of* $\mathbb{R}^L$ *and* $\mathbf{f}: \mathcal{U} \to \mathbb{R}^D$ *be a continuously differentiable function. Let* $\mathbf{x}_0 \in \mathcal{U}$ *be a point where the linear function* $\mathbf{f}'(\mathbf{x}_0): \mathbb{R}^L \to \mathbb{R}^D$ *is invertible (or, equivalently, the Jacobian of* $\mathbf{f}$ *is nonsingular). Then there exists an open neighbourhood* $\mathcal{W}$ *so that* $\mathcal{V} = \mathbf{f}(\mathcal{W})$ *is open in* $\mathbb{R}^D$ *and* $\mathbf{f}|_{\mathcal{W}}: \mathcal{W} \to \mathcal{V}$ *(the restriction of* $\mathbf{f}$ *to* $\mathcal{W}$*) is a diffeomorphism.*

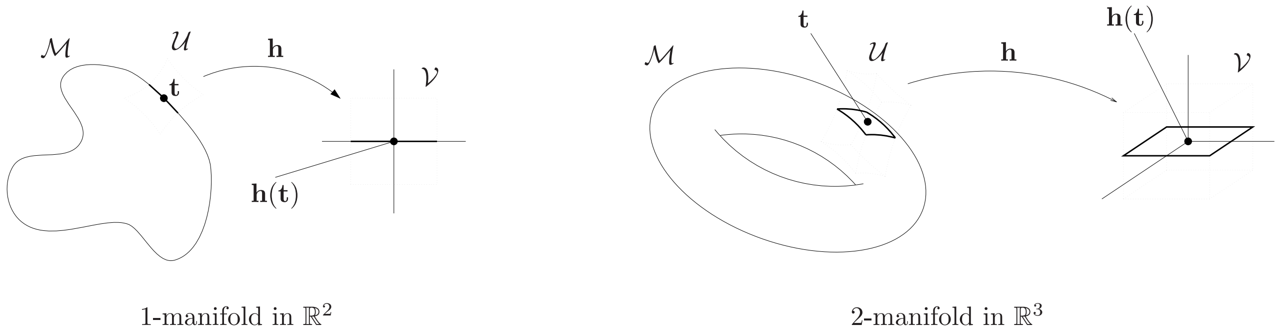


Figure A.1: Examples of manifolds.

An inverse function may also exist when the Jacobian of $\mathbf{f}$ at $\mathbf{x}_0$ is singular, e.g. for $f(x) \stackrel{\text{def}}{=} x^3$ at $x_0 = 0$.

An alternative formulation (derived from theorem A.7.1) of the inverse function theorem that remarks the dimensionality of the image space is the following:

Theorem A.6.2. *If $\mathbf{t} \in \mathbb{R}^D$ is a regular value of the continuously differentiable function $\mathbf{f} : \mathcal{U} \to \mathbb{R}^D$ with $\mathcal{U}$ open in $\mathbb{R}^L$, then $\mathbf{f}^{-1}(\mathbf{t})$ is empty or it is a submanifold of $\mathcal{U}$ of dimension $L - D$.*

Spivak (1965, pp. 34–43) gives proofs and more background about this central result and its relative, the implicit function theorem.

A.7 Manifolds in $\mathbb{R}^D$

This section (which is mainly based on the book by Spivak, 1965) briefly formalises the concept of L-dimensional manifold in $\mathbb{R}^D$. The main idea to keep in mind is that, while a manifold of $\mathbb{R}^D$ itself is just a subset of $\mathbb{R}^D$, it has a dimension from a geometrical point of view. Indeed, an L-dimensional manifold in $\mathbb{R}^D$ is a subset of $\mathbb{R}^D$ that has only L degrees of freedom, i.e., that can be described with only L coordinates. For example, if we restrict ourselves to the case of vector subspaces, a vector subspace of dimension L can be described by a system of L linearly independent vectors (a basis); the projection of a vector $\mathbf{t}$ of the subspace onto that basis gives L real numbers, which are the coordinates of $\mathbf{t}$ in the coordinate system of that basis. Needless to say, the election of the coordinate system is not unique.

Let us now define more formally the previous ideas. First, we introduce the following naming convention:

- We will call L-manifold an L-dimensional manifold in $\mathbb{R}^D$.
- We will consider that a mapping is *differentiable* if and only if it is continuous and has continuous derivatives of all orders.
- A *diffeomorphism* $\mathbf{h}$ is a differentiable mapping $\mathbf{h} : \mathcal{U} \longrightarrow \mathcal{V}$ between two open sets $\mathcal{U}, \mathcal{V} \subset \mathbb{R}^D$ with differentiable inverse $\mathbf{h}^{-1}$.

Definition A.7.1. $\mathcal{M} \subset \mathbb{R}^D$ is an L-manifold if and only if for all $\mathbf{t} \in \mathcal{M}$ the following condition holds:

(M) There exist two open sets $\mathcal{U}, \mathcal{V} \subset \mathbb{R}^D$ with $\mathbf{t} \in \mathcal{U}$ and a diffeomorphism $\mathbf{h} : \mathcal{U} \longrightarrow \mathcal{V}$ such that:

$$\mathbf{h}(\mathcal{U} \cap \mathcal{M}) = \mathcal{V} \cap (\mathbb{R}^L \times \{\mathbf{0}\}) = \{\mathbf{y} \in \mathcal{V} : \; y_{L+1} = \cdots = y_D = 0\}.$$

For example, in $\mathbb{R}^D$:

- A point is a 0-manifold.
- An L-dimensional vector subspace is an L-manifold.
- The hollow D-sphere is an $(D-1)$-manifold.
- Any open subset is an D-manifold.

Figure A.1 shows two examples of manifolds.

Most manifolds can be expressed by a functional formula, like the hollow unit D-sphere: $\sum_{d=1}^{D} t_d^2 = 1$. The following theorem helps to find the dimension in those cases.

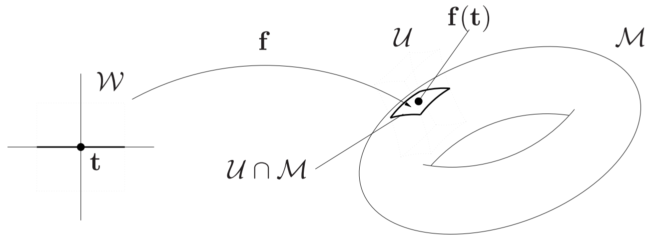


Figure A.2: Coordinate system of a 2-manifold in $\mathbb{R}^3$.

Theorem A.7.1. *Let $\mathcal{A}$ be an open subset in $\mathbb{R}^D$ and $\mathbf{g} : \mathcal{A} \longrightarrow \mathbb{R}^L$ differentiable. If the Jacobian $\mathbf{g}'(\mathbf{t})$ of $\mathbf{g}$ has rank L for $\mathbf{g}(\mathbf{t}) = \mathbf{0}$, then $\mathbf{g}^{-1}(\mathbf{0})$ is an $(D - L)$-manifold of $\mathbb{R}^D$.*

For example, $S_1^D = \mathbf{g}^{-1}(\mathbf{0})$ for $\mathbf{g} : \mathbb{R}^D \longrightarrow \mathbb{R}$ defined as $\mathbf{g}(\mathbf{t}) = \|\mathbf{t}\|^2 - 1 = \sum_{d=1}^D t_d^2 - 1$.
The following theorem introduces the concept of coordinate system of a manifold.

Theorem A.7.2. *$\mathcal{M} \subset \mathbb{R}^D$ is an L-manifold of $\mathbb{R}^D$ if and only if for all $\mathbf{t} \in \mathcal{M}$ the following condition holds:*

(C) There exist two open sets $\mathcal{U} \subset \mathbb{R}^D$, $\mathcal{W} \subset \mathbb{R}^L$ with $\mathbf{t} \in \mathcal{U}$, and a differentiable one-to-one mapping $\mathbf{f} : \mathcal{W} \longrightarrow \mathbb{R}^D$ such that:

1. *$\mathbf{f}(\mathcal{W}) = \mathcal{M} \cap \mathcal{U}$.*
2. *The Jacobian $\mathbf{f}'(\mathbf{x})$ has rank L for all $\mathbf{x} \in \mathcal{W}$.*
3. *$\mathbf{f}$ has a continuous inverse $\mathbf{f}^{-1} : \mathbf{f}(\mathcal{W}) \longrightarrow \mathcal{W}$.*

$\mathbf{f}$ is called a **coordinate system** around $\mathbf{t}$. $\mathbf{f}$ and $\mathcal{U} \cap \mathcal{M}$ define the **coordinate neighbourhood** of $\mathcal{M}$. Figure A.2 illustrates the point.

Finally we introduce the **manifolds-with-boundaries**.

Definition A.7.2. $\mathcal{M} \subset \mathbb{R}^D$ is an L-manifold-with-boundary if and only if for all $\mathbf{t} \in \mathcal{M}$ either condition (M) or the following condition hold:

(M') There exist $\mathcal{U}$, $\mathcal{V}$ open subsets of $\mathbb{R}^D$ with $\mathbf{t} \in \mathcal{U}$ and a diffeomorphism $\mathbf{h} : \mathcal{U} \longrightarrow \mathcal{V}$ such that:

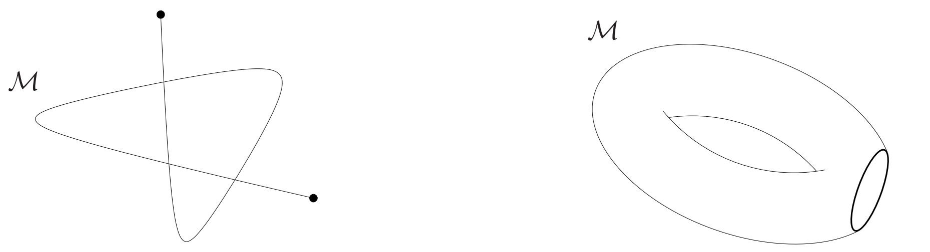
$$\mathbf{h}(\mathcal{U} \cap \mathcal{M}) = \mathcal{V} \cap (\mathbb{H}^L \times \{\mathbf{0}\}) = \{\mathbf{y} \in \mathcal{V} : \; y_L \geq 0 \text{ and } y_{L+1} = \cdots = y_D = 0\}$$

and $\mathbf{h}(\mathbf{t})$ has its Lth component equal to 0. $\mathbb{H}^L \stackrel{\text{def}}{=} \{\mathbf{t} \in \mathbb{R}^L : \; t_L \geq 0\}$ is the half-space of $\mathbb{R}^L$.

This definition separates a manifold $\mathcal{M}$ into two disjoint sets:

- The boundary of $\mathcal{M}$, $\partial\mathcal{M} \stackrel{\text{def}}{=} \{\mathbf{t} \in \mathcal{M} : \text{ (M') holds}\}$, which is a $(L-1)$-manifold.
- The rest, $\mathcal{M} \setminus \partial\mathcal{M} = \{\mathbf{t} \in \mathcal{M} : \text{ (M) holds}\}$, which is an L-manifold.

Figure A.3 gives two examples of manifolds-with-boundaries.



1-manifold-with-boundary in $\mathbb{R}^2$ 2-manifold-with-boundary in $\mathbb{R}^3$

Figure A.3: Examples of manifolds-with-boundary. The highlighted points belong to the boundary.

Appendix B

The ACCOR database

The EUR-ACCOR database (ESPRIT II Basic Research Actions 3279 and 7098; Marchal and Hardcastle, 1993) was designed for the cross-language study of coarticulation. It contains different kinds of measurements (electropalatography, pneumotachography, laryngography and some electromagnetic articulography, as well as the original acoustic signal) for utterances, isolated words and nonsense items spoken by several subjects in seven different European languages (French, English, German, Italian, Catalan, Swedish and Irish Gaelic) and varying speech styles (slow, fast, etc.).

The EPG data consists of 62-bit frames sampled at 200 Hz. Each subject was encouraged to wear the artificial palate for a least a 3 hour period prior to recording to become accustomed to the feel of the device in the mouth. The acoustic waveform was sampled at 20 kHz with 16 bits per sample.

Table B.1 shows the sentences used in our experiments and table B.2 the 6 different subjects used from the English EPG database (all English native speakers but with different accents).

I am grateful to Dr. Alan Wrench for providing me with the ACCOR data and for clarifying some technical questions about it.

Number	Sentence and phonemic transcription if available	Duration (s)
1	The hostess should always wear clean gloves /ðə ’həʊstɪs ʃəd ’ɔlwɪz ’weə ’klin ’glʌvz/	4.05
2	When the song’s the thing, the cast can’t sing	3.90
3	All the keys have been handed out, Bill /’ɔl ðə ’kiz həv bin ’hændɪd ’aʊt ’bɪl/	3.41
4	I prefer Kant to Hobbes for a good bedtime book /aɪ prɪ’fɜ ’kænt tə ’hɒbz fər ə ’gʊd ’bedtaɪm ’bʊk/	4.00
5	Fred can go, Susan can’t go, and Linda is uncertain /’frɛd kən gəʊ // ’suzən ’kɑnt gəʊ // ənd ’lɪndə ɪz ən’sɜtən/	4.83
6	We tore down the outbuildings	2.92
7	Bella ran past the school, then came towards us	4.09
8	The cold front is expected to pass this way on Sunday	4.21
9	The catalogue lists just his own brand of tyre /ðə ’kætəlɒg ’lɪsts ’ʤʌst hɪz ’əʊn ’brænd əv ’taɪə/	4.16
10	Put your hat on the hatrack and your coat in the cupboard	4.00
11	She climbed up to see behind the clock	3.25
12	It’s a good thing he’s not just fooling around	3.50
13	Stats by such a student will surely be trashed after tests	5.16
14	The lads showed us a strategy for trimming his moustache surreptitiously	5.23

Table B.1: ACCOR database sentences used in the experiments. Nicolaidis and Hardcastle (1994) give detailed annotations of some of the utterances (I am grateful to Prof. Hardcastle for sending me a copy of this report).

Speaker	Sex	Accent
FG	female	Southern English
HD	female	Northern English
KM	female	Southern English
PD	male	Southern English
RK	male	Southern English
SN	female	Southern English

Table B.2: Speakers recorded for the English EPG ACCOR database.

Appendix C

WWW files

The following online material is available from my web page at `http://www.dcs.shef.ac.uk/~miguel`:

- This thesis and some publications resulting from it.
- A BibTeX file containing the bibliography for the thesis. Most of the entries contain also the abstract, keywords, URL (if the reference is available online), related software and other additional information.
- Associated Matlab software:
  - Factor analysis and varimax rotation (section 2.6.1).
  - Principal component analysis (section 2.6.2).
  - Mixture of multivariate Bernoulli distributions (chapter 3).
  - Tools for imaging EPG frames (chapter 5).
  - Mode-finding algorithm and error bars computation (chapter 8).

  Further Matlab software of more difficult encapsulation, such as the extensions to GTM for sequence reconstruction and the continuity constraint implementation with dynamic programming, is available directly from the author (email `miguel@dcs.shef.ac.uk`).

Bibliography

S. Amari. Natural gradient learning for over- and under-complete bases in ICA. *Neural Computation*, 11(8): 1875–1883, Nov. 1999.

S. Amari and A. Cichoki. Adaptive blind signal processing—neural network approaches. *Proc. IEEE*, 86(10): 2026–2048, Oct. 1998.

T. W. Anderson. Asymptotic theory for principal component analysis. *Annals of Mathematical Statistics*, 34 (1):122–148, Mar. 1963.

T. W. Anderson and H. Rubin. Statistical inference in factor analysis. In J. Neyman, editor, *Proc. 3rd Berkeley Symp. Mathematical Statistics and Probability*, volume V, pages 111–150, Berkeley, 1956. University of California Press.

S. Arnfield. Artificial EPG palate image. The Reading EPG, 1995. Available online at `http://www.linguistics.reading.ac.uk/research/speechlab/epg/palate.jpg`, Feb. 1, 2000.

H. Asada and J.-J. E. Slotine. *Robot Analysis and Control.* John Wiley & Sons, New York, London, Sydney, 1986.

D. Asimov. The grand tour: A tool for viewing multidimensional data. *SIAM J. Sci. Stat. Comput.*, 6:128–143, 1985.

B. S. Atal, J. J. Chang, M. V. Mathews, and J. W. Tukey. Inversion of articulatory-to-acoustic transformation in the vocal tract by a computer-sorting technique. *J. Acoustic Soc. Amer.*, 63(5):1535–1555, May 1978.

C. G. Atkeson. Learning arm kinematics and dynamics. *Annu. Rev. Neurosci.*, 12:157–183, 1989.

H. Attias. EM algorithms for independent component analysis. In Niranjan (1998), pages 132–141.

H. Attias. Independent factor analysis. *Neural Computation*, 11(4):803–851, May 1999.

F. Aurenhammer. Voronoi diagrams—a survey of a fundamental geometric data structure. *ACM Computing Surveys*, 23(3):345–405, Sept. 1991.

A. Azzalini and A. Capitanio. Statistical applications of the multivariate skew-normal distribution. *Journal of the Royal Statistical Society, B*, 61(3):579–602, 1999.

R. J. Baddeley. Searching for filters with "interesting" output distributions: An uninteresting direction to explore? *Network: Computation in Neural Systems*, 7(2):409–421, 1996.

R. Bakis. Coarticulation modeling with continuous-state HMMs. In *Proc. IEEE Workshop Automatic Speech Recognition*, pages 20–21, Arden House, New York, 1991. Harriman.

R. Bakis. An articulatory-like speech production model with controlled use of prior knowledge. Frontiers in Speech Processing: Robust Speech Analysis '93, Workshop CDROM, NIST Speech Disc 15 (also available from the Linguistic Data Consortium), Aug. 6 1993.

P. Baldi and K. Hornik. Neural networks and principal component analysis: Learning from examples without local minima. *Neural Networks*, 2(1):53–58, 1989.

J. D. Banfield and A. E. Raftery. Ice floe identification in satellite images using mathematical morphology and clustering about principal curves. *J. Amer. Stat. Assoc.*, 87(417):7–16, Mar. 1992.

J. Barker, L. Josifovski, M. Cooke, and P. Green. Soft decisions in missing data techniques for robust automatic speech recognition. In *Proc. of the International Conference on Spoken Language Processing (ICSLP'00)*, Beijing, China, Oct. 16–20 2000.

J. P. Barker and F. Berthommier. Evidence of correlation between acoustic and visual features of speech. In Ohala et al. (1999), pages 199–202.

M. F. Barnsley. *Fractals Everywhere*. Academic Press, New York, 1988.

D. J. Bartholomew. The foundations of factor analysis. *Biometrika*, 71(2):221–232, Aug. 1984.

D. J. Bartholomew. Foundations of factor analysis: Some practical implications. *Brit. J. of Mathematical and Statistical Psychology*, 38:1–10 (discussion in pp. 127–140), 1985.

D. J. Bartholomew. *Latent Variable Models and Factor Analysis*. Charles Griffin & Company Ltd., London, 1987.

A. Basilevsky. *Statistical Factor Analysis and Related Methods*. Wiley Series in Probability and Mathematical Statistics. John Wiley & Sons, New York, London, Sydney, 1994.

H.-U. Bauer, M. Herrmann, and T. Villmann. Neural maps and topographic vector quantization. *Neural Networks*, 12(4–5):659–676, June 1999.

H.-U. Bauer and K. R. Pawelzik. Quantifying the neighbourhood preservation of self-organizing feature maps. *IEEE Trans. Neural Networks*, 3(4):570–579, July 1992.

J. Behboodian. On the modes of a mixture of two normal distributions. *Technometrics*, 12(1):131–139, Feb. 1970.

A. J. Bell and T. J. Sejnowski. An information-maximization approach to blind separation and blind deconvolution. *Neural Computation*, 7(6):1129–1159, 1995.

A. J. Bell and T. J. Sejnowski. The "independent components" of natural scenes are edge filters. *Vision Res.*, 37(23):3327–3338, Dec. 1997.

R. Bellman. *Dynamic Programming*. Princeton University Press, Princeton, 1957.

R. Bellman. *Adaptive Control Processes: A Guided Tour*. Princeton University Press, Princeton, 1961.

Y. Bengio and F. Gingras. Recurrent neural networks for missing or asynchronous data. In D. S. Touretzky, M. C. Mozer, and M. E. Hasselmo, editors, *Advances in Neural Information Processing Systems*, volume 8, pages 395–401. MIT Press, Cambridge, MA, 1996.

C. Benoît, M.-T. Lallouache, T. Mohamadi, and C. Abry. A set of French visemes for visual speech synthesis. In G. Bailly and C. Benoît, editors, *Talking Machines: Theories, Models and Designs*, pages 485–504. North Holland-Elsevier Science Publishers, Amsterdam, New York, Oxford, 1992.

P. M. Bentler and J. S. Tanaka. Problems with EM algorithms for ML factor analysis. *Psychometrika*, 48(2): 247–251, June 1983.

J. O. Berger. *Statistical Decision Theory and Bayesian Analysis*. Springer Series in Statistics. Springer-Verlag, Berlin, second edition, 1985.

M. Berkane, editor. *Latent Variable Modeling and Applications to Causality*. Number 120 in Springer Series in Statistics. Springer-Verlag, Berlin, 1997.

J. M. Bernardo and A. F. M. Smith. *Bayesian Theory*. Wiley Series in Probability and Mathematical Statistics. John Wiley & Sons, Chichester, 1994.

N. Bernstein. *The Coordination and Regulation of Movements*. Pergamon, Oxford, 1967.

D. P. Bertsekas. *Dynamic Programming. Deterministic and Stochastic Models*. Prentice-Hall, Englewood Cliffs, N.J., 1987.

J. Besag and P. J. Green. Spatial statistics and Bayesian computation. *Journal of the Royal Statistical Society, B*, 55(1):25–37, 1993.

J. C. Bezdek and N. R. Pal. An index of topological preservation for feature extraction. *Pattern Recognition*, 28(3):381–391, Mar. 1995.

E. L. Bienenstock, L. N. Cooper, and P. W. Munro. Theory for the development of neuron selectivity: Orientation specificity and binocular interaction in visual cortex. *J. Neurosci.*, 2(1):32–48, Jan. 1982.

C. M. Bishop. Mixture density networks. Technical Report NCRG/94/004, Neural Computing Research Group, Aston University, Feb. 1994. Available online at `http://www.ncrg.aston.ac.uk/Papers/postscript/NCRG_94_004.ps.Z`.

C. M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, New York, Oxford, 1995.

C. M. Bishop. Bayesian PCA. In Kearns et al. (1999), pages 382–388.

C. M. Bishop, G. E. Hinton, and I. G. D. Strachan. GTM through time. In *IEE Fifth International Conference on Artificial Neural Networks*, pages 111–116, 1997a.

C. M. Bishop and I. T. Nabney. Modeling conditional probability distributions for periodic variables. *Neural Computation*, 8(5):1123–1133, July 1996.

C. M. Bishop, M. Svensén, and C. K. I. Williams. Magnification factors for the SOM and GTM algorithms. In *WSOM'97: Workshop on Self-Organizing Maps*, pages 333–338, Finland, June 4–6 1997b. Helsinki University of Technology.

C. M. Bishop, M. Svensén, and C. K. I. Williams. Developments of the generative topographic mapping. *Neurocomputing*, 21(1–3):203–224, Nov. 1998a.

C. M. Bishop, M. Svensén, and C. K. I. Williams. GTM: The generative topographic mapping. *Neural Computation*, 10(1):215–234, Jan. 1998b.

C. M. Bishop and M. E. Tipping. A hierarchical latent variable model for data visualization. *IEEE Trans. on Pattern Anal. and Machine Intel.*, 20(3):281–293, Mar. 1998.

A. Bjerhammar. *Theory of Errors and Generalized Matrix Inverses*. North Holland-Elsevier Science Publishers, Amsterdam, New York, Oxford, 1973.

C. S. Blackburn and S. Young. A self-learning predictive model of articulator movements during speech production. *J. Acoustic Soc. Amer.*, 107(3):1659–1670, Mar. 2000.

T. L. Boullion and P. L. Odell. *Generalized Inverse Matrices*. John Wiley & Sons, New York, London, Sydney, 1971.

H. Bourlard and Y. Kamp. Autoassociation by the multilayer perceptrons and singular value decomposition. *Biol. Cybern.*, 59(4–5):291–294, 1988.

H. Bourlard and N. Morgan. *Connectionist Speech Recognition. A Hybrid Approach*. Kluwer Academic Publishers Group, Dordrecht, The Netherlands, 1994.

M. Brand. Structure learning in conditional probability models via an entropic prior and parameter extinction. *Neural Computation*, 11(5):1155–1182, July 1999.

C. Bregler and S. M. Omohundro. Surface learning with applications to lip-reading. In Cowan et al. (1994), pages 43–50.

C. Bregler and S. M. Omohundro. Nonlinear image interpolation using manifold learning. In Tesauro et al. (1995), pages 973–980.

L. J. Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, Aug. 1996.

S. P. Brooks. Markov chain Monte Carlo method and its application. *The Statistician*, 47(1):69–100, 1998.

C. P. Browman and L. M. Goldstein. Articulatory phonology: An overview. *Phonetica*, 49(3–4):155–180, 1992.

E. N. Brown, L. M. Frank, D. Tang, M. C. Quirk, and M. A. Wilson. A statistical paradigm for neural spike train decoding applied to position prediction from ensemble firing patterns of rat hippocampal place cells. *J. Neurosci.*, 18(18):7411–7425, Sept. 1998.

G. J. Brown and M. Cooke. Computational auditory scene analysis. *Computer Speech and Language*, 8(4): 297–336, Oct. 1994.

D. Byrd, E. Flemming, C. A. Mueller, and C. C. Tan. Using regions and indices in EPG data reduction. *Journal of Speech and Hearing Research*, 38(4):821–827, Aug. 1995.

J.-F. Cardoso. Infomax and maximum likelihood for blind source separation. *IEEE Letters on Signal Processing*, 4(4):112–114, Apr. 1997.

J.-F. Cardoso. Blind signal separation: Statistical principles. *Proc. IEEE*, 86(10):2009–2025, Oct. 1998.

M. Á. Carreira-Perpiñán. A review of dimension reduction techniques. Technical Report CS–96–09, Dept. of Computer Science, University of Sheffield, UK, Dec. 1996. Available online at `http://www.dcs.shef.ac.uk/~miguel/papers/cs-96-09.html`.

M. Á. Carreira-Perpiñán. Density networks for dimension reduction of continuous data: Analytical solutions. Technical Report CS–97–09, Dept. of Computer Science, University of Sheffield, UK, Apr. 1997. Available online at `http://www.dcs.shef.ac.uk/~miguel/papers/cs-97-09.html`.

M. Á. Carreira-Perpiñán. Mode-finding for mixtures of Gaussian distributions. Technical Report CS–99–03, Dept. of Computer Science, University of Sheffield, UK, Mar. 1999a. Revised August 4, 2000. Available online at `http://www.dcs.shef.ac.uk/~miguel/papers/cs-99-03.html`.

M. Á. Carreira-Perpiñán. One-to-many mappings, continuity constraints and latent variable models. In *Proc. of the IEE Colloquium on Applied Statistical Pattern Recognition*, Birmingham, UK, 1999b.

M. Á. Carreira-Perpiñán. Mode-finding for mixtures of Gaussian distributions. *IEEE Trans. on Pattern Anal. and Machine Intel.*, 22(11):1318–1323, Nov. 2000a.

M. Á. Carreira-Perpiñán. Reconstruction of sequential data with probabilistic models and continuity constraints. In Solla et al. (2000), pages 414–420.

M. Á. Carreira-Perpiñán and S. Renals. Dimensionality reduction of electropalatographic data using latent variable models. *Speech Communication*, 26(4):259–282, Dec. 1998a.

M. Á. Carreira-Perpiñán and S. Renals. Experimental evaluation of latent variable models for dimensionality reduction. In Niranjan (1998), pages 165–173.

M. Á. Carreira-Perpiñán and S. Renals. A latent variable modelling approach to the acoustic-to-articulatory mapping problem. In Ohala et al. (1999), pages 2013–2016.

M. Á. Carreira-Perpiñán and S. Renals. Practical identifiability of finite mixtures of multivariate Bernoulli distributions. *Neural Computation*, 12(1):141–152, Jan. 2000.

J. Casti. Flight over Wall St. *New Scientist*, 154(2078):38–41, Apr. 19 1997.

T. Chen and R. R. Rao. Audio-visual integration in multimodal communication. *Proc. IEEE*, 86(5):837–852, May 1998.

H. Chernoff. The use of faces to represent points in k-dimensional space graphically. *J. Amer. Stat. Assoc.*, 68(342):361–368, June 1973.

D. A. Cohn. Neural network exploration using optimal experiment design. *Neural Networks*, 9(6):1071–1083, Aug. 1996.

C. H. Coker. A model of articulatory dynamics and control. *Proc. IEEE*, 64(4):452–460, 1976.

P. Comon. Independent component analysis: A new concept? *Signal Processing*, 36(3):287–314, Apr. 1994.

S. C. Constable, R. L. Parker, and C. G. Constable. Occam's inversion—a practical algorithm for generating smooth models from electromagnetic sounding data. *Geophysics*, 52(3):289–300, 1987.

D. Cook, A. Buja, and J. Cabrera. Projection pursuit indexes based on orthonormal function expansions. *Journal of Computational and Graphical Statistics*, 2(3):225–250, 1993.

M. Cooke and D. P. W. Ellis. The auditory organization of speech and other sources in listeners and computational models. *Speech Communication*, 2000. To appear.

M. Cooke, P. Green, L. Josifovski, and A. Vizinho. Robust automatic speech recognition with missing and unreliable acoustic data. *Speech Communication*, 34(3):267–285, June 2001.

D. Cornford, I. T. Nabney, and D. J. Evans. Bayesian retrieval of scatterometer wind fields. Technical Report NCRG/99/015, Neural Computing Research Group, Aston University, 1999a. Submitted to J. of Geophysical Research. Available online at `ftp://cs.aston.ac.uk/cornford/bayesret.ps.gz`.

D. Cornford, I. T. Nabney, and C. K. I. Williams. Modelling frontal discontinuities in wind fields. Technical Report NCRG/99/001, Neural Computing Research Group, Aston University, Jan. 1999b. Submitted to Nonparametric Statistics. Available online at `http://www.ncrg.aston.ac.uk/Papers/postscript/NCRG_99_001.ps.Z`.

R. Courant and D. Hilbert. *Methods of Mathematical Physics*, volume 1. Interscience Publishers, New York, 1953.

T. M. Cover and J. A. Thomas. *Elements of Information Theory*. Wiley Series in Telecommunications. John Wiley & Sons, New York, London, Sydney, 1991.

J. D. Cowan, G. Tesauro, and J. Alspector, editors. *Advances in Neural Information Processing Systems*, volume 6, 1994. Morgan Kaufmann, San Mateo.

T. F. Cox and M. A. A. Cox. *Multidimensional Scaling*. Chapman & Hall, London, New York, 1994.

J. J. Craig. *Introduction to Robotics. Mechanics and Control*. Series in Electrical and Computer Engineering: Control Engineering. Addison-Wesley, Reading, MA, USA, second edition, 1989.

N. Cristianini and J. Shawe-Taylor. *An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods*. Cambridge University Press, 2000.

P. Dayan. Arbitrary elastic topologies and ocular dominance. *Neural Computation*, 5(3):392–401, 1993.

P. Dayan, G. E. Hinton, R. M. Neal, and R. S. Zemel. The Helmholtz machine. *Neural Computation*, 7(5): 889–904, Sept. 1995.

M. H. DeGroot. *Probability and Statistics*. Addison-Wesley, Reading, MA, USA, 1986.

D. DeMers and G. W. Cottrell. Non-linear dimensionality reduction. In S. J. Hanson, J. D. Cowan, and C. L. Giles, editors, *Advances in Neural Information Processing Systems*, volume 5, pages 580–587. Morgan Kaufmann, San Mateo, 1993.

D. DeMers and K. Kreutz-Delgado. Learning global direct inverse kinematics. In J. Moody, S. J. Hanson, and R. P. Lippmann, editors, *Advances in Neural Information Processing Systems*, volume 4, pages 589–595. Morgan Kaufmann, San Mateo, 1992.

D. DeMers and K. Kreutz-Delgado. Canonical parameterization of excess motor degrees of freedom with self-organizing maps. *IEEE Trans. Neural Networks*, 7(1):43–55, Jan. 1996.

D. DeMers and K. Kreutz-Delgado. Learning global properties of nonredundant kinematic mappings. *Int. J. of Robotics Research*, 17(5):547–560, May 1998.

A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the *EM* algorithm. *Journal of the Royal Statistical Society, B*, 39(1):1–38, 1977.

L. Deng. A dynamic, feature-based approach to the interface between phonology and phonetics for speech modeling and recognition. *Speech Communication*, 24(4):299–323, July 1998.

L. Deng, G. Ramsay, and D. Sun. Production models as a structural basis for automatic speech recognition. *Speech Communication*, 22(2–3):93–111, Aug. 1997.

P. Diaconis and D. Freedman. Asymptotics of graphical projection pursuit. *Annals of Statistics*, 12(3):793–815, Sept. 1984.

K. I. Diamantaras and S.-Y. Kung. *Principal Component Neural Networks. Theory and Applications*. Wiley Series on Adaptive and Learning Systems for Signal Processing, Communications, and Control. John Wiley & Sons, New York, London, Sydney, 1996.

T. G. Dietterich. Machine learning research: Four current directions. *AI Magazine*, 18(4):97–136, winter 1997.

M. P. do Carmo. *Differential Geometry of Curves and Surfaces*. Prentice-Hall, Englewood Cliffs, N.J., 1976.

R. D. Dony and S. Haykin. Optimally adaptive transform coding. *IEEE Trans. on Image Processing*, 4(10): 1358–1370, Oct. 1995.

R. O. Duda and P. E. Hart. *Pattern Classification and Scene Analysis*. John Wiley & Sons, New York, London, Sydney, 1973.

R. Durbin, S. R. Eddy, A. Krogh, and G. Mitchison, editors. *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge University Press, 1998.

R. Durbin and G. Mitchison. A dimension reduction framework for understanding cortical maps. *Nature*, 343 (6259):644–647, Feb. 15 1990.

R. Durbin, R. Szeliski, and A. Yuille. An analysis of the elastic net approach to the traveling salesman problem. *Neural Computation*, 1(3):348–358, Fall 1989.

R. Durbin and D. Willshaw. An analogue approach to the traveling salesman problem using an elastic net method. *Nature*, 326(6114):689–691, Apr. 16 1987.

H. W. Engl, M. Hanke, and A. Neubauer. *Regularization of Inverse Problems*. Kluwer Academic Publishers Group, Dordrecht, The Netherlands, 1996.

K. Erler and G. H. Freeman. An HMM-based speech recognizer using overlapping articulatory features. *J. Acoustic Soc. Amer.*, 100(4):2500–2513, Oct. 1996.

G. Eslava and F. H. C. Marriott. Some criteria for projection pursuit. *Statistics and Computing*, 4:13–20, 1994.

C. Y. Espy-Wilson, S. E. Boyce, M. Jackson, S. Narayanan, and A. Alwan. Acoustic modeling of American English /r/. *J. Acoustic Soc. Amer.*, 108(1):343–356, July 2000.

J. Etezadi-Amoli and R. P. McDonald. A second generation nonlinear factor analysis. *Psychometrika*, 48(3): 315–342, Sept. 1983.

D. J. Evans, D. Cornford, and I. T. Nabney. Structured neural network modelling of multi-valued functions for wind vector retrieval from satellite scatterometer measurements. *Neurocomputing*, 30(1–4):23–30, Jan. 2000.

B. S. Everitt. *An Introduction to Latent Variable Models*. Monographs on Statistics and Applied Probability. Chapman & Hall, London, New York, 1984.

B. S. Everitt and D. J. Hand. *Finite Mixture Distributions*. Monographs on Statistics and Applied Probability. Chapman & Hall, London, New York, 1981.

K. J. Falconer. *Fractal Geometry: Mathematical Foundations and Applications*. John Wiley & Sons, Chichester, 1990.

K. Fan. On a theorem of Weyl concerning the eigenvalues of linear transformations II. *Proc. Natl. Acad. Sci. USA*, 36:31–35, 1950.

G. Fant. *Acoustic Theory of Speech Production*. Mouton, The Hague, Paris, second edition, 1970.

E. Farnetani, W. J. Hardcastle, and A. Marchal. Cross-language investigation of lingual coarticulatory processes using EPG. In J.-P. Tubach and J.-J. Mariani, editors, *Proc. EUROSPEECH'89*, volume 2, pages 429–432, Paris, France, Sept. 26–28 1989.

W. Feller. *An Introduction to Probability Theory and Its Applications*, volume 2 of *Wiley Series in Probability and Mathematical Statistics*. John Wiley & Sons, New York, London, Sydney, third edition, 1971.

D. J. Field. What is the goal of sensory coding? *Neural Computation*, 6(4):559–601, July 1994.

J. L. Flanagan. *Speech Analysis, Synthesis and Perception.* Number 3 in Kommunication und Kybernetik in Einzeldarstellungen. Springer-Verlag, Berlin, second edition, 1972.

M. K. Fleming and G. W. Cottrell. Categorization of faces using unsupervised feature extraction. In *Proc. Int. J. Conf. on Neural Networks (IJCNN90)*, volume II, pages 65–70, San Diego, CA, June 17–21 1990.

P. Földiák. Adaptive network for optimal linear feature extraction. In *Proc. Int. J. Conf. on Neural Networks (IJCNN89)*, volume I, pages 401–405, Washington, DC, June 18–22 1989.

D. Fotheringhame and R. Baddeley. Nonlinear principal components analysis of neuronal data. *Biol. Cybern.*, 77(4):283–288, 1997.

I. E. Frank and J. H. Friedman. A statistical view of some chemometrics regression tools. *Technometrics*, 35 (2):109–135 (with comments: pp. 136–148), May 1993.

J. Frankel, K. Richmond, S. King, and P. Taylor. An automatic speech recognition system using neural networks and linear dynamic models to recover and model articulatory traces. In *Proc. of the International Conference on Spoken Language Processing (ICSLP'00)*, Beijing, China, Oct. 16–20 2000.

J. H. Friedman. Exploratory projection pursuit. *J. Amer. Stat. Assoc.*, 82(397):249–266, Mar. 1987.

J. H. Friedman. Multivariate adaptive regression splines. *Annals of Statistics*, 19(1):1–67 (with comments, pp. 67–141), Mar. 1991.

J. H. Friedman and N. I. Fisher. Bump hunting in high-dimensional data. *Statistics and Computing*, 9(2): 123–143 (with discussion, pp. 143–162), Apr. 1999.

J. H. Friedman and W. Stuetzle. Projection pursuit regression. *J. Amer. Stat. Assoc.*, 76(376):817–823, Dec. 1981.

J. H. Friedman, W. Stuetzle, and A. Schroeder. Projection pursuit density estimation. *J. Amer. Stat. Assoc.*, 79(387):599–608, Sept. 1984.

J. H. Friedman and J. W. Tukey. A projection pursuit algorithm for exploratory data analysis. *IEEE Trans. Computers*, C–23:881–889, 1974.

C. Fyfe and R. J. Baddeley. Finding compact and sparse distributed representations of visual images. *Network: Computation in Neural Systems*, 6(3):333–344, Aug. 1995.

J.-L. Gauvain and C.-H. Lee. Maximum a-posteriori estimation for multivariate Gaussian mixture observations of Markov chains. *IEEE Trans. Speech and Audio Process.*, 2:1291–1298, 1994.

A. Gelman, J. B. Carlin, H. S. Stern, and D. B. Rubin. *Bayesian Data Analysis*. Texts in Statistical Science. Chapman & Hall, London, New York, 1995.

C. Genest and J. V. Zidek. Combining probability distributions: A critique and an annotated bibliography. *Statistical Science*, 1(1):114–135 (with discussion, pp. 135–148), Feb. 1986.

Z. Ghahramani. Solving inverse problems using an EM approach to density estimation. In M. C. Mozer, P. Smolensky, D. S. Touretzky, J. L. Elman, and A. S. Weigend, editors, *Proceedings of the 1993 Connectionist Models Summer School*, pages 316–323, 1994.

Z. Ghahramani and M. J. Beal. Variational inference for Bayesian mixture of factor analysers. In Solla et al. (2000), pages 449–455.

Z. Ghahramani and G. E. Hinton. The EM algorithm for mixtures of factor analyzers. Technical Report CRG–TR–96–1, University of Toronto, May 21 1996. Available online at `ftp://ftp.cs.toronto.edu/pub/zoubin/tr-96-1.ps.gz`.

Z. Ghahramani and M. I. Jordan. Supervised learning from incomplete data via an EM approach. In Cowan et al. (1994), pages 120–127.

W. Gilks, S. Richardson, and D. J. Spiegelhalter, editors. *Markov Chain Monte Carlo in Practice*. Chapman & Hall, London, New York, 1996.

M. Girolami, A. Cichoki, and S. Amari. A common neural network model for exploratory data analysis and independent component analysis. *IEEE Trans. Neural Networks*, 9(6):1495–1501, 1998.

M. Girolami and C. Fyfe. Stochastic ICA contrast maximization using Oja's nonlinear PCA algorithm. *Int. J. Neural Syst.*, 8(5–6):661–678, Oct./Dec. 1999.

F. Girosi, M. Jones, and T. Poggio. Regularization theory and neural networks architectures. *Neural Computation*, 7(2):219–269, Mar. 1995.

S. J. Godsill and P. J. W. Rayner. *Digital Audio Restoration: A Statistical Model-Based Approach*. Springer-Verlag, Berlin, 1998a.

S. J. Godsill and P. J. W. Rayner. Robust reconstruction and analysis of autoregressive signals in impulsive noise using the Gibbs sampler. *IEEE Trans. Speech and Audio Process.*, 6(4):352–372, July 1998b.

B. Gold and N. Morgan. *Speech and Audio Signal Processing: Processing and Perception of Speech and Music*. John Wiley & Sons, New York, London, Sydney, 2000.

D. E. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley, 1989.

G. H. Golub and C. F. van Loan. *Matrix Computations*. Johns Hopkins Press, Baltimore, third edition, 1996.

G. J. Goodhill and T. J. Sejnowski. A unifying objective function for topographic mappings. *Neural Computation*, 9(6):1291–1303, Aug. 1997.

R. A. Gopinath, B. Ramabhadran, and S. Dharanipragada. Factor analysis invariant to linear transformations of data. In *Proc. of the International Conference on Spoken Language Processing (ICSLP'98)*, Sydney, Australia, Nov. 30 – Dec. 4 1998.

W. P. Gouveia and J. A. Scales. Resolution of seismic waveform inversion: Bayes versus Occam. *Inverse Problems*, 13(2):323–349, Apr. 1997.

W. P. Gouveia and J. A. Scales. Bayesian seismic waveform inversion: Parameter estimation and uncertainty analysis. *J. of Geophysical Research*, 130(B2):2759–2779, 1998.

I. S. Gradshteyn and I. M. Ryzhik. *Table of Integrals, Series, and Products*. Academic Press, San Diego, fifth edition, 1994. Corrected and enlarged edition, edited by Alan Jeffrey.

R. M. Gray. Vector quantization. *IEEE ASSP Magazine*, pages 4–29, Apr. 1984.

R. M. Gray and D. L. Neuhoff. Quantization. *IEEE Trans. Inf. Theory*, 44(6):2325–2383, Oct. 1998.

M. Gyllenberg, T. Koski, E. Reilink, and M. Verlaan. Non-uniqueness in probabilistic numerical identification of bacteria. *J. Appl. Prob.*, 31:542–548, 1994.

P. Hall. On polynomial-based projection indices for exploratory projection pursuit. *Annals of Statistics*, 17 (2):589–605, June 1989.

W. J. Hardcastle, F. E. Gibbon, and W. Jones. Visual display of tongue-palate contact: Electropalatography in the assessment and remediation of speech disorders. *Brit. J. of Disorders of Communication*, 26:41–74, 1991a.

W. J. Hardcastle, F. E. Gibbon, and K. Nicolaidis. EPG data reduction methods and their implications for studies of lingual coarticulation. *J. of Phonetics*, 19:251–266, 1991b.

W. J. Hardcastle and N. Hewlett, editors. *Coarticulation: Theory, Data, and Techniques*. Cambridge Studies in Speech Science and Communication. Cambridge University Press, Cambridge, U.K., 1999.

W. J. Hardcastle, W. Jones, C. Knight, A. Trudgeon, and G. Calder. New developments in electropalatography: A state-of-the-art report. *J. Clinical Linguistics and Phonetics*, 3:1–38, 1989.

H. H. Harman. *Modern Factor Analysis*. University of Chicago Press, Chicago, second edition, 1967.

A. C. Harvey. *Forecasting, Structural Time Series Models and the Kalman Filter*. Cambridge University Press, 1991.

T. J. Hastie and W. Stuetzle. Principal curves. *J. Amer. Stat. Assoc.*, 84(406):502–516, June 1989.

T. J. Hastie and R. J. Tibshirani. *Generalized Additive Models*. Number 43 in Monographs on Statistics and Applied Probability. Chapman & Hall, London, New York, 1990.

G. T. Herman. *Image Reconstruction from Projections. The Fundamentals of Computer Tomography*. Academic Press, New York, 1980.

H. Hermansky. Perceptual linear predictive (PLP) analysis of speech. *J. Acoustic Soc. Amer.*, 87(4):1738–1752, Apr. 1990.

H. Hermansky and N. Morgan. RASTA processing of speech. *IEEE Trans. Speech and Audio Process.*, 2(4): 578–589, Oct. 1994.

J. A. Hertz, A. S. Krogh, and R. G. Palmer. *Introduction to the Theory of Neural Computation*. Number 1 in Santa Fe Institute Studies in the Sciences of Complexity Lecture Notes. Addison-Wesley, Reading, MA, USA, 1991.

G. E. Hinton. Products of experts. In D. Wilshaw, editor, *Proc. of the Ninth Int. Conf. on Artificial Neural Networks (ICANN99)*, pages 1–6, Edinburgh, UK, Sept. 7–10 1999. The Institution of Electrical Engineers.

G. E. Hinton, P. Dayan, and M. Revow. Modeling the manifolds of images of handwritten digits. *IEEE Trans. Neural Networks*, 8(1):65–74, Jan. 1997.

T. Holst, P. Warren, and F. Nolan. Categorising [s], [ʃ] and intermediate electropalographic patterns: Neural networks and other approaches. *European Journal of Disorders of Communication*, 30(2):161–174, 1995.

H. Hotelling. Analysis of a complex of statistical variables into principal components. *J. of Educational Psychology*, 24:417–441 and 498–520, 1933.

P. J. Huber. *Robust Statistics*. Wiley Series in Probability and Mathematical Statistics. John Wiley & Sons, New York, London, Sydney, 1981.

P. J. Huber. Projection pursuit. *Annals of Statistics*, 13(2):435–475 (with comments, pp. 475–525), June 1985.

D. Husmeier. *Neural Networks for Conditional Probability Estimation*. Perspectives in Neural Computing. Springer-Verlag, Berlin, 1999.

J.-N. Hwang, S.-R. Lay, M. Maechler, R. D. Martin, and J. Schimert. Regression modeling in back-propagation and projection pursuit learning. *IEEE Trans. Neural Networks*, 5(3):342–353, May 1994.

A. Hyvärinen. New approximations of differential entropy for independent component analysis and projection pursuit. In Jordan et al. (1998), pages 273–279.

A. Hyvärinen. Fast and robust fixed-point algorithms for independent component analysis. *IEEE Trans. Neural Networks*, 10(3):626–634, Oct. 1999a.

A. Hyvärinen. Survey on independent component analysis. *Neural Computing Surveys*, 2:94–128, 1999b.

A. Hyvärinen, J. Karhunen, and E. Oja. *Independent Component Analysis*. Wiley Series on Adaptive and Learning Systems for Signal Processing, Communications, and Control. John Wiley & Sons, New York, London, Sydney, 2001.

A. Hyvärinen and E. Oja. Independent component analysis: Algorithms and applications. *Neural Networks*, 13(4–5):411–430, June 2000.

N. Intrator and L. N. Cooper. Objective function formulation of the BCM theory of visual cortical plasticity: Statistical connections, stability conditions. *Neural Networks*, 5(1):3–17, 1992.

E. Isaacson and H. B. Keller. *Analysis of Numerical Methods*. John Wiley & Sons, New York, London, Sydney, 1966.

M. Isard and A. Blake. CONDENSATION — conditional density propagation for visual tracking. *Int. J. Computer Vision*, 29(1):5–28, 1998.

J. E. Jackson. *A User's Guide to Principal Components*. Wiley Series in Probability and Mathematical Statistics. John Wiley & Sons, New York, London, Sydney, 1991.

R. A. Jacobs, M. I. Jordan, S. J. Nowlan, and G. E. Hinton. Adaptive mixtures of local experts. *Neural Computation*, 3(1):79–87, 1991.

M. Jamshidian and P. M. Bentler. A quasi-Newton method for minimum trace factor analysis. *J. of Statistical Computation and Simulation*, 62(1–2):73–89, 1998.

N. Japkowicz, S. J. Hanson, and M. A. Gluck. Nonlinear autoassociation is not equivalent to PCA. *Neural Computation*, 12(3):531–545, Mar. 2000.

E. T. Jaynes. Prior probabilities. *IEEE Trans. Systems, Science, and Cybernetics*, SSC-4(3):227–241, 1968.

F. V. Jensen. *An Introduction to Bayesian Networks*. UCL Press, London, 1996.

I. T. Jolliffe. *Principal Component Analysis*. Springer Series in Statistics. Springer-Verlag, Berlin, 1986.

M. C. Jones. *The Projection Pursuit Algorithm for Exploratory Data Analysis*. PhD thesis, University of Bath, 1983.

M. C. Jones and R. Sibson. What is projection pursuit? *Journal of the Royal Statistical Society, A*, 150(1): 1–18 (with comments, pp. 19–36), 1987.

W. Jones and W. J. Hardcastle. New developments in EPG3 software. *European Journal of Disorders of Communication*, 30(2):183–192, 1995.

M. I. Jordan. Motor learning and the degrees of freedom problem. In M. Jeannerod, editor, *Attention and Performance XIII*, pages 796–836. Lawrence Erlbaum Associates, Hillsdale, New Jersey and London, 1990.

M. I. Jordan, editor. *Learning in Graphical Models*, Adaptive Computation and Machine Learning series, 1998. MIT Press. Proceedings of the NATO Advanced Study Institute on Learning in Graphical Models, held in Erice, Italy, September 27 – October 7, 1996.

M. I. Jordan, Z. Ghahramani, T. S. Jaakkola, and L. K. Saul. An introduction to variational methods for graphical models. *Machine Learning*, 37(2):183–233, Nov. 1999.

M. I. Jordan and R. A. Jacobs. Hierarchical mixtures of experts and the EM algorithm. *Neural Computation*, 6(2):181–214, Mar. 1994.

M. I. Jordan, M. J. Kearns, and S. A. Solla, editors. *Advances in Neural Information Processing Systems*, volume 10, 1998. MIT Press, Cambridge, MA.

M. I. Jordan and D. E. Rumelhart. Forward models: Supervised learning with a distal teacher. *Cognitive Science*, 16(3):307–354, July–Sept. 1992.

K. G. Jöreskog. Some contributions to maximum likelihood factor analysis. *Psychometrika*, 32(4):443–482, Dec. 1967.

K. G. Jöreskog. A general approach to confirmatory maximum likelihood factor analysis. *Psychometrika*, 34 (2):183–202, June 1969.

H. F. Kaiser. The varimax criterion for analytic rotation in factor analysis. *Psychometrika*, 23(3):187–200, Sept. 1958.

N. Kambhatla and T. K. Leen. Dimension reduction by local principal component analysis. *Neural Computation*, 9(7):1493–1516, Oct. 1997.

G. K. Kanji. 100 *Statistical Tests*. Sage Publications, London, 1993.

J. N. Kapur. *Maximum-Entropy Models in Science and Engineering*. John Wiley & Sons, New York, London, Sydney, 1989.

J. Karhunen and J. Joutsensalo. Representation and separation of signals using nonlinear PCA type learning. *Neural Networks*, 7(1):113–127, 1994.

R. E. Kass and L. Wasserman. The selection of prior distributions by formal rules. *J. Amer. Stat. Assoc.*, 91 (435):1343–1370, Sept. 1996.

M. S. Kearns, S. A. Solla, and D. A. Cohn, editors. *Advances in Neural Information Processing Systems*, volume 11, 1999. MIT Press, Cambridge, MA.

B. Kégl, A. Krzyzak, T. Linder, and K. Zeger. Learning and design of principal curves. *IEEE Trans. on Pattern Anal. and Machine Intel.*, 22(3):281–297, Mar. 2000.

M. G. Kendall and A. Stuart. *The Advanced Theory of Statistics Vol. 1: Distribution Theory.* Charles Griffin & Company Ltd., London, fourth edition, 1977.

W. M. Kier and K. K. Smith. Tongues, tentacles and trunks: The biomechanics of movement in muscular-hydrostats. *Zoological Journal of the Linnean Society*, 83:307–324, 1985.

S. King and A. Wrench. Dynamical system modelling of articulator movement. In Ohala et al. (1999), pages 2259–2262.

B. E. D. Kingsbury, N. Morgan, and S. Greenberg. Robust speech recognition using the modulation spectrogram. *Speech Communication*, 25(1–3):117–132, Aug. 1998.

T. K. Kohonen. *Self-Organizing Maps.* Number 30 in Springer Series in Information Sciences. Springer-Verlag, Berlin, 1995.

A. C. Kokaram, R. D. Morris, W. J. Fitzgerald, and P. J. W. Rayner. Interpolation of missing data in image sequences. *IEEE Trans. on Image Processing*, 4(11):1509–1519, Nov. 1995.

J. F. Kolen and J. B. Pollack. Back propagation is sensitive to initial conditions. *Complex Systems*, 4(3): 269–280, 1990.

A. C. Konstantellos. Unimodality conditions for Gaussian sums. *IEEE Trans. Automat. Contr.*, AC–25(4): 838–839, Aug. 1980.

M. A. Kramer. Nonlinear principal component analysis using autoassociative neural networks. *Journal of the American Institute of Chemical Engineers*, 37(2):233–243, Feb. 1991.

J. B. Kruskal and M. Wish. *Multidimensional Scaling.* Number 07–011 in Sage University Paper Series on Quantitative Applications in the Social Sciences. Sage Publications, Beverly Hills, 1978.

W. J. Krzanowski. *Principles of Multivariate Analysis: A User's Perspective.* Number 3 in Oxford Statistical Science Series. Oxford University Press, New York, Oxford, 1988.

S. Y. Kung, K. I. Diamantaras, and J. S. Taur. Adaptive principal component extraction (APEX) and applications. *IEEE Trans. Signal Processing*, 42(5):1202–1217, May 1994.

O. M. Kvalheim. The latent variable. *Chemometrics and Intelligent Laboratory Systems*, 14:1–3, 1992.

P. Ladefoged. Articulatory parameters. *Language and Speech*, 23(1):25–30, Jan.–Mar. 1980.

P. Ladefoged. *A Course in Phonetics.* Harcourt College Publishers, Fort Worth, fourth edition, 2000.

N. Laird. Nonparametric maximum likelihood estimation of a mixing distribution. *J. Amer. Stat. Assoc.*, 73 (364):805–811, Dec. 1978.

J. N. Larar, J. Schroeter, and M. M. Sondhi. Vector quantisation of the articulatory space. *IEEE Trans. Acoust., Speech, and Signal Process.*, ASSP-36(12):1812–1818, Dec. 1988.

F. Lavagetto. Time-delay neural networks for estimating lip movements from speech analysis: A useful tool in audio-video synchronization. *IEEE Trans. Circuits and Systems for video technology*, 7(5):786–800, Oct. 1997.

E. L. Lawler, J. K. Lenstra, A. H. G. Rinnooy Kan, and D. B. Shmoys, editors. *The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization*. Wiley Series in Discrete Mathematics and Optimization. John Wiley & Sons, Chichester, England, 1985.

D. N. Lawley. A modified method of estimation in factor analysis and some large sample results. *Nord. Psykol. Monogr. Ser.*, 3:35–42, 1953.

P. F. Lazarsfeld and N. W. Henry. *Latent Structure Analysis*. Houghton-Mifflin, Boston, 1968.

M. LeBlanc and R. Tibshirani. Adaptive principal surfaces. *J. Amer. Stat. Assoc.*, 89(425):53–64, Mar. 1994.

D. D. Lee and H. Sompolinsky. Learning a continuous hidden variable model for binary data. In Kearns et al. (1999), pages 515–521.

T.-W. Lee, M. Girolami, and T. J. Sejnowski. Independent component analysis using an extended infomax algorithm for mixed subgaussian and supergaussian sources. *Neural Computation*, 11(2):417–441, Feb. 1999.

C. J. Leggeter and P. C. Woodland. Maximum likelihood linear regression for speaker adaptation of continuous density hidden Markov models. *Computer Speech and Language*, 9(2):171–185, Apr. 1995.

S. E. Levinson and C. E. Schmidt. Adaptive computation of articulatory parameters from the speech signal. *J. Acoustic Soc. Amer.*, 74(4):1145–1154, Oct. 1983.

M. S. Lewicki and T. J. Sejnowski. Learning overcomplete representations. *Neural Computation*, 12(2):337–365, Feb. 2000.

A. M. Liberman, F. S. Cooper, D. P. Shankweiler, and M. Studdert-Kennedy. Perception of the speech code. *Psychological Review*, 74(6):431–461, 1967.

A. M. Liberman and I. G. Mattingly. The motor theory of speech perception revised. *Cognition*, 21:1–36, 1985.

B. G. Lindsay. The geometry of mixture likelihoods: A general theory. *Annals of Statistics*, 11(1):86–94, Mar. 1983.

R. Linsker. An application of the principle of maximum information preservation to linear systems. In D. S. Touretzky, editor, *Advances in Neural Information Processing Systems*, volume 1, pages 186–194. Morgan Kaufmann, San Mateo, 1989.

R. J. A. Little. Regression with missing X's: A review. *J. Amer. Stat. Assoc.*, 87(420):1227–1237, Dec. 1992.

R. J. A. Little and D. B. Rubin. *Statistical Analysis with Missing Data*. Wiley Series in Probability and Mathematical Statistics. John Wiley & Sons, New York, London, Sydney, 1987.

S. P. Luttrell. A Bayesian analysis of self-organizing maps. *Neural Computation*, 6(5):767–794, Sept. 1994.

D. J. C. MacKay. Bayesian interpolation. *Neural Computation*, 4(3):415–447, May 1992a.

D. J. C. MacKay. A practical Bayesian framework for backpropagation networks. *Neural Computation*, 4(3): 448–472, May 1992b.

D. J. C. MacKay. Bayesian neural networks and density networks. *Nuclear Instruments and Methods in Physics Research A*, 354(1):73–80, Jan. 1995a.

D. J. C. MacKay. Probable networks and plausible predictions — a review of practical Bayesian methods for supervised neural networks. *Network: Computation in Neural Systems*, 6(3):469–505, 1995b.

D. J. C. MacKay. Maximum likelihood and covariant algorithms for independent component analysis. Draft 3.7, Cavendish Laboratory, University of Cambridge, Dec. 19 1996. Available online at `http://wol.ra.phy.cam.ac.uk/mackay/abstracts/ica.html`.

D. J. C. MacKay. Comparison of approximate methods for handling hyperparameters. *Neural Computation*, 11(5):1035–1068, July 1999.

S. Maeda. A digital simulation method of the vocal tract system. *Speech Communication*, 1(3–4):199–229, 1982.

S. Makeig, T.-P. Jung, A. J. Bell, D. Ghahremani, and T. J. Sejnowski. Blind separation of auditory event-related brain responses into independent components. *Proc. Natl. Acad. Sci. USA*, 94:10979–10984, Sept. 1997.

E. C. Malthouse. Limitations of nonlinear PCA as performed with generic neural networks. *IEEE Trans. Neural Networks*, 9(1):165–173, Jan. 1998.

J. Mao and A. K. Jain. Artificial neural networks for feature extraction and multivariate data projection. *IEEE Trans. Neural Networks*, 6(2):296–317, Mar. 1995.

A. Marchal and W. J. Hardcastle. ACCOR: Instrumentation and database for the cross-language study of coarticulation. *Language and Speech*, 36(2, 3):137–153, 1993.

K. V. Mardia, J. T. Kent, and J. M. Bibby. *Multivariate Analysis*. Probability and Mathematical Statistics Series. Academic Press, New York, 1979.

A. D. Marrs and A. R. Webb. Exploratory data analysis using radial basis function latent variable models. In Kearns et al. (1999), pages 529–535.

T. M. Martinetz and K. J. Schulten. Topology representing networks. *Neural Networks*, 7(3):507–522, 1994.

G. P. McCabe. Principal variables. *Technometrics*, 26(2):137–144, 1984.

R. P. McDonald. *Factor Analysis and Related Methods*. Lawrence Erlbaum Associates, Hillsdale, New Jersey and London, 1985.

R. S. McGowan. Recovering articulatory movement from formant frequency trajectories using task dynamics and a genetic algorithm: Preliminary model tests. *Speech Communication*, 14(1):19–48, Feb. 1994.

R. S. McGowan and A. Faber. Introduction to papers on speech recognition and perception from an articulatory point of view. *J. Acoustic Soc. Amer.*, 99(3):1680–1682, Mar. 1996.

G. J. McLachlan and T. Krishnan. *The EM Algorithm and Extensions*. Wiley Series in Probability and Mathematical Statistics. John Wiley & Sons, 1997.

G. J. McLachlan and D. Peel. *Finite Mixture Models*. Wiley Series in Probability and Mathematical Statistics. John Wiley & Sons, 2000.

X.-L. Meng and D. van Dyk. The EM algorithm — an old folk-song sung to a fast new tune. *Journal of the Royal Statistical Society, B*, 59(3):511–540 (with discussion, pp. 541–567), 1997.

P. Mermelstein. Determination of vocal-tract shape from measured formant frequencies. *J. Acoustic Soc. Amer.*, 41(5):1283–1294, 1967.

P. Mermelstein. Articulatory model for the study of speech production. *J. Acoustic Soc. Amer.*, 53(4):1070–1082, 1973.

L. Mirsky. *An Introduction to Linear Algebra*. Clarendon Press, Oxford, 1955. Reprinted in 1982 by Dover Publications.

B. Moghaddam and A. Pentland. Probabilistic visual learning for object representation. *IEEE Trans. on Pattern Anal. and Machine Intel.*, 19(7):696–710, July 1997.

J. Moody and C. J. Darken. Fast learning in networks of locally-tuned processing units. *Neural Computation*, 1(2):281–294, Summer 1989.

D. F. Morrison. *Multivariate Statistical Methods*. McGraw-Hill, New York, third edition, 1990.

K. Mosegaard and A. Tarantola. Monte-Carlo sampling of solutions to inverse problems. *J. of Geophysical Research—Solid Earth*, 100(B7):12431–12447, 1995.

É. Moulines, J.-F. Cardoso, and E. Gassiat. Maximum likelihood for blind separation and deconvolution of noisy signals using mixture models. In *Proc. of the IEEE Int. Conf. on Acoustics, Speech and Sig. Proc. (ICASSP'97)*, volume 5, pages 3617–3620, Munich, Germany, Apr. 21–24 1997.

J. R. Movellan, P. Mineiro, and R. J. Williams. Modeling path distributions using partially observable diffusion networks: A Monte-Carlo approach. Technical Report 99.01, Department of Cognitive Science, University of California, San Diego, June 1999. Available online at `http://hci.ucsd.edu/cogsci/tech_reports/faculty_pubs/99_01.ps`.

F. Mulier and V. Cherkassky. Self-organization as an iterative kernel smoothing process. *Neural Computation*, 7(6):1165–1177, Nov. 1995.

I. T. Nabney, D. Cornford, and C. K. I. Williams. Bayesian inference for wind field retrieval. *Neurocomputing*, 30(1–4):3–11, Jan. 2000.

J.-P. Nadal and N. Parga. Non-linear neurons in the low-noise limit: A factorial code maximizes information transfer. *Network: Computation in Neural Systems*, 5(4):565–581, Nov. 1994.

R. M. Neal. Probabilistic inference using Markov chain Monte Carlo methods. Technical Report CRG–TR–93–1, Dept. of Computer Science, University of Toronto, Sept. 1993. Available online at `ftp://ftp.cs.toronto.edu/pub/radford/review.ps.Z`.

R. M. Neal. *Bayesian Learning for Neural Networks*. Springer Series in Statistics. Springer-Verlag, Berlin, 1996.

R. M. Neal and P. Dayan. Factor analysis using delta-rule wake-sleep learning. *Neural Computation*, 9(8): 1781–1803, Nov. 1997.

R. M. Neal and G. E. Hinton. A view of the EM algorithm that justifies incremental, sparse, and other variants. In Jordan (1998), pages 355–368. Proceedings of the NATO Advanced Study Institute on Learning in Graphical Models, held in Erice, Italy, September 27 – October 7, 1996.

W. L. Nelson. Physical principles for economies of skilled movements. *Biol. Cybern.*, 46(2):135–147, 1983.

N. Nguyen. EPG bidimensional data reduction. *European Journal of Disorders of Communication*, 30:175–182, 1995.

N. Nguyen, P. Hoole, and A. Marchal. Regenerating the spectral shape of [s] and [ʃ] from a limited set of articulatory parameters. *J. Acoustic Soc. Amer.*, 96(1):33–39, July 1994.

N. Nguyen, A. Marchal, and A. Content. Modeling tongue-palate contact patterns in the production of speech. *J. of Phonetics*, 24(1):77–97, Jan. 1996.

K. Nicolaidis and W. J. Hardcastle. Articulatory-acoustic analysis of selected English sentences from the EUR-ACCOR corpus. Technical report, SPHERE (Human capital and mobility program), 1994.

K. Nicolaidis, W. J. Hardcastle, A. Marchal, and N. Nguyen-Trong. Comparing phonetic, articulatory, acoustic and aerodynamic signal representations. In M. Cooke, S. Beet, and M. Crawford, editors, *Visual Representations of Speech Signals*, pages 55–82. John Wiley & Sons, 1993.

M. A. L. Nicolelis. Actions from thoughts. *Nature*, 409(6818):403–407, Jan. 18 2001.

M. Niranjan, editor. *Proc. of the 1998 IEEE Signal Processing Society Workshop on Neural Networks for Signal Processing (NNSP98)*, Cambridge, UK, Aug. 31 – Sept. 2 1998.

D. A. Nix and J. E. Hogden. Maximum-likelihood continuity mapping (MALCOM): An alternative to HMMs. In Kearns et al. (1999), pages 744–750.

J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer Series in Operations Research. Springer-Verlag, 1999.

J. J. Ohala, Y. Hasegawa, M. Ohala, D. Granville, and A. C. Bailey, editors. *Proc. of the 14th International Congress of Phonetic Sciences (ICPhS'99)*, San Francisco, USA, Aug. 1–7 1999.

E. Oja. Principal components, minor components, and linear neural networks. *Neural Networks*, 5(6):927–935, Nov.–Dec. 1992.

B. A. Olshausen and D. J. Field. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, 381(6583):607–609, June 13 1996.

B. A. Olshausen and D. J. Field. Sparse coding with an overcomplete basis set: A strategy employed by V1? *Vision Res.*, 37(23):3311–3325, Dec. 1997.

M. W. Oram, P. Földiák, D. I. Perret, and F. Sengpiel. The 'ideal homunculus': Decoding neural population signals. *Trends Neurosci.*, 21(6):259–265, June 1998.

D. Ormoneit and V. Tresp. Penalized likelihood and Bayesian estimation for improving Gaussian mixture probability density estimates. *IEEE Trans. Neural Networks*, 9(4):639–650, July 1998.

M. Ostendorf, V. V. Digalakis, and O. A. Kimball. From HMM's to segment models: A unified view of stochastic modeling for speech recognition. *IEEE Trans. Speech and Audio Process.*, 4(5):360–378, Sept. 1996.

G. Papcun, J. Hochberg, T. R. Thomas, F. Laroche, J. Zacks, and S. Levy. Inferring articulation and recognizing gestures from acoustics with a neural network trained on x-ray microbeam data. *J. Acoustic Soc. Amer.*, 92(2):688–700, Aug. 1992.

J. Park and I. W. Sandberg. Approximation and radial-basis-function networks. *Neural Computation*, 5(2): 305–316, Mar. 1993.

R. L. Parker. *Geophysical Inverse Theory.* Princeton University Press, Princeton, 1994.

J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference.* Morgan Kaufmann, San Mateo, 1988.

B. A. Pearlmutter. Gradient calculation for dynamic recurrent neural networks: A survey. *IEEE Trans. Neural Networks*, 6(5):1212–1228, 1995.

B. A. Pearlmutter and L. C. Parra. A context-sensitive generalization of ICA. In *International Conference on Neural Information Processing (ICONIP–96), Hong Kong*, pages 151–157, Sept. 1996.

K. Pearson. On lines and planes of closest fit to systems of points in space. *Philosophical Magazine*, 2(6): 559–572, 1901.

D. Peel and G. J. McLachlan. Robust mixture modelling using the t distribution. *Statistics and Computing*, 10(4):339–348, Oct. 2000.

H.-O. Peitgen, H. Jürgens, and D. Saupe. *Chaos and Fractals: New Frontiers of Science.* Springer-Verlag, New York, 1992.

J. Picone, S. Pike, R. Reagan, T. Kamm, J. Bridle, L. Deng, J. Ma, H. Richards, and M. Schuster. Initial evaluation of hidden dynamic models on conversational speech. In *Proc. of the IEEE Int. Conf. on Acoustics, Speech and Sig. Proc. (ICASSP'99)*, volume 1, pages 109–112, Phoenix, Arizona, USA, May 15–19 1999.

A. Pisani. A nonparametric and scale-independent method for cluster-analysis. 1. The univariate case. *Monthly Notices of the Royal Astronomical Society*, 265(3):706–726, Dec. 1993.

C. Posse. An effective two-dimensional projection pursuit algorithm. *Communications in Statistics — Simulation and Computation*, 19(4):1143–1164, 1990.

C. Posse. Tools for two-dimensional exploratory projection pursuit. *Journal of Computational and Graphical Statistics*, 4:83–100, 1995.

S. Pratt, A. T. Heintzelman, and D. S. Ensrud. The efficacy of using the IBM Speech Viewer vowel accuracy module to treat young children with hearing impairment. *Journal of Speech and Hearing Research*, 29: 99–105, 1993.

F. P. Preparata and M. I. Shamos. *Computational Geometry: An Introduction.* Monographs in Computer Science. Springer-Verlag, New York, 1985.

W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes in C: The Art of Scientific Computing.* Cambridge University Press, Cambridge, U.K., second edition, 1992.

L. Rabiner and B.-H. Juang. *Fundamentals of Speech Recognition.* Signal Processing Series. Prentice-Hall, Englewood Cliffs, N.J., 1993.

M. G. Rahim, C. C. Goodyear, W. B. Kleijn, J. Schroeter, and M. M. Sondhi. On the use of neural networks in articulatory speech synthesis. *J. Acoustic Soc. Amer.*, 93(2):1109–1121, Feb. 1993.

R. A. Redner and H. F. Walker. Mixture densities, maximum likelihood and the EM algorithm. *SIAM Review*, 26(2):195–239, Apr. 1984.

K. Reinhard and M. Niranjan. Parametric subspace modeling of speech transitions. *Speech Communication*, 27(1):19–42, Feb. 1999.

M. Revow, C. K. I. Williams, and G. Hinton. Using generative models for handwritten digit recognition. *IEEE Trans. on Pattern Anal. and Machine Intel.*, 18(6):592–606, June 1996.

H. B. Richards and J. S. Bridle. The HDM: a segmental hidden dynamic model of coarticulation. In *Proc. of the IEEE Int. Conf. on Acoustics, Speech and Sig. Proc. (ICASSP'99)*, volume I, pages 357–360, Phoenix, Arizona, USA, May 15–19 1999.

S. Richardson and P. J. Green. On Bayesian analysis of mixtures with an unknown number of components. *Journal of the Royal Statistical Society, B*, 59(4):731–758, 1997.

B. D. Ripley. *Pattern Recognition and Neural Networks*. Cambridge University Press, Cambridge, U.K., 1996.

S. J. Roberts. Parametric and non-parametric unsupervised cluster analysis. *Pattern Recognition*, 30(2): 261–272, Feb. 1997.

S. J. Roberts, D. Husmeier, I. Rezek, and W. Penny. Bayesian approaches to Gaussian mixture modeling. *IEEE Trans. on Pattern Anal. and Machine Intel.*, 20(11):1133–1142, 1998.

W. J. J. Roberts and Y. Ephraim. Hidden Markov modeling of speech using Toeplitz covariance matrices. *Speech Communication*, 31(1):1–14, May 2000.

A. J. Robinson. An application of recurrent nets to phone probability estimation. *IEEE Trans. Neural Networks*, 5(2):298–305, Mar. 1994.

T. Rögnvaldsson. On Langevin updating in multilayer perceptrons. *Neural Computation*, 6(5):916–926, Sept. 1994.

R. Rohwer and J. C. van der Rest. Minimum description length, regularization, and multimodal data. *Neural Computation*, 8(3):595–609, Apr. 1996.

E. T. Rolls and A. Treves. *Neural Networks and Brain Function*. Oxford University Press, 1998.

K. Rose. Deterministic annealing for clustering, compression, classification, regression, and related optimization problems. *Proc. IEEE*, 86(11):2210–2239, Nov. 1998.

R. C. Rose, J. Schroeter, and M. M. Sondhi. The potential role of speech production models in automatic speech recognition. *J. Acoustic Soc. Amer.*, 99(3):1699–1709 (with comments, pp. 1710–1717), Mar. 1996.

E. Z. Rothkopf. A measure of stimulus similarity and errors in some paired-associate learning tasks. *J. of Experimental Psychology*, 53(2):94–101, 1957.

B. Rotman and G. T. Kneebone. *The Theory of Sets & Transfinite Numbers*. Oldbourne, London, 1966.

S. Roweis. EM algorithms for PCA and SPCA. In Jordan et al. (1998), pages 626–632.

S. Roweis. Constrained hidden Markov models. In Solla et al. (2000), pages 782–788.

S. T. Roweis and L. K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290 (5500):2323–2326, Dec. 22 2000.

A. E. Roy. *Orbital Motion*. Adam Hilger Ltd., Bristol, 1978.

D. B. Rubin. *Multiple Imputation for Nonresponse in Surveys*. Wiley Series in Probability and Mathematical Statistics. John Wiley & Sons, 1987.

D. B. Rubin and D. T. Thayer. EM algorithms for ML factor analysis. *Psychometrika*, 47(1):69–76, Mar. 1982.

D. B. Rubin and D. T. Thayer. More on EM for ML factor analysis. *Psychometrika*, 48(2):253–257, June 1983.

P. Rubin, T. Baer, and P. Mermelstein. An articulatory synthesizer for perceptual research. *J. Acoustic Soc. Amer.*, 70(2):321–328, Aug. 1981.

E. Saltzman and J. A. Kelso. Skilled actions: a task-dynamic approach. *Psychological Review*, 94(1):84–106, Jan. 1987.

J. W. Sammon, Jr. A nonlinear mapping for data structure analysis. *IEEE Trans. Computers*, C–18(5): 401–409, May 1969.

T. D. Sanger. Optimal unsupervised learning in a single-layer linear feedforward neural network. *Neural Networks*, 2:459–473, 1989.

T. D. Sanger. Probability density estimation for the interpretation of neural population codes. *J. Neurophysiol.*, 76(4):2790–2793, Oct. 1996.

L. K. Saul and M. G. Rahim. Markov processes on curves. *Machine Learning*, 41(3):345–363, Dec. 2000a.

L. K. Saul and M. G. Rahim. Maximum likelihood and minimum classification error factor analysis for automatic speech recognition. *IEEE Trans. Speech and Audio Process.*, 8(2):115–125, Mar. 2000b.

E. Saund. Dimensionality-reduction using connectionist networks. *IEEE Trans. on Pattern Anal. and Machine Intel.*, 11(3):304–314, Mar. 1989.

J. A. Scales and M. L. Smith. *Introductory Geophysical Inverse Theory*. Samizdat Press, 1998. Freely available in draft form from `http://samizdat.mines.edu/inverse_theory/`.

F. Scarselli and A. C. Tsoi. Universal approximation using feedforward neural networks: A survey of some existing methods, and some new results. *Neural Networks*, 11(1):15–37, Jan. 1998.

J. L. Schafer. *Analysis of Incomplete Multivariate Data*. Number 72 in Monographs on Statistics and Applied Probability. Chapman & Hall, London, New York, 1997.

B. Schölkopf, C. J. C. Burges, and A. J. Smola, editors. *Advances in Kernel Methods. Support Vector Learning*. MIT Press, 1999a.

B. Schölkopf, S. Mika, C. J. C. Burges, P. Knirsch, K.-R. Müller, G. Rätsch, and A. Smola. Input space vs. feature space in kernel-based methods. *IEEE Trans. Neural Networks*, 10(5):1000–1017, Sept. 1999b.

B. Schölkopf, A. Smola, and K.-R. Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10(5):1299–1319, July 1998.

M. R. Schroeder. Determination of the geometry of the human vocal tract by acoustic measurements. *J. Acoustic Soc. Amer.*, 41(4):1002–1010, 1967.

J. Schroeter and M. M. Sondhi. Dynamic programming search of articulatory codebooks. In *Proc. of the IEEE Int. Conf. on Acoustics, Speech and Sig. Proc. (ICASSP'89)*, volume 1, pages 588–591, Glasgow, UK, May 23–26 1989.

J. Schroeter and M. M. Sondhi. Techniques for estimating vocal-tract shapes from the speech signal. *IEEE Trans. Speech and Audio Process.*, 2(1):133–150, Jan. 1994.

M. Schuster. *On Supervised Learning from Sequential Data with Applications for Speech Recognition*. PhD thesis, Graduate School of Information Science, Nara Institute of Science and Technology, 1999.

D. W. Scott. *Multivariate Density Estimation. Theory, Practice, and Visualization*. Wiley Series in Probability and Mathematical Statistics. John Wiley & Sons, New York, London, Sydney, 1992.

D. W. Scott and J. R. Thompson. Probability density estimation in higher dimensions. In J. E. Gentle, editor, *Computer Science and Statistics: Proceedings of the Fifteenth Symposium on the Interface*, pages 173–179, Amsterdam, New York, Oxford, 1983. North Holland-Elsevier Science Publishers.

R. N. Shepard. Analysis of proximities as a technique for the study of information processing in man. *Human Factors*, 5:33–48, 1963.

K. Shirai and T. Kobayashi. Estimating articulatory motion from speech wave. *Speech Communication*, 5(2): 159–170, June 1986.

M. F. Shlesinger, G. M. Zaslavsky, and U. Frisch, editors. *Lévy Flights and Related Topics in Physics*. Number 450 in Lecture Notes in Physics. Springer-Verlag, Berlin, 1995. Proceedings of the International Workshop held at Nice, France, 27–30 June 1994.

B. W. Silverman. *Density Estimation for Statistics and Data Analysis*. Number 26 in Monographs on Statistics and Applied Probability. Chapman & Hall, London, New York, 1986.

L. Sirovich and M. Kirby. Low-dimensional procedure for the identification of human faces. *J. Opt. Soc. Amer. A*, 4(3):519–524, Mar. 1987.

D. S. Sivia. *Data Analysis. A Bayesian Tutorial*. Oxford University Press, New York, Oxford, 1996.

R. Snieder and J. Trampert. *Inverse Problems in Geophysics*. Samizdat Press, 1999. Freely available from `http://samizdat.mines.edu/snieder_trampert/`.

S. A. Solla, T. K. Leen, and K.-R. Müller, editors. *Advances in Neural Information Processing Systems*, volume 12, 2000. MIT Press, Cambridge, MA.

V. N. Sorokin. Determination of vocal-tract shape for vowels. *Speech Communication*, 11(1):71–85, Mar. 1992.

V. N. Sorokin, A. S. Leonov, and A. V. Trushkin. Estimation of stability and accuracy of inverse problem solution for the vocal tract. *Speech Communication*, 30(1):55–74, Jan. 2000.

C. Spearman. General intelligence, objectively determined and measured. *Am. J. Psychol.*, 15:201–293, 1904.

D. F. Specht. A general regression neural network. *IEEE Trans. Neural Networks*, 2(6):568–576, Nov. 1991.

M. Spivak. *Calculus on Manifolds: A Modern Approach to Classical Theorems of Advanced Calculus*. Addison-Wesley, Reading, MA, USA, 1965.

M. Spivak. *Calculus*. Addison-Wesley, Reading, MA, USA, 1967.

M. Stone. Toward a model of three-dimensional tongue movement. *J. of Phonetics*, 19:309–320, 1991.

N. V. Swindale. The development of topography in the visual cortex: A review of models. *Network: Computation in Neural Systems*, 7(2):161–247, May 1996.

A. Tarantola. *Inverse Problem Theory: Methods for Data Fitting and Model Parameter Estimation*. Elsevier Science Publishers B.V., Amsterdam, The Netherlands, 1987.

J. B. Tenenbaum. Mapping a manifold of perceptual observations. In Jordan et al. (1998), pages 682–688.

J. B. Tenenbaum, V. de Silva, and J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, Dec. 22 2000.

G. Tesauro, D. S. Touretzky, and T. K. Leen, editors. *Advances in Neural Information Processing Systems*, volume 7, 1995. MIT Press, Cambridge, MA.

R. J. Tibshirani. Principal curves revisited. *Statistics and Computing*, 2:183–190, 1992.

A. N. Tikhonov and V. Y. Arsenin. *Solutions of Ill-Posed Problems*. Scripta Series in Mathematics. John Wiley & Sons, New York, London, Sydney, 1977. Translation editor: Fritz John.

M. E. Tipping and C. M. Bishop. Mixtures of probabilistic principal component analyzers. *Neural Computation*, 11(2):443–482, Feb. 1999a.

M. E. Tipping and C. M. Bishop. Probabilistic principal component analysis. *Journal of the Royal Statistical Society, B*, 61(3):611–622, 1999b.

D. M. Titterington, A. F. M. Smith, and U. E. Makov. *Statistical Analysis of Finite Mixture Distributions*. Wiley Series in Probability and Mathematical Statistics. John Wiley & Sons, New York, London, Sydney, 1985.

L. Tong, R.-W. Liu, V. C. Soon, and Y.-F. Huang. The indeterminacy and identifiability of blind identification. *IEEE Trans. Circuits and Systems*, 38(5):499–509, May 1991.

V. Tresp, R. Neuneier, and S. Ahmad. Efficient methods for dealing with missing data in supervised learning. In Tesauro et al. (1995), pages 689–696.

A. Treves, S. Panzeri, E. T. Rolls, M. Booth, and E. A. Wakeman. Firing rate distributions and efficiency of information transmission of inferior temporal cortex neurons to natural visual stimuli. *Neural Computation*, 11(3):601–632, Mar. 1999.

A. Treves and E. T. Rolls. What determines the capacity of autoassociative memories in the brain? *Network: Computation in Neural Systems*, 2(4):371–397, Nov. 1991.

A. C. Tsoi. Recurrent neural network architectures — an overview. In C. L. Giles and M. Gori, editors, *Adaptive Processing of Temporal Information*, volume 1387 of *Lecture Notes in Artificial Intelligence*, pages 1–26. Springer-Verlag, New York, 1998.

UCLA. Artificial EPG palate image. The UCLA Phonetics Lab. Available online at `http://www.humnet.ucla.edu/humnet/linguistics/faciliti/facilities/physiology/EGP_picture.JPG`, Feb. 1, 2000.

N. Ueda, R. Nakano, Z. Ghahramani, and G. E. Hinton. SMEM algorithm for mixture models. *Neural Computation*, 12(9):2109–2128, Sept. 2000.

A. Utsugi. Hyperparameter selection for self-organizing maps. *Neural Computation*, 9(3):623–635, Apr. 1997a.

A. Utsugi. Topology selection for self-organizing maps. *Network: Computation in Neural Systems*, 7(4): 727–740, 1997b.

A. Utsugi. Bayesian sampling and ensemble learning in generative topographic mapping. *Neural Processing Letters*, 12(3):277–290, Dec. 2000.

A. Utsugi and T. Kumagai. Bayesian analysis of mixtures of factor analyzers. *Neural Computation*, 13(5): 993–1002, May 2001.

V. N. Vapnik and S. Mukherjee. Support vector method for multivariate density estimation. In Solla et al. (2000), pages 659–665.

S. V. Vaseghi. *Advanced Signal Processing and Digital Noise Reduction.* John Wiley & Sons, New York, London, Sydney, second edition, 2000.

S. V. Vaseghi and P. J. W. Rayner. Detection and suppression of impulsive noise in speech-communication systems. *IEE Proc. I (Communications, Speech and Vision)*, 137(1):38–46, Feb. 1990.

T. Villmann, R. Der, M. Hermann, and T. M. Martinetz. Topology preservation in self-organizing feature maps: Exact definition and measurement. *IEEE Trans. Neural Networks*, 8(2):256–266, Mar. 1997.

W. E. Vinje and J. L. Gallant. Sparse coding and decorrelation in primary visual cortex during natural vision. *Science*, 287(5456):1273–1276, Feb. 18 2000.

H. M. Wagner. *Principles of Operations Research with Applications to Managerial Decisions*. Prentice-Hall, Englewood Cliffs, N.J., second edition, 1975.

A. Webb. *Statistical Pattern Recognition.* Edward Arnold, 1999.

A. R. Webb. Multidimensional scaling by iterative majorization using radial basis functions. *Pattern Recognition*, 28(5):753–759, May 1995.

E. J. Wegman. Hyperdimensional data analysis using parallel coordinates. *J. Amer. Stat. Assoc.*, 85(411): 664–675, Sept. 1990.

J. R. Westbury. *X-Ray Microbeam Speech Production Database User's Handbook Version 1.0.* Waisman Center on Mental Retardation & Human Development, University of Wisconsin, Madison, WI, June 1994. With the assistance of Greg Turner & Jim Dembowski.

J. R. Westbury, M. Hashi, and M. J. Lindstrom. Differences among speakers in lingual articulation for American English /ɹ/. *Speech Communication*, 26(3):203–226, Nov. 1998.

J. Weston, A. Gammerman, M. O. Stitson, V. Vapnik, V. Vovk, and C. Watkins. Support vector density estimation. In Schölkopf et al. (1999a), chapter 18, pages 293–306.

J. Whittaker. *Graphical Models in Applied Multivariate Statistics.* Wiley Series in Probability and Mathematical Statistics. John Wiley & Sons, New York, London, Sydney, 1990.

P. Whittle. On principal components and least square methods of factor analysis. *Skand. Aktur. Tidskr.*, 36: 223–239, 1952.

J. Wiles, P. Bakker, A. Lynton, M. Norris, S. Parkinson, M. Staples, and A. Whiteside. Using bottlenecks in feedforward networks as a dimension reduction technique: An application to optimization tasks. *Neural Computation*, 8(6):1179–1183, Aug. 1996.

J. H. Wilkinson. *The Algebraic Eigenvalue Problem.* Oxford University Press, New York, Oxford, 1965.

P. M. Williams. Using neural networks to model conditional multivariate densities. *Neural Computation*, 8 (4):843–854, May 1996.

B. Willmore, P. A. Watters, and D. J. Tolhurst. A comparison of natural-image-based models of simple-cell coding. *Perception*, 29(9):1017–1040, Sept. 2000.

R. Wilson and M. Spann. A new approach to clustering. *Pattern Recognition*, 23(12):1413–1425, 1990.

J. H. Wolfe. Pattern clustering by multivariate mixture analysis. *Multivariate Behavioral Research*, 5:329–350, July 1970.

D. M. Wolpert and Z. Ghahramani. Computational principles of movement neuroscience. *Nat. Neurosci.*, 3 (Supp.):1212–1217, Nov. 2000.

D. M. Wolpert and M. Kawato. Multiple paired forward and inverse models for motor control. *Neural Networks*, 11(7–8):1317–1329, Oct. 1998.

A. A. Wrench. A multi-channel/multi-speaker articulatory database for continuous speech recognition research. In *Phonus*, volume 5, Saarbrücken, 2000. Institute of Phonetics, University of Saarland.

F. Xie and D. van Compernolle. Speech enhancement by spectral magnitude estimation —a unifying approach. *Speech Communication*, 19(2):89–104, Aug. 1996.

L. Xu, C. C. Cheung, and S. Amari. Learned parametric mixture based ICA algorithm. *Neurocomputing*, 22 (1–3):69–80, Nov. 1998.

E. Yamamoto, S. Nakamura, and K. Shikano. Lip movement synthesis from speech based on hidden Markov models. *Speech Communication*, 26(1–2):105–115, 1998.

H. H. Yang and S. Amari. Adaptive on-line learning algorithms for blind separation — maximum entropy and minimum mutual information. *Neural Computation*, 9(7):1457–1482, Oct. 1997.

H. Yehia and F. Itakura. A method to combine acoustic and morphological constraints in the speech production inverse problem. *Speech Communication*, 18(2):151–174, Apr. 1996.

H. Yehia, T. Kuratate, and E. Vatikiotis-Bateson. Using speech acoustics to drive facial motion. In Ohala et al. (1999), pages 631–634.

H. Yehia, P. Rubin, and E. Vatikiotis-Bateson. Quantitative association of vocal-tract and facial behavior. *Speech Communication*, 26(1–2):23–43, Oct. 1998.

G. Young. Maximum likelihood estimation and factor analysis. *Psychometrika*, 6:49–53, 1940.

S. J. Young. A review of large vocabulary continuous speech recognition. *IEEE Signal Processing Magazine*, 13(5):45–57, Sept. 1996.

K. Zhang, I. Ginzburg, B. L. McNaughton, and T. J. Sejnowski. Interpreting neuronal population activity by reconstruction: Unified framework with application to hippocampal place cells. *J. Neurophysiol.*, 79(2): 1017–1044, Feb. 1998.

R. D. Zhang and J.-G. Postaire. Convexity dependent morphological transformations for mode detection in cluster-analysis. *Pattern Recognition*, 27(1):135–148, 1994.

Y. Zhao and C. G. Atkeson. Implementing projection pursuit learning. *IEEE Trans. Neural Networks*, 7(2): 362–373, Mar. 1996.

I. Zlokarnik. Adding articulatory features to acoustic features for automatic speech recognition. *J. Acoustic Soc. Amer.*, 97(5):3246, May 1995a.

I. Zlokarnik. Articulatory kinematics from the standpoint of automatic speech recognition. *J. Acoustic Soc. Amer.*, 98(5):2930–2931, Nov. 1995b.