In: IEEE Trans. on Pattern Analysis and Machine Intelligence 22(11): 1318-1323 (Nov. 2000)
URL: http://www.dcs.shef.ac.uk/~miguel/papers/pami00.html

Mode-finding for mixtures of Gaussian distributions

Miguel Á. Carreira-Perpiñán*
Dept. of Computer Science, University of Sheffield, UK
M.Carreira@dcs.shef.ac.uk

August 4, 2000

Abstract

Gradient-quadratic and fixed-point iteration algorithms and appropriate values for their control parameters are derived for finding all modes of a Gaussian mixture, a problem with applications in clustering and regression. The significance of the modes found is quantified locally by Hessian-based error bars and globally by the entropy as sparseness measure.

Keywords: Gaussian mixtures, maximisation algorithms, mode finding, bump finding, error bars, sparseness.

1 Introduction

Gaussian mixtures [1] are ubiquitous probabilistic models for density estimation in machine learning applications due to several reasons: they benefit from the analytical tractability and asymptotic properties of the Gaussian distribution and naturally occur in many situations; they are a universal approximator for continuous densities [1, 2] and, in particular, can model multimodal distributions; many complex models result in a Gaussian mixture after some assumptions are made in order to obtain tractable models (e.g. Monte Carlo approximations to integrals, as in the generative topographic mapping (GTM) [3]); finally, a number of convenient algorithms for estimating the mixture parameters (means, covariance matrices and mixing proportions) exist, such as the traditional EM for maximum-likelihood [4] or more recent varieties, including Bayesian (and non-Bayesian) methods that can tune the number of components needed as well as the other parameters (e.g. [5]).

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 mentioned before, kernel density estimation [2], radial basis function networks [4], mixtures of probabilistic principal component analysers [6], mixtures of factor analysers [7], support vector machines for density estimation [8], models for conditional density estimation such as mixtures of experts [9] and mixture density networks [4], the emission distribution of hidden Markov models for automatic speech recognition and other applications [10] and, of course, the Gaussian mixture model itself. An extensive list of successful applications of Gaussian mixtures is given in [1].

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 [11] or in the recent product of experts model [12]. This may be a more efficient way to model high-dimensional data which simultaneously satisfies several low-dimensional constraints: each expert is associated to 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 visual scene should account for multimodal distributions so that multiple objects can be represented [13, 14]. In missing data reconstruction and inverse problems, multivalued mappings can be derived from the modes of the conditional distribution of the missing variables given the present ones [15]. Finding the modes of posterior distributions is also important in Bayesian analysis [16, 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 7.1).

Thus, the problem approached in this paper 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

*Currently at the Department of Neuroscience, Georgetown University Medical Center. Washington, DC 20007, USA. Email: miguel@giccs.georgetown.edu.

mixtures, so iterative numerical algorithms are necessary. Intuitively, it seems reasonable that the number of modes will be smaller or equal than 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 have formalised these notions in conjecture B.1 of [17], which also provides a partial proof. This conjecture suggests that 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. [18]).

The rest of the paper is organised as follows. Section 2 gives the equations for the moments, gradient and Hessian of the Gaussian mixture density with respect to the independent variables. Sections 3–4 describe algorithms for locating the modes. The significance of the modes thus obtained is quantified locally by computing error bars for each mode (section 5) and globally by measuring the sparseness of the mixture via the entropy (section 6). Section 7 summarises the paper, mentions some applications and compares mode finding with bump finding.

2 Moments, gradient and Hessian of a Gaussian mixture

Consider a mixture distribution [1] of $M > 1$ components in $\mathbb{R}^D$ for $D \geq 1$:

$$p(\mathbf{x}) \stackrel{\text{def}}{=} \sum_{m=1}^{M} p(m)p(\mathbf{x}|m) \stackrel{\text{def}}{=} \sum_{m=1}^{M} \pi_m p(\mathbf{x}|m) \qquad \forall \mathbf{x} \in \mathbb{R}^D \tag{1}$$

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 $\mathbf{x}|m \sim \mathcal{N}_D(\boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m)$, where $\boldsymbol{\mu}_m \stackrel{\text{def}}{=} \mathrm{E}_{p(\mathbf{x}|m)}\left\{\mathbf{x}\right\}$ and $\boldsymbol{\Sigma}_m \stackrel{\text{def}}{=} \mathrm{E}_{p(\mathbf{x}|m)}\left\{(\mathbf{x}-\boldsymbol{\mu}_m)(\mathbf{x}-\boldsymbol{\mu}_m)^T\right\} > 0$ are the mean vector and covariance matrix, respectively, of component m. Note that we write $p(\mathbf{x})$ and not $p(\mathbf{x}|\{\pi_m, \boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m\}_{m=1}^M)$ 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 and covariance are:

$$\text{Mean } \boldsymbol{\mu} \stackrel{\text{def}}{=} \mathrm{E}_{p(\mathbf{x})}\left\{\mathbf{x}\right\} = \sum_{m=1}^{M} \pi_m \boldsymbol{\mu}_m \tag{2}$$

$$\text{Covariance } \boldsymbol{\Sigma} \stackrel{\text{def}}{=} \mathrm{E}_{p(\mathbf{x})}\left\{(\mathbf{x}-\boldsymbol{\mu})(\mathbf{x}-\boldsymbol{\mu})^T\right\} = \sum_{m=1}^{M} \pi_m \left(\boldsymbol{\Sigma}_m + (\boldsymbol{\mu}_m-\boldsymbol{\mu})(\boldsymbol{\mu}_m-\boldsymbol{\mu})^T\right) \tag{3}$$

(these results are valid for any mixture, not necessarily of Gaussian distributions). The gradient and Hessian[1] of a Gaussian mixture with respect to the independent variables $\mathbf{x}$ (not with respect to the parameters $\{\pi_m, \boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m\}_{m=1}^M$) exist $\forall \mathbf{x} \in \mathbb{R}^D$ and are [17]:

$$\text{Gradient } \mathbf{g} \stackrel{\text{def}}{=} \nabla p(\mathbf{x}) = \sum_{m=1}^{M} p(\mathbf{x},m)\boldsymbol{\Sigma}_m^{-1}(\boldsymbol{\mu}_m - \mathbf{x}) \tag{4}$$

$$\text{Hessian } \mathbf{H} \stackrel{\text{def}}{=} (\nabla\nabla^T)p(\mathbf{x}) = \sum_{m=1}^{M} p(\mathbf{x},m)\boldsymbol{\Sigma}_m^{-1}\left((\boldsymbol{\mu}_m-\mathbf{x})(\boldsymbol{\mu}_m-\mathbf{x})^T - \boldsymbol{\Sigma}_m\right)\boldsymbol{\Sigma}_m^{-1}. \tag{5}$$

The gradient and Hessian of the log-density $L(\mathbf{x}) \stackrel{\text{def}}{=} \ln p(\mathbf{x})$ are related to those of p as follows:

$$\text{Gradient } \nabla L(\mathbf{x}) = \frac{1}{p}\mathbf{g} \tag{6}$$

$$\text{Hessian } (\nabla\nabla^T)L(\mathbf{x}) = -\frac{1}{p^2}\mathbf{g}\mathbf{g}^T + \frac{1}{p}\mathbf{H}. \tag{7}$$

Note from eq. (7) 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 paper, we will always implicitly refer to the gradient $\mathbf{g}$ or Hessian $\mathbf{H}$ of p, eqs. (4) and (5), rather than those of L, eqs. (6) and (7), unless otherwise noted.

[1] For clarity of notation, we omit the dependence on $\mathbf{x}$ of both the gradient and the Hessian, writing $\mathbf{g}$ and $\mathbf{H}$ where we should write $\mathbf{g}(\mathbf{x})$ and $\mathbf{H}(\mathbf{x})$.

3 Exhaustive mode search by a gradient-quadratic search

Consider a Gaussian mixture p with $M > 1$ components as in equation (1). Since the family of Gaussian mixtures is a density universal approximator, the landscape of p could be very complex. However, assuming that conjecture B.1 of [17] 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, i.e., starting from every vertex of the convex hull.

Due to the ease of calculation of the gradient (4) and the Hessian (5), it is straightforward to use quadratic maximisation (i.e., Newton's method) combined with gradient ascent [19]. Assuming we are at a point $\mathbf{x}_0$, let us expand $p(\mathbf{x})$ around $\mathbf{x}_0$ as a Taylor series to second order:

$$p(\mathbf{x}) \approx p(\mathbf{x}_0) + (\mathbf{x}-\mathbf{x}_0)^T \mathbf{g}(\mathbf{x}_0) + \frac{1}{2}(\mathbf{x}-\mathbf{x}_0)^T \mathbf{H}(\mathbf{x}_0)(\mathbf{x}-\mathbf{x}_0)$$

where $\mathbf{g}(\mathbf{x}_0)$ and $\mathbf{H}(\mathbf{x}_0)$ are the gradient and Hessian of p at $\mathbf{x}_0$, respectively. The zero-gradient point of the previous quadratic form is given by:

$$\nabla p(\mathbf{x}) = \mathbf{g}(\mathbf{x}_0) + \mathbf{H}(\mathbf{x}_0)(\mathbf{x}-\mathbf{x}_0) = \mathbf{0} \Longrightarrow \mathbf{x} = \mathbf{x}_0 - \mathbf{H}^{-1}(\mathbf{x}_0)\mathbf{g}(\mathbf{x}_0) \tag{8}$$

which jumps from $\mathbf{x}_0$ 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:

$$\mathbf{x} = \mathbf{x}_0 + s\mathbf{g}(\mathbf{x}_0) \tag{9}$$

where $s > 0$ is the step size. That is, we jump a distance $s \left\|\mathbf{g}(\mathbf{x}_0)\right\|$ 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 3.3.

Once found a point for which $\mathbf{g} = \mathbf{0}$, the Hessian (5) can confirm that the 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 $p(\mathbf{x})$ this should not be a problem (at least for a small dimensionality D). Section 3.3 discusses the control parameters for the gradient ascent. Fig. 1 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.
- 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.
- The gradient ascent should not suffer too much in higher dimensions because the search follows a one-dimensional path; and 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 techniques given in [17].
- Other optimisation strategies based on the gradient or the Hessian, such as the Levenberg-Marquardt algorithm [19], can also be easily constructed.

3.1 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 $p(\mathbf{x})$ and using those of $L(\mathbf{x}) = \ln p(\mathbf{x})$; although there is difference from other starting points, e.g. far from the convex hull, where $p(\mathbf{x})$ is very small. Fig. 1 illustrates this: in the top row, observe the slow search in points lying in areas of near-zero probability in the case of $p(\mathbf{x})$ 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 $L(\mathbf{x})$ in regions where $p(\mathbf{x})$ is small, as noted in section 2. This means that, for starting points in regions where $p(\mathbf{x})$ is small, quadratic steps can be taken more often and thus convergence is faster. However, the centroids are usually in areas of high $p(\mathbf{x})$ 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 $p(\mathbf{x})$ and $\ln p(\mathbf{x})$ 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—the usual case when the mixture is acting as a density approximator (as in kernel estimation).

3.2 Low-probability components

Gaussian mixtures are often applied to high-dimensional data. Due to computational difficulties and to the usual lack of sufficient training data (both issues arising from the curse of the dimensionality [2]), 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 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. $\theta = 0.01$).

A similar situation arises 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. For example, if $p(\mathbf{x}, \mathbf{y}) = \sum_{m=1}^{M} p(m)p(\mathbf{x}, \mathbf{y}|m)$ with $(\mathbf{x}, \mathbf{y}|m) \sim \mathcal{N}(\boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m)$ then $p(\mathbf{x}|\mathbf{y})$ is again a Gaussian mixture where the new m-th mixing proportion is proportional to $p(m)\exp(-\frac{1}{2}(\mathbf{y} - \boldsymbol{\mu}_{m,y})^T \boldsymbol{\Sigma}_{m,yy}^{-1}(\mathbf{y} - \boldsymbol{\mu}_{m,y}))$; $\boldsymbol{\mu}_{m,y}$ and $\boldsymbol{\Sigma}_{m,yy}$ are obtained by crossing out the columns and rows of variables $\mathbf{x}$ in $\boldsymbol{\mu}_m$ and $\boldsymbol{\Sigma}_m$, respectively. 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. Filtering out such low-probability components will accelerate considerably the mode search without missing any important mode.

3.3 Control parameters for the gradient-quadratic mode-finding algorithm

The following five parameters control the behaviour of the gradient-quadratic algorithm (all of them are nonnegative real numbers):

- `min_step`: minimum length of a gradient step that would climb up the hill monotonically, without overshooting. Our gradient ascent algorithm starts with a step size of several times `min_step` and halves it every time the new point has a worse probability.
- `min_grad`: minimum gradient norm, below which the gradient is considered numerically zero.
- `min_diff`: minimum absolute difference between two modes to be assimilated as one; its goal is to avoid different numerical realizations of the same mode.
- `max_eig`: maximum value of the algebraically largest eigenvalue of the Hessian for the Hessian to be considered negative definite; its goal is to rule out minima (for which $\|\mathbf{g}\| = 0$ but $\mathbf{H} > 0$) without missing any maximum.
- `max_it`: maximum number of iterations to be performed, to limit the computation time.

Heuristic values for these parameters, based on the smallest eigenvalue σ^2 of the covariance matrix of any of the components, are given in [17]. 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 [2]. 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 $\mathbf{x}_1$ and $\mathbf{x}_2$ differ in a small value δ in each component, so that $\|\mathbf{x}_1 - \mathbf{x}_2\| = \sqrt{D}\delta$. For high D, $\|\mathbf{x}_1 - \mathbf{x}_2\|$ will be large even though one would probably consider $\mathbf{x}_1$ and $\mathbf{x}_2$ as the same vector. However, if that difference was concentrated in 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 [19, pp. 398–399], so that in general we cannot get arbitrarily close to a scalar value μ: at best, our estimate x we will get to $|x - \mu| \sim \mu\sqrt{\epsilon_m}$, 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

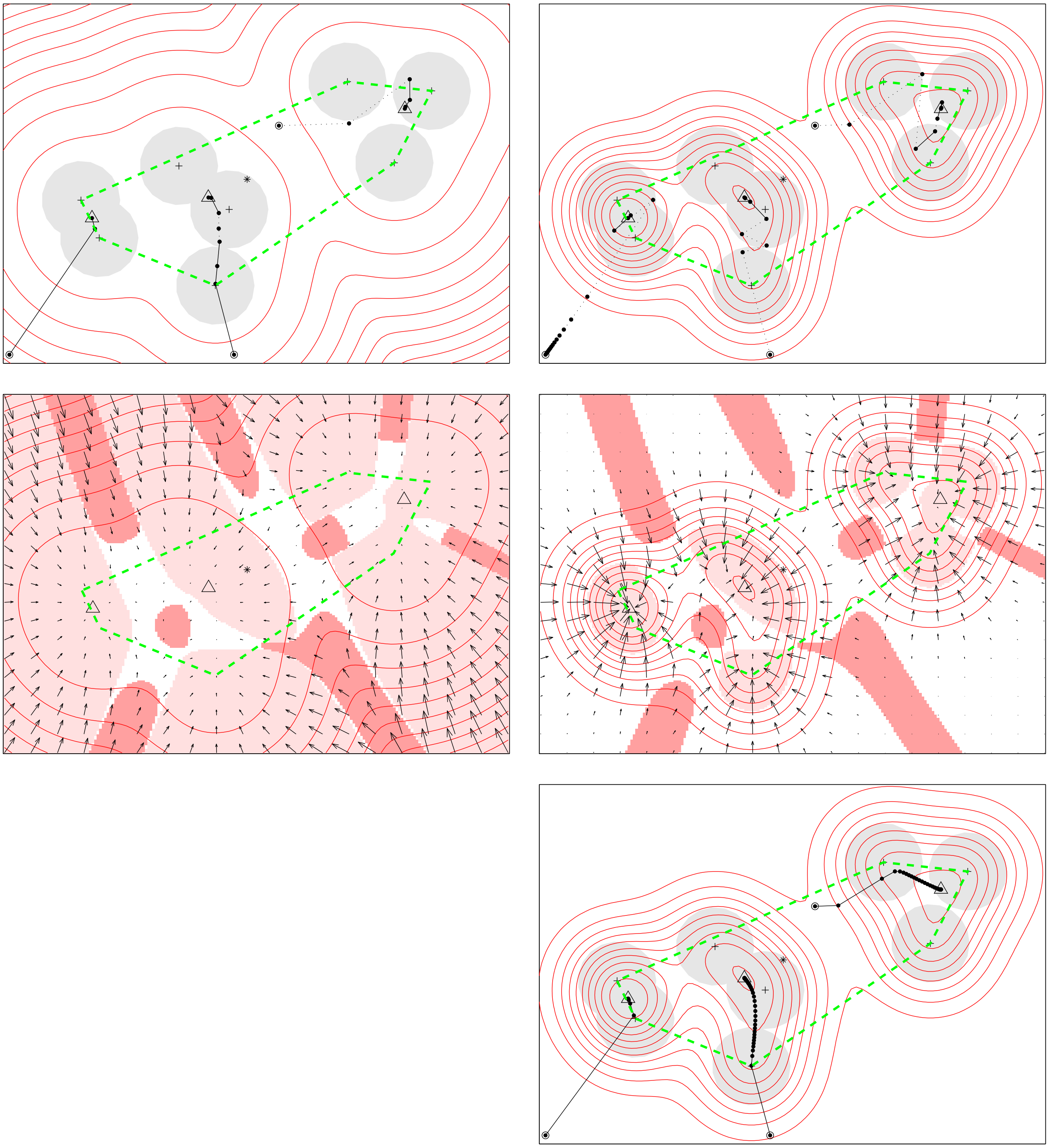

Figure 1: Example of mode searching in a two-dimensional Gaussian mixture. In this example, the mixing proportions are equal and the components are isotropic with equal covariance $\sigma^2\mathbf{I}_2$. The surface has 3 modes and various other features (saddle points, ridges, plateaux, etc.). The mixture modes are marked "$\triangle$" and the mixture mean "$*$". The dashed, thick-line polygon is the convex hull of the centroids. The left column shows the surface of $L(\mathbf{x}) = \ln p(\mathbf{x})$ and the right column the surface of $p(\mathbf{x})$. *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 "$\circ$") 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.

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.

4 Exhaustive mode search by a fixed-point search

Equating the gradient expression (4) to zero we obtain immediately a fixed-point iterative scheme:

$$\mathbf{g} = \sum_{m=1}^{M} p(\mathbf{x}, m)\boldsymbol{\Sigma}_m^{-1}(\boldsymbol{\mu}_m - \mathbf{x}) = \mathbf{0} \Longrightarrow \mathbf{x} = \mathbf{f}(\mathbf{x}) \text{ with } \mathbf{f}(\mathbf{x}) \stackrel{\text{def}}{=} \left(\sum_{m=1}^{M} p(m|\mathbf{x})\boldsymbol{\Sigma}_m^{-1}\right)^{-1} \sum_{m=1}^{M} p(m|\mathbf{x})\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\mu}_m \quad (10)$$

where we have used Bayes' theorem. Note that using the gradient of $L(\mathbf{x}) = \ln p(\mathbf{x})$ makes no difference here. A fixed point $\mathbf{x}$ of the mapping $\mathbf{f}$ verifies by definition $\mathbf{x} = \mathbf{f}(\mathbf{x})$. The fixed points of $\mathbf{f}$ are thus the stationary points of the mixture density p, including maxima, minima and saddle points. An iterative scheme $\mathbf{x}^{(n+1)} = \mathbf{f}(\mathbf{x}^{(n)})$ will converge to a fixed point of $\mathbf{f}$ under certain conditions, e.g. if $\mathbf{f}$ is a contractive mapping in an environment of the fixed point [20]. Unfortunately, the potential existence of several fixed points in the convex hull of $\{\boldsymbol{\mu}_m\}_{m=1}^{M}$ and the complexity of eq. (10) make a convergence analysis of the method difficult. However, in a number of experiments it has found exactly the same modes as the gradient-quadratic method. Thus, as in section 3, 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. (5) will determine whether the point found is actually a maximum.

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. 1 (bottom) the quick jump to the area of high probability and the slow convergence thereafter). The inverse matrix $\left(\sum_{m=1}^{M} p(m|\mathbf{x})\boldsymbol{\Sigma}_m^{-1}\right)^{-1}$ may be trivially computed in some cases (e.g. if all the components are diagonal). In the particular[2] case where $\boldsymbol{\Sigma}_m = \boldsymbol{\Sigma}$ for all $m = 1, \dots, M$, the fixed-point scheme reduces to the extremely simple form

$$\mathbf{x}^{(n+1)} = \sum_{m=1}^{M} p(m|\mathbf{x}^{(n)})\boldsymbol{\mu}_m,$$

i.e., the new point $\mathbf{x}^{(n+1)}$ is the conditional mean of the mixture under the current point $\mathbf{x}^{(n)}$. This is formally akin to EM algorithms for parameter estimation of mixture distributions [21], to clustering by deterministic annealing [22] and to algorithms for finding pre-images in kernel-based methods [23].

Whether this 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.

4.1 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. A new nonnegative tolerance parameter `tol` is defined 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 3.3. The control parameters `min_diff`, `max_eig` and `max_it` remain. Note that `min_diff` should be several times larger than `tol`.

5 Error bars for the modes

Here we derive error bars for an arbitrary mode of a Gaussian mixture, i.e., a confidence hyperrectangle (a hyperellipse is also possible) containing the mode and with a probability under the mixture distribution of value P fixed in advance. 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. 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

[2]Important models fall in this case, such as Gaussian kernel density estimation [2] or the generative topographic mapping [3].

certain covariance matrix; then we compute symmetric error bars for this normal distribution. This results in the following algorithm (see [17] for details). Choose a confidence level $0 < P < 1$ and compute $\rho = \sqrt{2}\,\arg\operatorname{erf}(P^{1/D})$, where "erf" is the error function. 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:

- If the mixture is unimodal, then decompose $\boldsymbol{\Sigma} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^T$ with $\mathbf{U}$ orthogonal and $\boldsymbol{\Lambda} > 0$ diagonal (singular value decomposition of a symmetric matrix). 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 we use local curvature info:
  - If $\mathbf{H}$ is the Hessian of $p(\mathbf{x})$, 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 $L(\mathbf{x}) = \ln p(\mathbf{x})$, 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 $\frac{2\rho}{\sqrt{\lambda_1}}, \dots, \frac{2\rho}{\sqrt{\lambda_D}}$.

The natural error bars are aligned with the principal axes of $\boldsymbol{\Sigma}$ or $\mathbf{H}$, which in general will not coincide with the original axes of the $x_1, \dots, x_D$ variables. Observe that, while the directions of the bars obtained from $L(\mathbf{x}) = \ln p(\mathbf{x})$ coincide with those from $p(\mathbf{x})$ always, the lengths are different in general (except when $p(\mathbf{x})$ is Gaussian).

Since we are using only second order local information, valid in a small neighbourhood around the mode, sometimes we can get poor estimates of the error bars. Also, 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. Finally, note that 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 [2].

6 Quantifying the sparseness of a Gaussian mixture

If the Gaussian mixture under consideration represents the conditional distribution of variables $\mathbf{x}$ given the values of other variables $\mathbf{y}$, the modes could be taken as possible values of the mapping $\mathbf{y} \to \mathbf{x}$ provided that the distribution is sparse [15]. That is, a sparse distribution—sharply peaked around the modes, with most of the probability mass concentrated around a small region around each mode—would roughly correspond to a functional relationship (perhaps multivalued) while a flat distribution would indicate independence. While the error bars locally characterise the peak widths, we can globally characterise the degree of sparseness of a distribution $p(\mathbf{x})$ by its differential entropy $h(p) \overset{\text{def}}{=} \mathrm{E}\{-\ln p\}$: the lower the entropy (compared to the 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), the sparser the density. There is no analytic expression for the entropy of a Gaussian mixture, but upper and lower bounds, as well as approximated values, are available [17].

7 Summary and applications

We have presented algorithms to find all the modes of a given Gaussian mixture based on the intuitive conjecture that the number of modes is upper bounded by the number of components and that the modes are contained in the convex hull of the component centroids. While only partial proofs of this conjecture are available (see [17] and references therein), no counterexample has been found in a number of simulations. All algorithms have been extensively tested for the case of spherical components in simulated mixtures, in an inverse kinematics problem for a robot arm (unpublished results) and in the context of a missing data reconstruction algorithm applied to a speech inverse problem, the acoustic-to-articulatory mapping [15], which requires finding all the modes of a Gaussian mixture of about 1000 components in over 60 dimensions for every speech frame in an utterance.

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 [24].

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 [15]. 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 $\mathbf{x} \rightarrow \mathbf{y}$ is defined as the modes of the conditional 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 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 $p(\mathbf{x}) \rightarrow 0$ when $\|\mathbf{x}\| \rightarrow \infty$.

7.1 Bump-finding rather than mode-finding

If we want to pick representative points of an arbitrary density $p(\mathbf{x})$, not necessarily a Gaussian mixture, using a mode as a reconstructed point is not appropriate in general because the optimal value (in the L_2 sense) is the mean and because the modes are not robust statistics. This suggests that, when the conditional distribution is multimodal, we should look for *bumps*[3] associated to the correct values and take the means of these bumps as reconstructed values instead of the modes. If these bumps are symmetrical then the result would 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 $p(\mathbf{x})$ as a mixture $p(\mathbf{x}) = \sum_{k=1}^{K} p(k)p(\mathbf{x}|k)$ where $p(\mathbf{x}|k)$ is the density associated to the k-th bump. This density should be localised in the space of $\mathbf{x}$ but can be asymmetrical. If $p(\mathbf{x})$ is modelled by a mixture of Gaussians (as is the case in this paper) then the previous decomposition 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, replacing it by a grouping and averaging procedure—probably much faster. And again, in the situations of section 3.2, low-probability components from the Gaussian mixture may be discarded to accelerate the procedure.

These ideas operate exclusively with the functional form of a Gaussian mixture as starting point, as do our mode-finding algorithms. Bump-finding methods that work directly with a data sample exist, such as algorithms that partition the space of the $\mathbf{x}$ variables into boxes where $p(\mathbf{x})$ (or some arbitrary function of $\mathbf{x}$) takes a large value on the average compared to the average value over the entire space, e.g. PRIM [25]; or some nonparametric and parametric clustering methods, e.g. scale-space clustering [18, 26] or methods based on morphological transformations [27].

8 Internet files

A Matlab implementation of the mode-finding algorithms and error bars computation and a technical report including mathematical details and pseudocode [17] are available in the WWW at `http://www.dcs.shef.ac.uk/~miguel/papers/cs-99-03.html`.

Acknowledgements

We thank Steve Renals for useful conversations and for comments on the original manuscript.

References

[1] D. M. Titterington, A. F. M. Smith, and U. E. Makov, *Statistical Analysis of Finite Mixture Distributions*. Wiley Series in Probability and Mathematical Statistics, New York, London, Sydney: John Wiley & Sons, 1985.

[2] D. W. Scott, *Multivariate Density Estimation. Theory, Practice, and Visualization*. Wiley Series in Probability and Mathematical Statistics, New York, London, Sydney: John Wiley & Sons, 1992.

[3] Bumps of a density function $p(\mathbf{x})$ are usually defined as continuous regions where $p''(\mathbf{x}) < 0$, while modes are points where $p'(\mathbf{x}) = 0$ and $p''(\mathbf{x}) < 0$ [2]. However, the literature has sometimes used interchangeably the terms "bumps," "modes" and "peaks."

[3] C. M. Bishop, M. Svensén, and C. K. I. Williams, "GTM: The generative topographic mapping," *Neural Computation*, vol. 10, pp. 215–234, Jan. 1998.

[4] C. M. Bishop, *Neural Networks for Pattern Recognition.* New York, Oxford: Oxford University Press, 1995.

[5] S. J. Roberts, D. Husmeier, I. Rezek, and W. Penny, "Bayesian approaches to Gaussian mixture modeling," *IEEE Trans. on Pattern Anal. and Machine Intel.*, vol. 20, no. 11, pp. 1133–1142, 1998.

[6] M. E. Tipping and C. M. Bishop, "Mixtures of probabilistic principal component analyzers," *Neural Computation*, vol. 11, pp. 443–482, Feb. 1999.

[7] G. E. Hinton, P. Dayan, and M. Revow, "Modeling the manifolds of images of handwritten digits," *IEEE Trans. Neural Networks*, vol. 8, pp. 65–74, Jan. 1997.

[8] V. N. Vapnik and S. Mukherjee, "Support vector method for multivariate density estimation," in Solla *et al.* [28], pp. 659–665.

[9] R. A. Jacobs, M. I. Jordan, S. J. Nowlan, and G. E. Hinton, "Adaptive mixtures of local experts," *Neural Computation*, vol. 3, no. 1, pp. 79–87, 1991.

[10] L. Rabiner and B.-H. Juang, *Fundamentals of Speech Recognition.* Signal Processing Series, Englewood Cliffs, N.J.: Prentice-Hall, 1993.

[11] C. Genest and J. V. Zidek, "Combining probability distributions: A critique and an annotated bibliography," *Statistical Science*, vol. 1, pp. 114–135 (with discussion, pp. 135–148), Feb. 1986.

[12] G. E. Hinton, "Products of experts," in *Proc. of the Ninth Int. Conf. on Artificial Neural Networks (ICANN99)*, (Edinburgh, UK), pp. 1–6, The Institution of Electrical Engineers, Sept. 7–10 1999.

[13] B. Moghaddam and A. Pentland, "Probabilistic visual learning for object representation," *IEEE Trans. on Pattern Anal. and Machine Intel.*, vol. 19, pp. 696–710, July 1997.

[14] M. Isard and A. Blake, "CONDENSATION — conditional density propagation for visual tracking," *Int. J. Computer Vision*, vol. 29, no. 1, pp. 5–28, 1998.

[15] M. Á. Carreira-Perpiñán, "Reconstruction of sequential data with probabilistic models and continuity constraints," in Solla *et al.* [28], pp. 414–420.

[16] A. Gelman, J. B. Carlin, H. S. Stern, and D. B. Rubin, *Bayesian Data Analysis.* Texts in Statistical Science, London, New York: Chapman & Hall, 1995.

[17] M. Á. Carreira-Perpiñán, "Mode-finding for mixtures of Gaussian distributions," Tech. Rep. CS–99–03, Dept. of Computer Science, University of Sheffield, UK, Mar. 1999 (revised August 4, 2000). Available online at `http://www.dcs.shef.ac.uk/~miguel/papers/cs-99-03.html`.

[18] R. Wilson and M. Spann, "A new approach to clustering," *Pattern Recognition*, vol. 23, no. 12, pp. 1413–1425, 1990.

[19] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes in C: The Art of Scientific Computing.* Cambridge, U.K.: Cambridge University Press, second ed., 1992.

[20] E. Isaacson and H. B. Keller, *Analysis of Numerical Methods.* New York, London, Sydney: John Wiley & Sons, 1966.

[21] 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*, vol. 39, no. 1, pp. 1–38, 1977.

[22] K. Rose, "Deterministic annealing for clustering, compression, classification, regression, and related optimization problems," *Proc. IEEE*, vol. 86, pp. 2210–2239, Nov. 1998.

[23] 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*, vol. 10, pp. 1000–1017, Sept. 1999.

[24] A. Pisani, "A nonparametric and scale-independent method for cluster-analysis. 1. the univariate case," *Monthly Notices of the Royal Astronomical Society*, vol. 265, pp. 706–726, Dec. 1993.

[25] J. H. Friedman and N. I. Fisher, "Bump hunting in high-dimensional data," *Statistics and Computing*, vol. 9, pp. 123–143 (with discussion, pp. 143–162), Apr. 1999.

[26] S. J. Roberts, "Parametric and non-parametric unsupervised cluster analysis," *Pattern Recognition*, vol. 30, pp. 261–272, Feb. 1997.

[27] R. D. Zhang and J.-G. Postaire, "Convexity dependent morphological transformations for mode detection in cluster-analysis," *Pattern Recognition*, vol. 27, no. 1, pp. 135–148, 1994.

[28] S. A. Solla, T. K. Leen, and K.-R. Müller, eds., *Advances in Neural Information Processing Systems*, vol. 12, MIT Press, Cambridge, MA, 2000.