Partial-Hessian Strategies for Fast Learning of Nonlinear Embeddings

Max Vladymyrov and **Miguel Á. Carreira-Perpiñán**

Electrical Engineering and Computer Science
University of California, Merced

`https://eecs.ucmerced.edu`

June 29, 2012

Introduction

We focus on graph-based dimensionality reduction techniques:

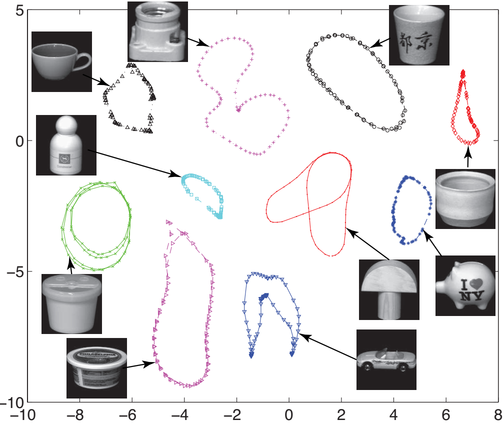
- Input is a (sparse) affinity matrix.
- Objective function is a minimization over the location of the latent points.
- Examples:
  - Spectral methods: Laplacian Eigenmaps (LE), LLE;
    - ✓ have a closed-form solution;
    - ✗ results are often not satisfactory.
  - Nonlinear methods: SNE, s-SNE, *t*-SNE, elastic embedding (EE);
    - ✓ produce good quality embedding;
    - ✗ notoriously slow to train, limited to small data sets.

One reason for slow training is inefficient optimization algorithms that take many iterations and move very slowly towards a solution.

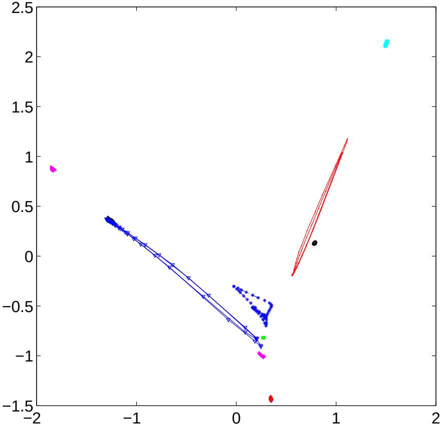
COIL-20 Dataset

Rotations of 10 objects every 5°; input is greyscale images of 128 × 128.

Elastic Embedding



Laplacian Eigenmaps



Teaser

We are proposing a new training algorithm that:

- generalizes over multiple algorithms (s-SNE, t-SNE, EE);
- fast (1-2 orders of magnitude compared to current techniques);
- allows deep, inexpencive steps;
- scalable to larger datasets;
- intuitive and easy to implement.

General Embedding Formulation (Carreira-Perpiñán 2010)

For $\mathbf{Y} = (\mathbf{y}_1, \dots, \mathbf{y}_N) \in \mathcal{R}^{D\times N}$ matrix of high-dimensional points and $\mathbf{X} = (\mathbf{x}_1, \dots, \mathbf{x}_N) \in \mathcal{R}^{d\times N}$ matrix of low-dimensional points, define an objective function:

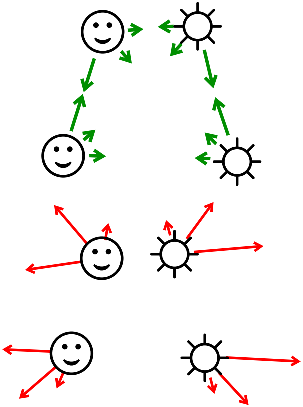
$$E(\mathbf{X}, \lambda) = E^+(\mathbf{X}) + \lambda E^-(\mathbf{X}) \qquad \lambda \geq 0$$

E^+ is the *attractive term*:

- often quadratic,
- minimal with coincident points;

E^- is the *repulsive term*:

- often very nonlinear,
- minimal with points separated infinitely.



Optimal embeddings balance both forces.

Example: SNE (Hinton & Roweis 2003)

Define P_n and Q_n as distributions for each data point over the neighbors in high- and low-dimensional spaces respectively:

$$p_{nm} = \frac{\exp(-\frac{\|\mathbf{y}_n - \mathbf{y}_m\|^2}{\sigma^2})}{\sum_{k=1,k\neq n}^{N} \exp(-\frac{\|\mathbf{y}_n - \mathbf{y}_m\|^2}{\sigma^2})}; \quad q_{nm} = \frac{\exp(-\|\mathbf{x}_n - \mathbf{x}_m\|^2)}{\sum_{k=1,k\neq n}^{N} \exp(-\|\mathbf{x}_n - \mathbf{x}_m\|^2)}$$

The goal is to position points $\mathbf{X}$ such that P_n matches the Q_n for every n:

$$\begin{aligned}
E_{\mathrm{SNE}}(\mathbf{X}) &= \sum_{n=1}^{N} \mathrm{D}\left(P_n \| Q_n\right) \\
&= \sum_{n,m=1}^{N} p_{nm} \log \frac{p_{nm}}{q_{nm}} = -\sum_{n,m=1}^{N} p_{nm} \log q_{nm} + C \\
&= \sum_{n,m=1}^{N} p_{nm} \|\mathbf{x}_n - \mathbf{x}_m\|^2 + \sum_{n=1}^{N} \log \sum_{m\neq n} \exp\left(-\|\mathbf{x}_n - \mathbf{x}_m\|^2\right) + C \\
&= E^{+}(\mathbf{X}) + \lambda E^{-}(\mathbf{X}) \qquad \text{(In this formulation } \lambda = 1\text{)}
\end{aligned}$$

General Embedding Formulation: Other Special Cases

	$E^+(\mathbf{X})$	$E^-(\mathbf{X})$
SNE: (Hinton&Roweis,'03)	$\sum_{n,m=1}^{N} p_{nm} \left\|\mathbf{x}_n - \mathbf{x}_m\right\|^2$	$\sum_{n=1}^{N} \log \sum_{m=1}^{N} e^{-\left\|\mathbf{x}_n - \mathbf{x}_m\right\|^2}$
s-SNE: (Cook at al,'07)	$\sum_{n,m=1}^{N} p_{nm} \left\|\mathbf{x}_n - \mathbf{x}_m\right\|^2$	$\log \sum_{n,m=1}^{N} e^{-\left\|\mathbf{x}_n - \mathbf{x}_m\right\|^2}$
t-SNE: (van der Maaten & Hinton,'08)	$\sum_{n,m=1}^{N} p_{nm} \log\left(1 + \left\|\mathbf{x}_n - \mathbf{x}_m\right\|^2\right)$	$\log \sum_{n,m=1}^{N} \left(1 + \left\|\mathbf{x}_n - \mathbf{x}_m\right\|^2\right)^{-1}$
EE: (Carreira-Perpiñán,'10)	$\sum_{n,m=1}^{N} w_{nm}^{+} \left\|\mathbf{x}_n - \mathbf{x}_m\right\|^2$	$\sum_{n,m=1}^{N} w_{nm}^{-} e^{-\left\|\mathbf{x}_n - \mathbf{x}_m\right\|^2}$
LE & LLE: (Belkin & Niyogi,'03) (Roweis & Saul,'00)	$\sum_{n,m=1}^{N} w_{nm}^{+} \left\|\mathbf{x}_n - \mathbf{x}_m\right\|^2$ s.t. constraints	0

w_{nm}^{+} and w_{nm}^{-} are affinity matrices elements

Optimization Strategy

Look for a search direction $\mathbf{p}_k$ at iteration k as a solution of a linear system $\mathbf{B}_k\mathbf{p}_k = -\mathbf{g}_k$, where $\mathbf{g}_k$ is the current gradient and $\mathbf{B}_k$ is a partial Hessian matrix.

$$\mathbf{B}_k = \mathbf{I} \text{ (grad. descent) } \xrightarrow[\text{faster convergence rate}]{\text{more Hessian information}} \mathbf{B}_k = \nabla^2 E \text{ (Newton's method)}$$

We want $\mathbf{B}_k$:

- contain as much information about the Hessian as possible;
- positive definite (pd);
- fast to solve the linear system and scale up to larger N.

After $\mathbf{p}_k$ is obtained, a line search algorithm finds the step size α for the next iteration $\mathbf{X}_{k+1} = \mathbf{X}_k + \alpha\mathbf{p}_k$. We used backtracking line search.

Structure of the Hessian of the Generalized Embedding

Given a symmetric matrix of weights $\mathbf{W}$, we can always define its degree matrix $\mathbf{D} = \operatorname{diag}\left(\sum_{n=1}^{N} w_{nm}\right)$ and its graph Laplacian $\mathbf{L} = \mathbf{D} - \mathbf{W}$.
$\mathbf{L}$ is positive semi-definite (psd) when entries of $\mathbf{W}$ are non-negative.

The $Nd \times Nd$ Hessian can be written in terms of certain graph Laplacians:

$\nabla^2 E = 4\mathbf{L} \otimes \mathbf{I}_d$	$\mathbf{L} = \mathbf{L}^+ - \lambda \mathbf{L}^-$; $\nabla^2 E^+(\mathbf{X}) = \mathbf{L}^+ \otimes \mathbf{I}_d$ $\mathbf{L}^+$ is psd and data-independent for Gaussian kernel.
$+8\mathbf{L}^{xx}$	data-dependent, overall not definite, but has psd diagonal blocks.[†]
$-16\lambda \operatorname{vec}(\mathbf{X}\mathbf{L}^q) \operatorname{vec}(\mathbf{X}\mathbf{L}^q)^T$	always negative definite.[†]

[†] exact expressions for $\mathbf{L}^{xx}$ and $\mathbf{L}^q$ are in the paper.

Thus, there are several choices for psd parts of the Hessian:

- The best choice depends on the problem.
- We focus in particular on the one that does generally well.

The Spectral Direction (definition)

$$\nabla^2 E = 4\underset{\substack{\downarrow \\ \mathbf{L}^+ - \lambda \mathbf{L}^-}}{\mathbf{L}} \otimes \mathbf{I}_d + 8\mathbf{L}^{xx} - 16\lambda\,\mathrm{vec}\left(\mathbf{X}\mathbf{L}^q\right)\mathrm{vec}\left(\mathbf{X}\mathbf{L}^q\right)^T$$

$\mathbf{B}_k = 4\mathbf{L}^+ \otimes \mathbf{I}_d$ is a convenient Hessian approximation:

- equal to the Hessian of the spectral methods: $\nabla^2 E^+(\mathbf{X})$;
- always psd $\Rightarrow$ global convergence under mild assumptions;
- block-diagonal and has d blocks of $N \times N$ graph Laplacian $4\mathbf{L}^+$;
- **constant** for Gaussian kernel. For other kernels we can fix it at some $\mathbf{X}$;
- "bends" the gradient of the nonlinear E using the curvature of the spectral E^+;

The Spectral Direction (computation)

We need to solve a linear system $\mathbf{B}_k \mathbf{p}_k = \mathbf{g}_k$ efficiently for every iteration (naively $\mathcal{O}(N^3 d)$).

- Cache the (also sparse) Cholesky factor of $\mathbf{L}^+$ in the first iteration. Now, there are just two triangular systems for each iteration.
- For scalability, we can make $\mathbf{W}^+$ even more sparse than it was with a κ-NN graph ($\kappa \in [1, N]$ is a user parameter). This affects only the runtime, convergence is still guaranteed.
- $\mathbf{B}_k$ is psd $\Rightarrow$ add small constant μ to the diagonal elements.

	Cost per iteration
Objective function	$\mathcal{O}(N^2 d)$
Gradient	$\mathcal{O}(N^2 d)$
Spectral direction	$\mathcal{O}(N \kappa d)$

This strategy adds almost no overhead when compared to the objective function and the gradient computation.

The Spectral Direction (pseudocode)

```
SpectralDirection($\mathbf{X}_0$, $\mathbf{W}^+$, $\kappa$)
(optional) Further sparsify $\mathbf{W}^+$ with $\kappa$-NN graph
$\mathbf{L}^+ \leftarrow \mathbf{D}^+ - \mathbf{W}^+$                    Compute graph Laplacian $\mathcal{O}(N)$
$\mathbf{R} \leftarrow \texttt{chol}(\mathbf{L}^+ + \mu\mathbf{I})$      compute Cholesky decomposition $\mathcal{O}(N^2\kappa)$
$k \leftarrow 1$
repeat
    Compute $E_k$ and $\mathbf{g}_k$                                     Objective function and the gradient $\mathcal{O}(N^2 d)$
    $\mathbf{p}_k \leftarrow -\mathbf{R}^{-T}(\mathbf{R}^{-1}\mathbf{g}_k)$   Solve two triangular systems $\mathcal{O}(N\kappa d)$
    $\alpha \leftarrow$ backtracking line search
    $\mathbf{X}_k \leftarrow \mathbf{X}_{k-1} + \alpha\mathbf{p}_k$
    $k \leftarrow k + 1$
until stop
return $\mathbf{X}$
```

Experimental Evaluation: Methods Compared

- Gradient descent (GD), (Hinton&Roweis,'03) $\mathbf{B}_k = \mathbf{I}$

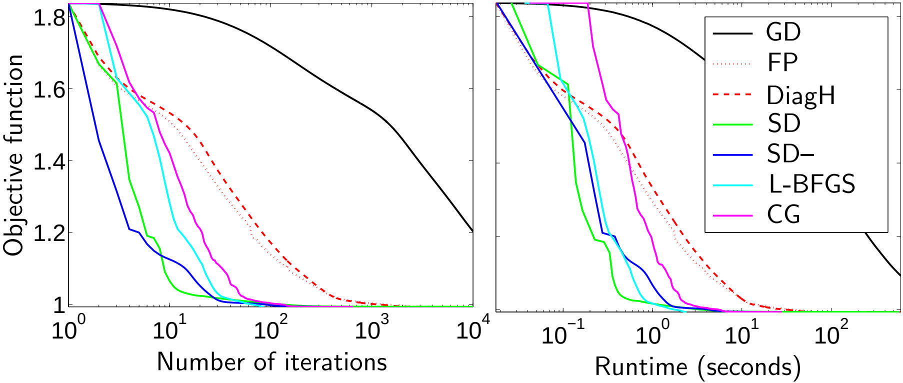
- Diagonal methods:
  - fixed-point iterations (FP), (Carreira-Perpiñán,'10) $\mathbf{B}_k = 4\mathbf{D}^+ \otimes \mathbf{I}_d$
  - the diagonal of the Hessian (DiagH); $\mathbf{B}_k = 4\mathbf{D}^+ \otimes \mathbf{I}_d + 8\lambda\mathbf{D}^{xx}$

- Our methods:
  - spectral direction (SD); $\mathbf{B}_k = 4\mathbf{L}^+ \otimes \mathbf{I}_d$
  - partial Hessian SD–, solve linear system with conjugate gradient; $\mathbf{B}_k = 4\mathbf{L}^+ \otimes \mathbf{I}_d + 8\lambda\mathbf{L}^{xx}_{i*,i*}$

- Standard large-scale methods:
  - nonlinear Conjugate Gradient (CG);
  - L-BFGS.

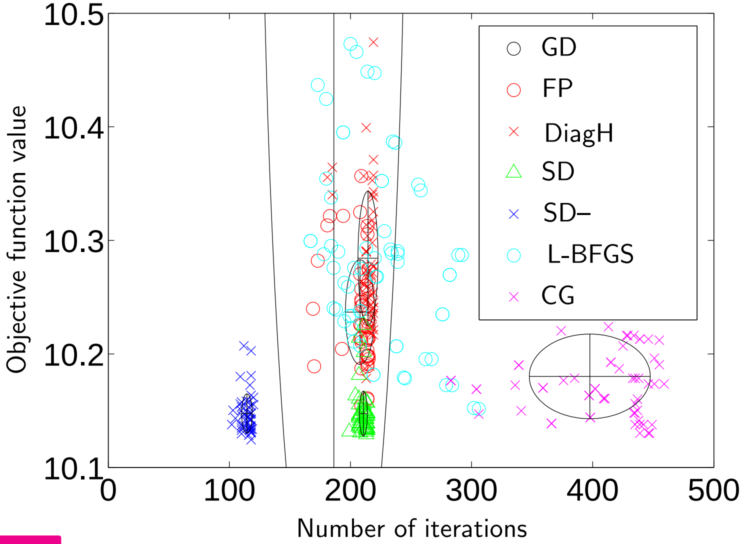
COIL-20. Convergence to the same minimum, EE

Initialize $\mathbf{X}_0$ close enough to $\mathbf{X}_\infty$ so that all methods have the same initial and final points.



COIL-20. Convergence from random initial **X**, s-SNE

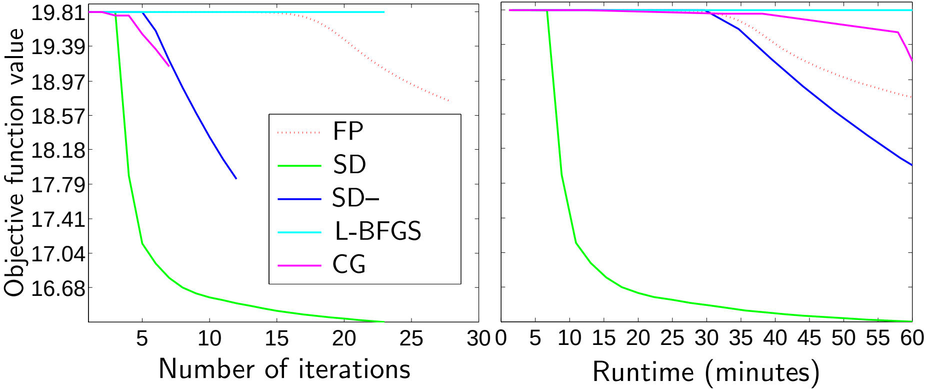
Run the algorithms 50 times for 20 seconds each with different initialization.



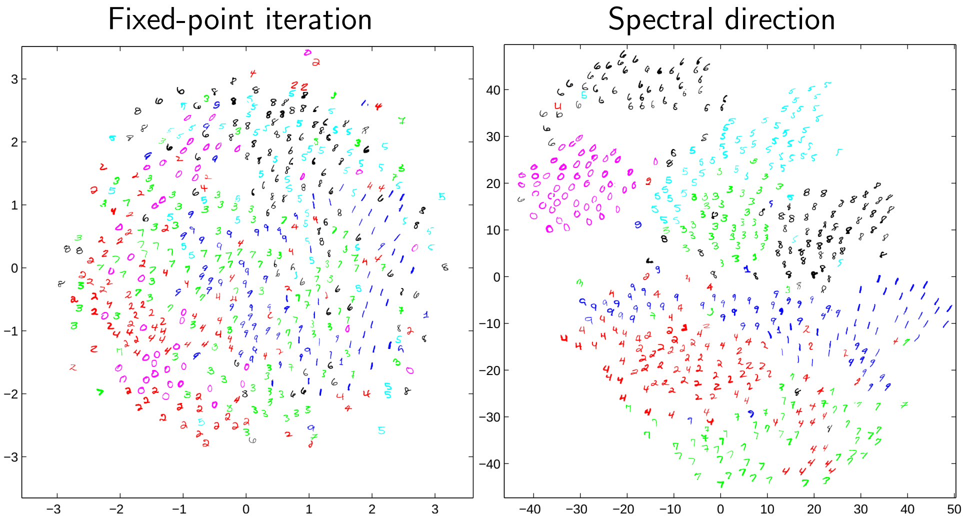
Animation

MNIST. *t*-SNE

$N = 20\,000$ images of handwritten digits (each a 28×28 pixel grayscale image, $D = 784$). 1 hour of optimization.



MNIST. Embedding after 1 hour of *t*-SNE optimization



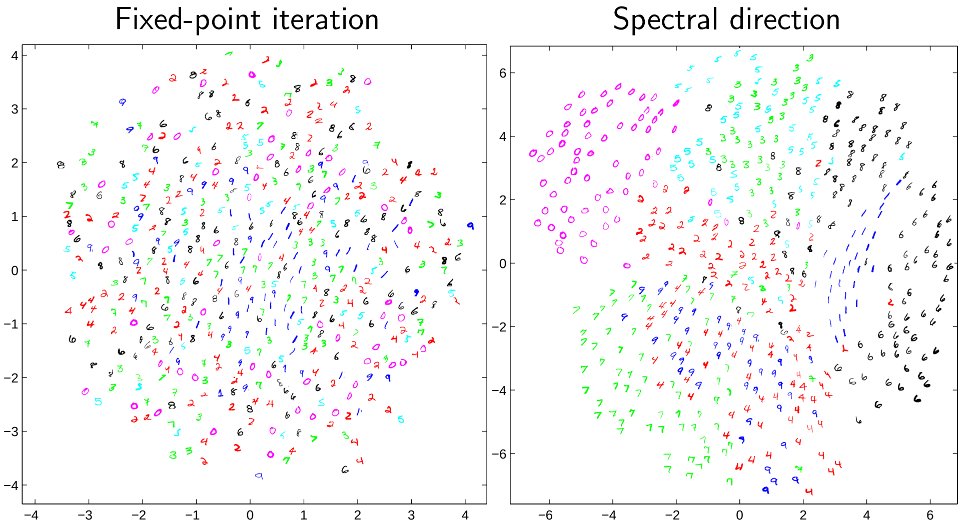
Animation

Conclusions

- We presented a common framework for many well-known dimensionality reduction techniques.
- We showed the role of graph Laplacians in the Hessian and derived several partial Hessian optimization strategies.
- We presented the **spectral direction**: a new simple, generic and scalable optimization strategy that runs one to two orders of magnitude faster compared to traditional methods.
- The evaluation of E and ∇E remains the bottleneck ($\mathcal{O}(N^2 d)$) that can be addressed in the future works (e.g. with Fast Multipole Methods).
- Matlab code (very soon): `http://eecs.ucmerced.edu/`.

Partially supported by NSF CAREER award IIS–0754089.

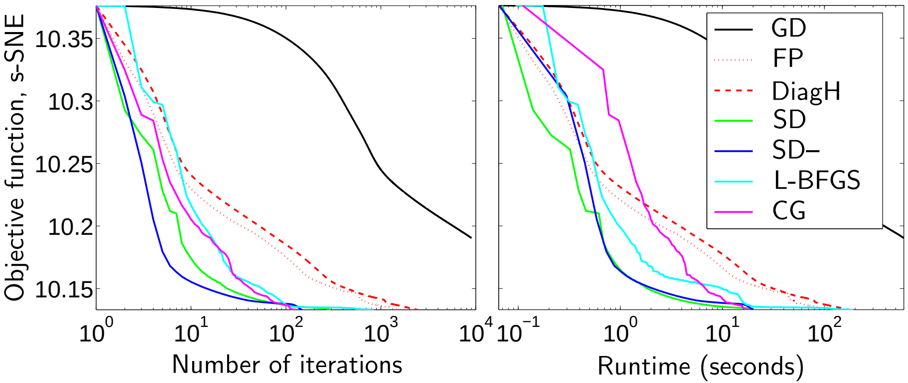
MNIST. Embedding after 20 min of EE optimization



Animation

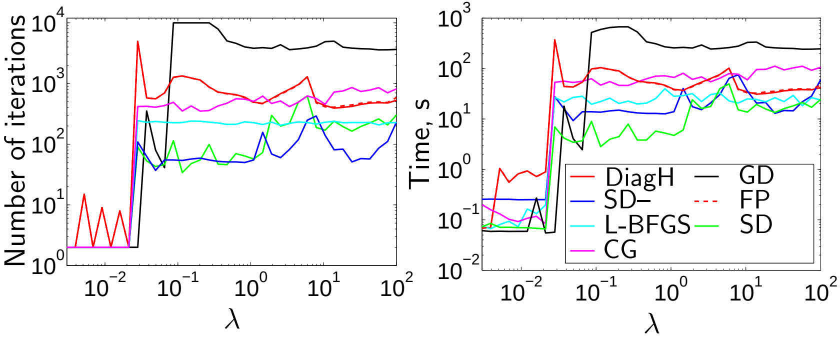
COIL-20. Convergence to the same minimum, s-SNE

We initialized $\mathbf{X}_0$ close enough to $\mathbf{X}_\infty$ so that all methods have the same initial and final points.



COIL-20: Homotopy optimization for EE

Start with small λ where E is convex and follow the path of minima to desired λ by minimizing over $\mathbf{X}$ as λ increases. We used 50 log-spaced values from 10^{-4} to 10^2.



Animation