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 $128 \times 128$. 

Elastic Embedding

Laplacian Eigenmaps
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, inexpensive steps;
- scalable to larger datasets;
- intuitive and easy to implement.
General Embedding Formulation (Carreira-Perpiñán 2010)

For $Y = (y_1, \ldots, y_N) \in \mathcal{R}^{D \times N}$ matrix of high-dimensional points and $X = (x_1, \ldots, x_N) \in \mathcal{R}^{d \times N}$ matrix of low-dimensional points, define an objective function:

$$E(X, \lambda) = E^+(X) + \lambda E^-(X) \quad \lambda \geq 0$$

$E^+$ 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:

$$p_{nm} = \frac{\exp(-\frac{||y_n - y_m||^2}{\sigma^2})}{\sum_{k=1, k\neq n}^{N} \exp(-\frac{||y_n - y_m||^2}{\sigma^2})}, \quad q_{nm} = \frac{\exp(-\frac{||x_n - x_m||^2}{\sigma^2})}{\sum_{k=1, k\neq n}^{N} \exp(-\frac{||x_n - x_m||^2}{\sigma^2})}$$

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

$$E_{\text{SNE}}(X) = \sum_{n=1}^{N} D(P_n || Q_n)$$

$$= \sum_{n,m=1}^{N} p_{nm} \log \frac{p_{nm}}{q_{nm}} = -\sum_{n,m=1}^{N} p_{nm} \log q_{nm} + C$$

$$= \sum_{n,m=1}^{N} p_{nm} ||x_n - x_m||^2 + \sum_{n=1}^{N} \log \sum_{m\neq n} \exp (-||x_n - x_m||^2) + C$$

$$= E^+(X) + \lambda E^-(X) \quad \text{(In this formulation } \lambda = 1)$$
### General Embedding Formulation: Other Special Cases

<table>
<thead>
<tr>
<th></th>
<th>( E^+(X) )</th>
<th>( E^-(X) )</th>
</tr>
</thead>
</table>
| **SNE:**  | \[
    \sum_{n,m=1}^N p_{nm} \|x_n - x_m\|^2
    \] | \[
    \sum_{n=1}^N \log \sum_{m=1}^N e^{-\|x_n-x_m\|^2}
    \] |
| \*(Hinton&Roweis,'03)* |  |  |
| **s-SNE:**| \[
    \sum_{n,m=1}^N p_{nm} \|x_n - x_m\|^2
    \] | \[
    \log \sum_{n,m=1} e^{-\|x_n-x_m\|^2}
    \] |
| \*(Cook et al,'07)* |  |  |
| **t-SNE:**| \[
    \sum_{n,m=1}^N p_{nm} \log (1 + \|x_n - x_m\|^2)
    \] | \[
    \log \sum_{n,m=1}^N (1 + \|x_n - x_m\|^2)^{-1}
    \] |
| \*(van der Maaten & Hinton, '08)* |  |  |
| **EE:**   | \[
    \sum_{n,m=1}^N w_{nm}^+ \|x_n - x_m\|^2
    \] | \[
    \sum_{n,m=1}^N w_{nm}^- e^{-\|x_n-x_m\|^2}
    \] |
| \*(Carreira-Perpiñán,'10)* |  |  |
| **LE & LLE:**| \[
    \sum_{n,m=1}^N w_{nm}^+ \|x_n - x_m\|^2
    \] | 0 |
| \*(Belkin & Niyogi,'03)  \*(Roweis & Saul,'00)* |  |  |

\( w_{nm}^+ \) 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.

$$
\mathbf{B}_k = \mathbf{I} \text{ (grad. descent) } \stackrel{\text{more Hessian information}}{\longrightarrow} \mathbf{B}_k = \nabla^2 E \text{ (Newton’s method)}
$$

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 $\alpha$ 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 $\mathbf{D} = \text{diag} \left( \sum_{n=1}^{N} w_{nm} \right)$ 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:

$$\nabla^2 E = 4 \mathbf{L} \otimes I_d$$

$$+ 8 \mathbf{L}^{xx}$$

$$- 16 \lambda \text{vec}(\mathbf{XL}^q) \text{vec}(\mathbf{XL}^q)^T$$

$L = L^+ - \lambda L^-$; $\nabla^2 E^+(\mathbf{X}) = L^+ \otimes I_d$

$L^+$ is psd and data-independent for Gaussian kernel.

data-dependent, overall not definite, but has psd diagonal blocks.†

always negative definite.†

†exact expressions for $L^{xx}$ and $L^q$ 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)

\[ \nabla^2 E = 4L \otimes I_d + 8L^x - 16\lambda \text{vec}(XL^q) \text{vec}(XL^q)^T \]

\[ L^+ - \lambda L^- \]

\( B_k = 4L^+ \otimes I_d \) is a convenient Hessian approximation:

- equal to the Hessian of the spectral methods: \( \nabla^2 E^+(X) \);
- always psd \( \Rightarrow \) global convergence under mild assumptions;
- block-diagonal and has \( d \) blocks of \( N \times N \) graph Laplacian \( 4L^+ \);
- constant for Gaussian kernel. For other kernels we can fix it at some \( X \);
- “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 \( O(N^3 d) \)).

- 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 \( \kappa \)-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 \( \mu \) to the diagonal elements.

<table>
<thead>
<tr>
<th></th>
<th>Cost per iteration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Objective function</td>
<td>( O(N^2 d) )</td>
</tr>
<tr>
<td>Gradient</td>
<td>( O(N^2 d) )</td>
</tr>
<tr>
<td><strong>Spectral direction</strong></td>
<td>( O(N \kappa d) )</td>
</tr>
</tbody>
</table>

This strategy adds almost no overhead when compared to the objective function and the gradient computation.
The Spectral Direction (pseudocode)

\textbf{SpectralDirection}(X_0, W^+, \kappa)

(optional) Further sparsify $W^+$ with $\kappa$-NN graph

$L^+ \leftarrow D^+ - W^+$ \hspace{1cm} \text{Compute graph Laplacian $O(N)$}

$R \leftarrow \text{chol}(L^+ + \mu I)$ \hspace{1cm} \text{compute Cholesky decomposition $O(N^2\kappa)$}

$k \leftarrow 1$

\textbf{repeat}

Compute $E_k$ and $g_k$

$p_k \leftarrow -R^{-T}(R^{-1}g_k)$ \hspace{1cm} \text{Objective function and the gradient $O(N^2d)$}

$\alpha \leftarrow$ backtracking line search \hspace{1cm} \text{Solve two triangular systems $O(N\kappa d)$}

$X_k \leftarrow X_{k-1} + \alpha p_k$

$k \leftarrow k + 1$

\textbf{until} stop

\textbf{return} $X$
Experimental Evaluation: Methods Compared

• Gradient descent (GD),
  (Hinton&Roweis,'03)
  \[ B_k = I \]

• Diagonal methods:
  ▶ fixed-point iterations (FP),
    (Carreira-Perpiñán,'10)
  ▶ the diagonal of the Hessian (DiagH);
  \[ B_k = 4D^+ \otimes I_d \]

• Our methods:
  ▶ spectral direction (SD);
  \[ B_k = 4L^+ \otimes I_d \]
  ▶ partial Hessian SD–, solve linear system with conjugate gradient;
  \[ B_k = 4L^+ \otimes I_d + 8\lambda L_{xx}^{i*,i*} \]

• Standard large-scale methods:
  ▶ nonlinear Conjugate Gradient (CG);
  ▶ L-BFGS.
COIL-20. Convergence to the same minimum, EE

Initialize $X_0$ close enough to $X_\infty$ so that all methods have the same initial and final points.

![Graph showing convergence and runtime](image)
COIL-20. Convergence from random initial $\mathbf{X}$, s-SNE

Run the algorithms 50 times for 20 seconds each with different initialization.
MNIST. \(t\)-SNE

\[ N = 20,000 \text{ images of handwritten digits (each a } 28 \times 28 \text{ pixel grayscale image, } D = 784) \]. 1 hour of optimization.
MNIST. Embedding after 1 hour of $t$-SNE optimization

Fixed-point iteration

Spectral direction

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 $\nabla E$ remains the bottleneck ($O(N^2d)$) that can be addressed in the future works (e.g. with Fast Multipole Methods).

Partially supported by NSF CAREER award IIS–0754089.
MNIST. Embedding after 20 min of EE optimization

Fixed-point iteration

Spectral direction

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

We initialized $X_0$ close enough to $X_\infty$ so that all methods have the same initial and final points.
Start with small $\lambda$ where $E$ is convex and follow the path of minima to desired $\lambda$ by minimizing over $X$ as $\lambda$ increases. We used 50 log-spaced values from $10^{-4}$ to $10^2$. 

**Animation**