PDEs and Graph Based Learning

Summer School on Random Structures in Optimizations and Related Applications

Lecture 3: t-SNE

Instructor: Jeff Calder (jcalder@umn.edu)

Web: http://www-users.math.umn.edu/~jwcalder

Lecture Notes: http://www-users.math.umn.edu/~jwcalder/5467S21

Embeddings of high dimensional data

High dimensional data is hard to visualize and work with. Embeddings to low dimensional spaces help us visualize data and improve the performance of data analysis algorithms.

- PCA (linear dimension reduction)
- Spectral embedding (today)
- t-Distributed Stochastic Neighbor Embedding (t-SNE) (also today)

The key is to embed the data while still preserving important structures.

Spectral embeddings

Let $v_1, v_2, v_3, \ldots$ be the normalized eigenvectors of L, in order of increasing eigenvalues $0 = \lambda_1 \leq \lambda_2 \leq \cdots$. The spectral embedding corresponding to L is the map $\Phi : I_m \to \mathbb{R}^k$ (recall $I_m = \{1, 2, \ldots, m\}$ are the indices of our datapoints) given by

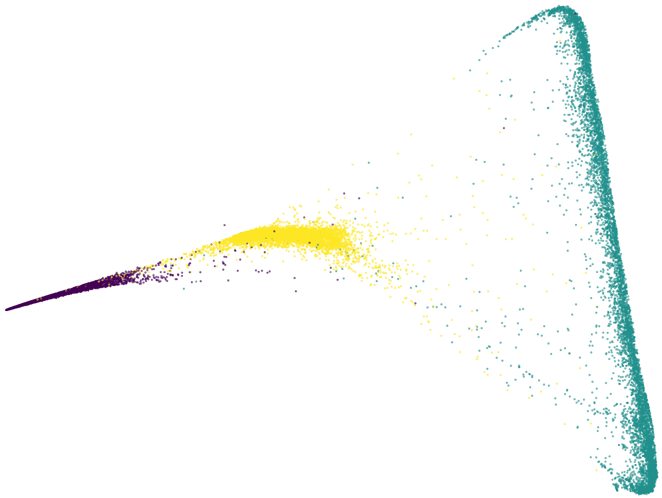
$$\Phi(i) = (v_1(i), v_2(i), \ldots, v_k(i)). \tag{1}$$

Since the first eigenvector v_1 is the trivial constant eigenvector, it is also common to omit this to obtain the embedding

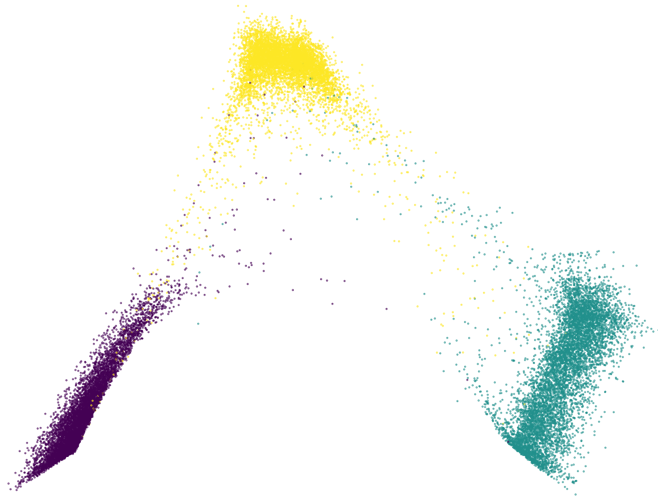
$$\Phi(i) = (v_2(i), v_3(i), \ldots, v_{k+1}(i)).$$

There are other normalizations of the graph Laplacian that are commonly used, such as the symmetric normalization $L = D^{-1/2}(D - W)D^{-1/2}$, and the spectral embedding for a normalized Laplacian is defined analagously.

Spectral embedding of MNIST



(a) Unnormalized



(b) Normalized

Figure 1: Example of spectral embeddings in the plane $k = 2$ of the 0, 1, and 2 digits of the MNIST dataset using the unnormalized $L = D - W$ and symmetric normalized $L = D^{-1/2}(D - W)D^{-1/2}$ graph Laplacians.

t-SNE

The t-Stochastic Neighbor Embedding (t-SNE) tries to find embedded points whose pairwise similarities match as closely as possible the given weight matrix W for the graph.

From the weight matrix W, t-SNE constructs a probability weight matrix

$$P = \frac{1}{2m}(D^{-1}W + W^T D^{-1}), \tag{2}$$

where D is the diagonal matrix of degrees $d(i) = \sum_{j=1}^{m} W(i,j)$. We say *probability* since all entries of P sum to one, i.e., $\mathbf{1}^T P \mathbf{1} = 1$.

t-SNE

t-SNE aims to find embedded points $y_1, y_2, \ldots, y_m \in \mathbb{R}^k$, where usually $k = 2$ or $k = 3$, so that the similarity between y_i and y_j matches $P(i,j)$ as closely as possible. The similarity matrix for the y_i, denoted Q, is the $m \times m$ matrix defined by

$$Q(i,j) = \frac{(1 + \|y_i - y_j\|^2)^{-1}}{\sum_{\ell \neq s}(1 + \|y_s - y_\ell\|^2)^{-1}}. \tag{3}$$

The discrepancy between P and Q is measured with the Kullback-Leibler divergence

$$E(y_1, y_2, \ldots, y_k) = D(P\|Q) := \sum_{i \neq j} P(i,j) \log\left(\frac{P(i,j)}{Q(i,j)}\right). \tag{4}$$

t-SNE finds the embedded points y_i by minimizing E with gradient descent.

Why Kullbck-Leibler?

$$E(y_1, y_2, \ldots, y_k) = D(P\|Q) := \sum_{i \neq j} P(i,j) \log \left(\frac{P(i,j)}{Q(i,j)} \right).$$

- When $P(i,j) \gg 0$, forces $Q(i,j) \sim P(i,j)$; i.e., preserve local structure.
- When $P(i,j) \sim 0$ we don't care what $Q(i,j)$ does; i.e., allow global structure to change.
- We cannot preserve all information in a dimension reduction.

Gradient descent

The gradient of the Kullback-Leibler divergence

$$E(y_1, y_2, \ldots, y_k) = D(P\|Q) := \sum_{i\neq j} P(i,j) \log\left(\frac{P(i,j)}{Q(i,j)}\right)$$

in the variable y_i is given by

$$\nabla_{y_i} E = \underbrace{4Z \sum_{j:j\neq i} P(i,j)Q(i,j)(y_i - y_j)}_{\text{Attraction}} - \underbrace{4Z \sum_{j:j\neq i} Q(i,j)^2 (y_i - y_j)}_{\text{Repulsion}}.$$

where $Z = \sum_{i\neq j}(1 + \|y_i - y_j\|^2)^{-1}$.

Gradient descent is

$$y_i^{k+1} = y^k - h\nabla_{y_i} E(y_1^k, y_2^k, \ldots, y_m^k),$$

where h is the time-step.

t-SNE embedding of MNIST

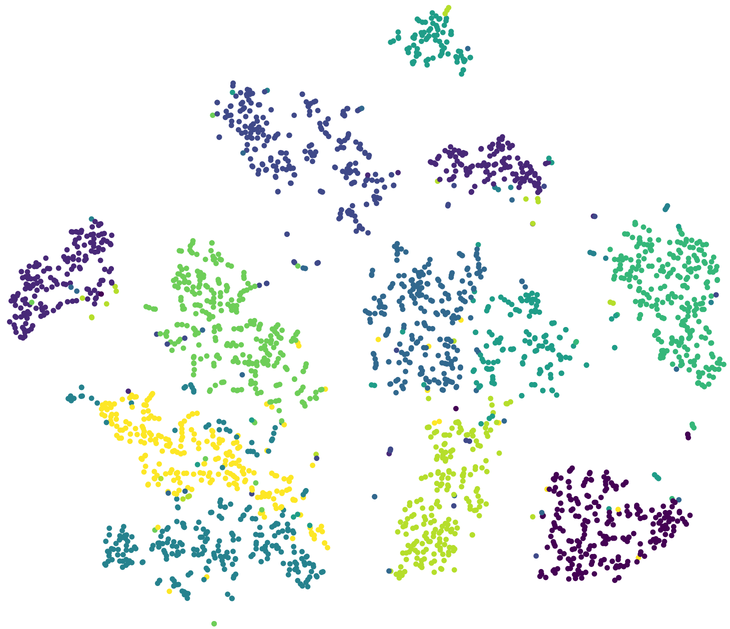


Figure 2: A t-SNE embedding of 2500 images from the MNIST dataset, with colors corresponding to the digit labels of each image.

$$E = \sum_{i \neq j} P(i,j) \log \left(\frac{P(i,j)}{Q(i,j)} \right)$$

We may as well write

$$E = - \sum_{i \neq j} P(i,j) \log \left(Q(i,j) \right)$$

<u>Recall</u>

$$Q(i,j) = \frac{\left(1 + |y_i - y_j|^2\right)^{-1}}{\sum_{l \neq s} \left(1 + |y_l - y_s|^2\right)^{-1}}$$

$$E = \underbrace{\sum_{i \neq j} P(i,j) \log \left(1 + |y_i - y_j|^2 \right)}_{A}$$

$$+\underbrace{\left(\sum_{i\neq j} p(i,j)\right)}_{=1} \underbrace{\log\left(\sum_{l\neq s}\left(1+|y_l-y_s|^2\right)^{-1}\right)}_{B}$$

$$\nabla_{y_l} A = \sum_{i\neq j} p(i,j)\, \nabla_{y_l} \log\left(1+|y_i-y_j|^2\right)$$

$$= \sum_{i\neq j} p(i,j)\left(1+|y_i-y_j|^2\right)^{-1} \nabla_{y_l} |y_i-y_j|^2$$

$$= 2\sum_{i\neq j} p(i,j)\left(1+|y_i-y_j|^2\right)^{-1}(y_i-y_j)(\delta_{il}-\delta_{jl})$$

If $i=l$, $\nabla_{y_l}|y_i-y_j|^2 = 2(y_i-y_j)$

If $j=\ell$, $\nabla_{y_\ell} |y_i - y_j|^2 = 2(y_j - y_i)$

$$= -2(y_i - y_j)$$

$$\delta_{ij} = \begin{cases} 1, & i=j \\ 0, & i \neq j \end{cases}$$

$\nabla_{y_\ell} A$

$$= 2 \sum_{i \neq j} P(i,j)(1 + |y_i - y_j|^2)^{-1} (y_i - y_j)(\delta_{i\ell} - \delta_{j\ell})$$

$$= 2 \sum_{j \neq \ell} P(\ell,j)(1 + |y_\ell - y_j|^2)^{-1} (y_\ell - y_j)$$

$$-2 \sum_{j \neq \ell} P(j,\ell)(1 + |y_j - y_\ell|^2)^{-1} (y_j - y_\ell)$$

$$= 4 \sum_{j \neq l} P(l,j) \left(1 + |y_j - y_l|^2\right)^{-1} (y_l - y_j)$$

Define $$Z = \sum_{i \neq j} \left(1 + |y_i - y_j|^2\right)^{-1}$$

so that $$Q(i,j) = \frac{\left(1 + |y_i - y_j|^2\right)^{-1}}{Z}$$

$$= \underbrace{4 Z \sum_{j \neq l} P(l,j)\, Q(l,j)\, (y_l - y_j)}_{\text{Attraction term.}}$$

$$B = \log\left(\sum_{i\neq j}(1+|y_i-y_j|^2)^{-1}\right)$$

$$D_{y_l} B = D_{y_l} \log\left(\sum_{i\neq j}(1+|y_i-y_j|^2)^{-1}\right)$$

$$= Z^{-1} D_{y_l} \sum_{i\neq j}(1+|y_i-y_j|^2)^{-1}$$

$$= -Z^{-1} \sum_{i\neq j} \underbrace{(1+|y_i-y_j|^2)^{-2}}_{Q(i,j)^2 Z^2} D_{y_l} |y_i-y_j|^2$$

$$= -Z \sum_{i\neq j} Q(i,j)^2\, 2(y_i-y_j)(d_{il}-d_{jl})$$

$$= -Z \sum_{i \neq j} Q(i,j)^2 \, 2 (y_i - y_j) \delta_{il}$$

$$+ Z \sum_{i \neq j} Q(i,j)^2 \, 2 (y_i - y_j) \delta_{lj}$$

$$= -2Z \sum_{j \neq l} Q(l,j)^2 (y_l - y_j)$$

$$+ 2Z \sum_{i \neq l} Q(i,l) (y_i - y_l)$$

$$= - \underbrace{4 Z \sum_{j \neq l} Q(l,j)^2 (y_l - y_j)}_{\text{Repulsion term.}}$$

Early exaggeration

Gradient descent for t-SNE is *very* slow. To speed it up, *early exaggeration* is used, which amplifies the attraction forces for the first few hunderd iterations:

$$\nabla_{y_i} E = 4Z\alpha \sum_{j:j\neq i} P(i,j)Q(i,j)(y_i - y_j) - 4Z \sum_{j:j\neq i} Q(i,j)^2(y_i - y_j),$$

The parameter α is the amplification factor, often $\alpha \approx 10$.

Early exaggeration

(a) After early exaggeration (b) Final embedding

Figure 3: An example of the t-SNE embedding after the early exaggeration phase and the final embedded, for a small version of MNIST with only 500 images from the digits 0, 1, 2, and 3.

Perplexity

The construction of the weight matrix W is important for the performance of t-SNE. The *perplexity* construction has the form

$$W(i,j) = \exp\left(-\frac{\|x_i - x_j\|^2}{2\sigma_i^2}\right),$$

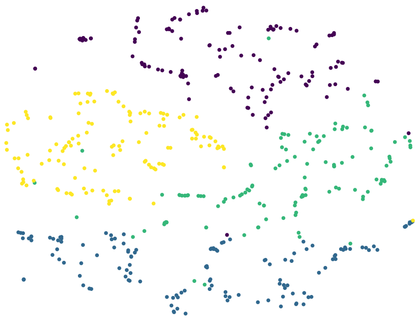
where σ_i is tuned independently for each x_i depending on a specified *perplexity* level (usually in the range 5 to 50).

The perplexity of the i^{th} row of $W(i,j)$ is $2^{H(i)}$, where

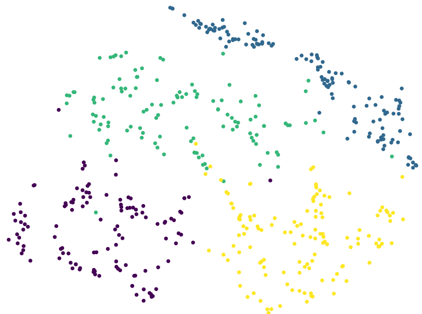
$$H(i) = -\sum_{j=1}^{m} p(j) \log p(j), \quad p(j) = \frac{W(i,j)}{\sum_{k=1}^{m} W(i,k)}.$$

The value of σ_i is determined so that the perplexity $2^{H(i)}$ equals a desired user-specified value.

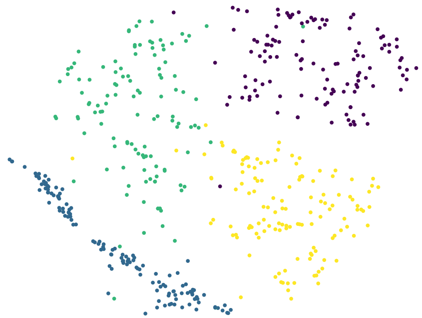
MNIST



Perplexity 5

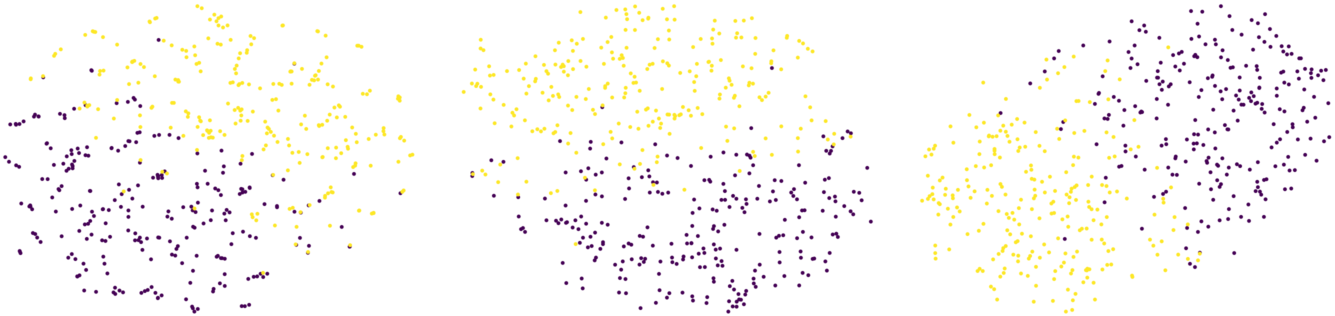


Perplexity 30



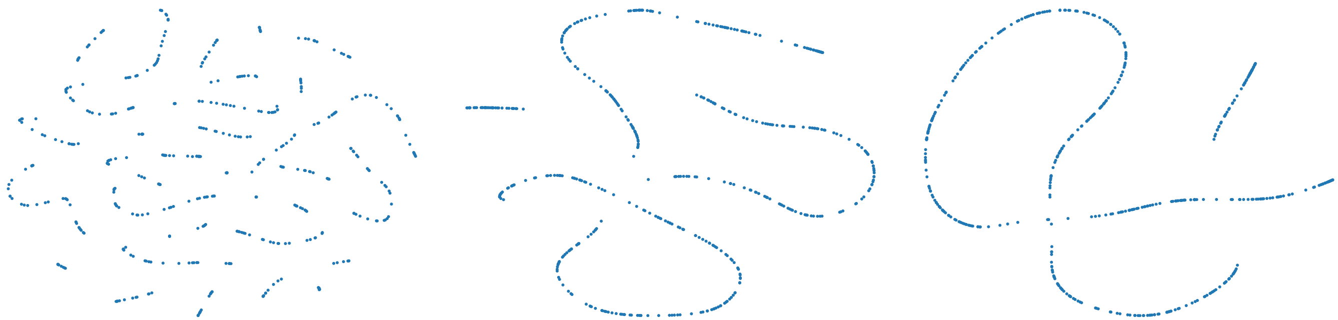
Perplexity 50.

Gaussian mixture in 10 dimensions



Perplexity 5 Perplexity 30 Perplexity 50.

Parabolic curve in 5 dimensions



Perplexity 5 Perplexity 30 Perplexity 50.

Graph-based embeddings (.ipynb)