Lecture 7: Nonlinear Dimensionality Reduction

Shukai Gong

1 Manifold Learning

1.1 Multi-dimensional Scaling

Metric MDS

Given a set of data $\{\boldsymbol{x_n}\}_{n=1}^N$, we can compute a distance matrix

$$\boldsymbol{D} = [d_{ij}] \in \mathbb{R}^{N\times N}, \; d_{ij} = d(\boldsymbol{x_i}, \boldsymbol{x_j})$$

Metric MDS aims at finding low-dimensional latent representation $\{\boldsymbol{z}_n\}_{n=1}^N$ to keep isometry as much as possible via

$$\min_{\{\boldsymbol{z}_n\}_{n=1}^N} \text{Stress}_d(\{\boldsymbol{z}_n\}_{n=1}^N) = \left(\min_{\{\boldsymbol{z}_n\}_{n=1}^N} \sum_{i\neq j}(d_{ij} - \|\boldsymbol{z}_i - \boldsymbol{z}_j\|_p)^2\right)^{\frac{1}{2}}$$

where $p = 1, 2$ in general.

[Note]

- There's no explicit expression for $\{\boldsymbol{x_n}\} \to \{\boldsymbol{z_n}\}$
- There isn't unique solution for $\{\boldsymbol{z_n}\}$. For example, if we take $p = 2$ and

$$\{\boldsymbol{z}_n^*\} = \arg\min_{\{\boldsymbol{z}_n\}_{n=1}^N} \left(\sum_{i\neq j}(d_{ij} - \|\boldsymbol{z}_i - \boldsymbol{z}_j\|_2)^2\right)^{\frac{1}{2}}$$

  and U as any unitary matrix ($\boldsymbol{U}^*\boldsymbol{U} = \boldsymbol{I}$), then $\{\boldsymbol{U}\boldsymbol{z}_n^*\}$ is also a solution since $\|\boldsymbol{U}\boldsymbol{z}_i - \boldsymbol{U}\boldsymbol{z}_j\|_2 = \|\boldsymbol{z}_i - \boldsymbol{z}_j\|_2$.

Classic MDS

Classic MDS is a special case of Metric MDS where $d_{ij} = \|\boldsymbol{x_i} - \boldsymbol{x_j}\|_2$ is Euclidean. We replace our optimization goal from $\min \text{Stress}_d(\{\boldsymbol{z}_n\}_{n=1}^N)$, **which minimizes the difference between pairwise distances in the original space and the latent space**, to

$$\min \text{Strain}_d(\{\boldsymbol{z}_n\}_{n=1}^N) = \min_{\{\boldsymbol{z}_n\}_{n=1}^N} \left(\frac{\sum_{i,j=1}^N (k_{ij} - \boldsymbol{z}_i^\top \boldsymbol{z}_j)^2}{\sum_{i,j=1}^N k_{ij}^2}\right)^{\frac{1}{2}}$$

which minimizes the difference between inner product in the original space and the latent space.

Denote our dataset as $\boldsymbol{X} \in \mathbb{R}^{N\times D}$. Here the Gram Matrix is defined as $\boldsymbol{K} = [k_{ij}] = -\dfrac{1}{2}\boldsymbol{C}(\boldsymbol{D}\odot\boldsymbol{D})\boldsymbol{C}$ with **centering matrix** $\boldsymbol{C} = \boldsymbol{I}_N - \dfrac{1}{N}\boldsymbol{1}_{N\times N}$. The low-dimension embedding $\boldsymbol{Z}^*$ is derived first by performing EVD on $\boldsymbol{K} := \boldsymbol{V}\boldsymbol{\Delta}\boldsymbol{V}^\top$, then

$$\boldsymbol{Z}^* = \boldsymbol{V}_L\boldsymbol{\Delta}_L^{\frac{1}{2}}$$

Denote $\tilde{\boldsymbol{X}} = \boldsymbol{C}\boldsymbol{X}$, then $\boldsymbol{K} = \tilde{\boldsymbol{X}}\tilde{\boldsymbol{X}}^\top$ (See Appendix for derivation). Back to our optimization goal of

$$\begin{aligned}
\min_{\{\boldsymbol{z}_n\}_{n=1}^N} \text{Strain}_d(\{\boldsymbol{z}_n\}_{n=1}^N) &= \min_{\{\boldsymbol{z}_n\}_{n=1}^N} \left(\frac{\sum_{i,j=1}^N (k_{ij} - \boldsymbol{z}_i^\top \boldsymbol{z}_j)^2}{\sum_{i,j=1}^N k_{ij}^2} \right)^{\frac{1}{2}} = \min_{\{\boldsymbol{z}_n\}_{n=1}^N} \left(\sum_{i,j=1}^N (k_{ij} - \boldsymbol{z}_i^\top \boldsymbol{z}_j)^2 \right)^{\frac{1}{2}} \\
&= \min_{\boldsymbol{Z}} \|\boldsymbol{K} - \boldsymbol{Z}\boldsymbol{Z}^\top\|_F = \min_{\boldsymbol{Z}} \|\boldsymbol{K} - \boldsymbol{Z}\boldsymbol{Z}^\top\|_F^2 \\
&= \min_{\boldsymbol{Z}} \text{tr}[(\boldsymbol{K} - \boldsymbol{Z}\boldsymbol{Z}^\top)^\top(\boldsymbol{K} - \boldsymbol{Z}\boldsymbol{Z}^\top)] = \min_{\boldsymbol{Z}} \text{tr}[(\boldsymbol{K} - \boldsymbol{Z}\boldsymbol{Z}^\top)^2]
\end{aligned}$$

Performing EVD on $\boldsymbol{K}$ and $\boldsymbol{Z}\boldsymbol{Z}^\top$, we have

$$\boldsymbol{K} = \boldsymbol{V}\boldsymbol{\Delta}\boldsymbol{V}^\top,\ \boldsymbol{Z}\boldsymbol{Z}^\top = \boldsymbol{Q}\boldsymbol{\Psi}\boldsymbol{Q}^\top$$

and then

$$\begin{aligned}
\|\boldsymbol{K} - \boldsymbol{Z}\boldsymbol{Z}^\top\|_F^2 &= \text{tr}[(\boldsymbol{V}\boldsymbol{\Delta}\boldsymbol{V}^\top - \boldsymbol{Q}\boldsymbol{\Psi}\boldsymbol{Q}^\top)]^2 = \text{tr}[(\boldsymbol{V}\boldsymbol{\Delta}\boldsymbol{V}^\top - \boldsymbol{V}\boldsymbol{V}^\top\boldsymbol{Q}\boldsymbol{\Psi}\boldsymbol{Q}^\top\boldsymbol{V}\boldsymbol{V}^\top)^2] \\
&= \text{tr}[(\boldsymbol{V}(\boldsymbol{\Delta} - \boldsymbol{V}^\top\boldsymbol{Q}\boldsymbol{\Psi}\boldsymbol{Q}^\top\boldsymbol{V})\boldsymbol{V}^\top)^2] = \text{tr}[\boldsymbol{V}^2(\boldsymbol{\Delta} - \boldsymbol{V}^\top\boldsymbol{Q}\boldsymbol{\Psi}\boldsymbol{Q}^\top\boldsymbol{V})^2(\boldsymbol{V}^\top)^2] \\
&= \text{tr}[(\boldsymbol{V}^\top)^2\boldsymbol{V}^2(\boldsymbol{\Delta} - \boldsymbol{V}^\top\boldsymbol{Q}\boldsymbol{\Psi}\boldsymbol{Q}^\top\boldsymbol{V})^2] = \text{tr}[(\boldsymbol{\Delta} - \boldsymbol{V}^\top\boldsymbol{Q}\boldsymbol{\Psi}\boldsymbol{Q}^\top\boldsymbol{V})^2]
\end{aligned}$$

Let $\boldsymbol{M} := \boldsymbol{V}^\top\boldsymbol{Q}$, then

$$\begin{aligned}
\min_{\boldsymbol{Z}} \|\boldsymbol{K} - \boldsymbol{Z}\boldsymbol{Z}^\top\|_F^2 &= \min_{\boldsymbol{M},\boldsymbol{\Psi}} \text{tr}[(\boldsymbol{\Delta} - \boldsymbol{M}\boldsymbol{\Psi}\boldsymbol{M}^\top)^2] \\
&= \min_{\boldsymbol{M},\boldsymbol{\Psi}} \text{tr}(\boldsymbol{\Delta}^2) - 2\text{tr}(\boldsymbol{\Delta}\boldsymbol{M}\boldsymbol{\Psi}\boldsymbol{M}^\top) + \text{tr}[(\boldsymbol{M}\boldsymbol{\Psi}\boldsymbol{M}^\top)^2]
\end{aligned}$$

Denote $\mathcal{L} = \text{tr}(\boldsymbol{\Delta}^2) - 2\text{tr}(\boldsymbol{\Delta}\boldsymbol{M}\boldsymbol{\Psi}\boldsymbol{M}^\top) + \text{tr}[(\boldsymbol{M}\boldsymbol{\Psi}\boldsymbol{M}^\top)^2]$. First we take the derivative w.r.t. $\boldsymbol{M}$ and set it to zero:

$$\begin{aligned}
\frac{\partial \mathcal{L}}{\partial \boldsymbol{M}} &= -2\boldsymbol{\Delta}\boldsymbol{M}\boldsymbol{\Psi} + 2(\boldsymbol{M}\boldsymbol{\Psi}\boldsymbol{M}^\top)\boldsymbol{M}\boldsymbol{\Psi} = 0 \\
\Rightarrow \boldsymbol{M}\boldsymbol{\Psi}\boldsymbol{M}^\top &= \boldsymbol{\Delta}
\end{aligned}$$

Before taking the derivative w.r.t. $\boldsymbol{\Psi}$, we first change $\mathcal{L}$ into:

$$\begin{aligned}
\mathcal{L} &= \text{tr}(\boldsymbol{\Delta}^2) - 2\text{tr}(\boldsymbol{\Delta}\boldsymbol{M}\boldsymbol{\Psi}\boldsymbol{M}^\top) + \text{tr}[(\boldsymbol{M}\boldsymbol{\Psi}\boldsymbol{M}^\top)^2] \\
&= \text{tr}(\boldsymbol{\Delta}^2) - 2\text{tr}(\boldsymbol{M}^\top\boldsymbol{\Delta}\boldsymbol{M}\boldsymbol{\Psi}) + \text{tr}[(\boldsymbol{M}^\top\boldsymbol{M}\boldsymbol{\Psi})^2]
\end{aligned}$$

then

$$\begin{aligned}
\frac{\partial \mathcal{L}}{\partial \boldsymbol{\Psi}} &= -2\boldsymbol{M}^\top\boldsymbol{\Delta}\boldsymbol{M} + 2(\boldsymbol{M}^\top\boldsymbol{M}\boldsymbol{\Psi})\boldsymbol{M}^\top\boldsymbol{M} \\
&= -2\boldsymbol{M}^\top\boldsymbol{\Delta}\boldsymbol{M} + 2\boldsymbol{M}^\top(\boldsymbol{M}\boldsymbol{\Psi}\boldsymbol{M}^\top)\boldsymbol{M} = 0 \\
\Rightarrow \boldsymbol{M}^\top\boldsymbol{\Psi}\boldsymbol{M} &= \boldsymbol{\Delta}
\end{aligned}$$

Both FOC points to $\boldsymbol{M}^\top\boldsymbol{\Psi}\boldsymbol{M} = \boldsymbol{\Delta}$. One possible solution to this is

$$\boldsymbol{M} = \boldsymbol{I},\ \boldsymbol{\Psi} = \boldsymbol{\Delta}$$

which means that the minimum of the non-negative objective function $\text{tr}[(\boldsymbol{\Delta} - \boldsymbol{M}\boldsymbol{\Psi}\boldsymbol{M}^\top)^2]$ is 0. Therefore, we have

$$\boldsymbol{M} = \boldsymbol{I} = \boldsymbol{V}^\top\boldsymbol{Q} \Rightarrow \boldsymbol{Q} = \boldsymbol{V}$$

Recall that

$$\boldsymbol{Z}\boldsymbol{Z}^\top = \boldsymbol{Q}\boldsymbol{\Psi}\boldsymbol{Q}^\top = \boldsymbol{V}\boldsymbol{\Delta}\boldsymbol{V}^\top = \boldsymbol{V}\boldsymbol{\Delta}^{\frac{1}{2}}\boldsymbol{\Delta}^{\frac{1}{2}}\boldsymbol{V}^\top \Rightarrow \boldsymbol{Z} = \boldsymbol{V}\boldsymbol{\Delta}^{\frac{1}{2}}$$

Truncating this $\boldsymbol{Z}$ gives us $\boldsymbol{Z}^* = \boldsymbol{V}_L\boldsymbol{\Delta}_L^{\frac{1}{2}} \in \mathbb{R}^{N\times L}$.

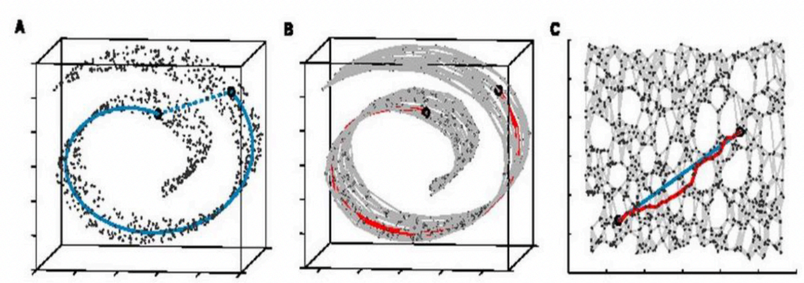
1.2 ISOMAP

ISOMAP

ISOMAP is a special case of MDS where **isometry is kept under geodesic distance** as much as possible. Given a set of data $\{\boldsymbol{x_n}\}_{n=1}^N$

1. Determine the neighbors of each data point and construct a $K-$nearest neighbor (KNN) graph of the data.
2. Compute the shortest path (**Dijkstra/Floyd**) distance between arbitary two nodes and obtain an approximate geodesic distance matrix $\boldsymbol{D} = [d_{ij}] \in \mathbb{R}^{N\times N}$.
3. Compute low-dimensional embedding by MDS similarily

$$\begin{cases} \boldsymbol{K} = -\frac{1}{2}\boldsymbol{C}(\boldsymbol{D}\odot\boldsymbol{D})\boldsymbol{C} \\ \boldsymbol{K} = \boldsymbol{V}\boldsymbol{\Delta}\boldsymbol{V}^\top \end{cases} \Rightarrow \boldsymbol{Z}^* = \boldsymbol{V}_L\boldsymbol{\Delta}_L^{\frac{1}{2}}$$



ISOMAP

1.3 Locally Linear Embedding

Locally Linear Embedding (LLE)

LLE keeps isometry indirectly through inheriting **local linear self-representation power**. Local linear self-representation means that each data point can be represented by a linear combination of its neighbors: given a sample $\boldsymbol{x_i}$ and its K neighbors $\boldsymbol{X_i} = [\boldsymbol{x_1}, \cdots, \boldsymbol{x_K}] \in \mathbb{R}^{D\times K}$ where $d(\boldsymbol{x_i}, \boldsymbol{x_k}) < \tau$, $\forall k = 1, \cdots, K$, $\exists \boldsymbol{w} \in \mathbb{R}^K$, s.t. $\boldsymbol{x_i} \approx \boldsymbol{X_i}\boldsymbol{w_i}$.
In this sense, given $\boldsymbol{X} = [\boldsymbol{x_1}, \cdots, \boldsymbol{x_N}] \in \mathbb{R}^{D\times N}$, LLE aims at finding a low-dimensional embedding $\boldsymbol{Z} = [\boldsymbol{z_1}, \cdots, \boldsymbol{z_N}] \in \mathbb{R}^{L\times N}$ $(L < D)$ that inherits the local linear self-representation relations.

Closed-form Solution for LLE

LLE can be decomposed into 3 steps

1. **Linear Reconstruction by Neighbors:** First, we compute the linear coefficients $\tilde{\boldsymbol{w}}$ by

$$\tilde{\boldsymbol{W}}^* = \arg\min_{\tilde{\boldsymbol{W}}} \sum_{i=1}^N \|\boldsymbol{x_i} - \boldsymbol{X_i}\tilde{\boldsymbol{w}}_{\boldsymbol{i}}\|_2^2 \quad \text{s.t. } \tilde{\boldsymbol{W}}\boldsymbol{1}_K = \boldsymbol{1}_N$$

Here $\tilde{\boldsymbol{W}} = [\tilde{\boldsymbol{w}}_{\boldsymbol{1}}, \cdots, \tilde{\boldsymbol{w}}_{\boldsymbol{N}}]^\top \in \mathbb{R}^{N\times K}$. The coefficient $\tilde{\boldsymbol{w}}_{\boldsymbol{i}} = [\tilde{w}_{i1}, \cdots, \tilde{w}_{iK}]^\top$ for each sample is constrained such that coefficients weighted on each neighbor sums up to 1. $\boldsymbol{x_i}$ refers to the 'sample' and $\boldsymbol{X_i}$ refers to its 'neighbors'.

2. **Linear Embedding:** First we expand the old $\tilde{\boldsymbol{W}} = [\tilde{w}_{ij}] \in \mathbb{R}^{N\times K}$ to $\boldsymbol{W} = [w_{ij}] \in \mathbb{R}^{N\times N}$ by

$$w_{ij} = \begin{cases} \tilde{w}_{ij} & \text{if } \boldsymbol{x_j} \in \text{KNN}(\boldsymbol{x_i}) \\ 0 & \text{otherwise} \end{cases}$$

Compute the embedding $\boldsymbol{Z} \in \mathbb{R}^{L\times N}$ by

$$\boldsymbol{Z}^* = \arg\min_{\boldsymbol{Z}} \sum_{i=1}^{N} \|\boldsymbol{z_i} - \sum_{j=1}^{N} w_{ij}\boldsymbol{z_j}\|_2^2 \quad \text{s.t. } \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{z_i}\boldsymbol{z_i}^\top = \boldsymbol{I}_L, \sum_{i=1}^{N} \boldsymbol{z_i} = \boldsymbol{0}$$

We constraint the embedding to ensure that $\text{Cov}(\boldsymbol{Z}) = \boldsymbol{I}_L$. The second constraint can be temporarily ignored since it can be achieved implicitly. We want to rewrite the object funciton in a more compact form.

$$\begin{aligned} \sum_{i=1}^{N} \|\boldsymbol{z_i} - \sum_{j=1}^{N} w_{ij}\boldsymbol{z_j}\|_2^2 = \sum_{i=1}^{N} \|\boldsymbol{z_i} - \boldsymbol{Z}\boldsymbol{w_i}\|_2^2 &= \sum_{i=1}^{N} \|\boldsymbol{Z}\boldsymbol{1}_i - \boldsymbol{Z}\boldsymbol{w}_i\|_2^2 = \|\boldsymbol{Z} - \boldsymbol{Z}\boldsymbol{W}^\top\|_F^2 \\ &= \text{tr}\left((\boldsymbol{Z} - \boldsymbol{Z}\boldsymbol{W}^\top)(\boldsymbol{Z} - \boldsymbol{Z}\boldsymbol{W}^\top)^\top\right) \\ &= \text{tr}\left(\boldsymbol{Z}(\boldsymbol{I} - \boldsymbol{W} - \boldsymbol{W}^\top + \boldsymbol{W}^\top\boldsymbol{W})\boldsymbol{Z}^\top\right) \end{aligned}$$

where the **alignment matrix** $\boldsymbol{\Phi} = \boldsymbol{I}_N - \boldsymbol{W} - \boldsymbol{W}^\top + \boldsymbol{W}^\top\boldsymbol{W}$.

3. Conduct EVD on $\boldsymbol{\Phi} := \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^\top$. After sorting the eigenvectors from smallest to largest eigenvalues, **we ignore the first eigenvector having zero eigenvalue** and take the L smallest eigenvectors of $\boldsymbol{U}$ with non-zero eigenvalues as the embedding $(\boldsymbol{Z}^\top)^* \in \mathbb{R}^{N\times L}$.

First, for the **linear reconstruction by neighbors, the coefficients $\boldsymbol{W}$ can be computed as follows:** Note that

$$\begin{aligned} \|\boldsymbol{x_i} - \boldsymbol{X_i}\boldsymbol{w_i}\|_2^2 = \|\boldsymbol{x_i}(\boldsymbol{1_K}^\top\boldsymbol{w_i}) - \boldsymbol{X_i}\boldsymbol{w_i}\|_2^2 &= \|(\boldsymbol{x_i}\boldsymbol{1_K}^\top - \boldsymbol{X_i})\boldsymbol{w_i}\|_2^2 \\ &= \boldsymbol{w_i}^\top(\boldsymbol{x_i}\boldsymbol{1_K}^\top - \boldsymbol{X_i})^\top(\boldsymbol{x_i}\boldsymbol{1_K}^\top - \boldsymbol{X_i})\boldsymbol{w_i} \\ &\equiv \boldsymbol{w_i}^\top\boldsymbol{G_i}\boldsymbol{w_i} \end{aligned}$$

where we denote $\boldsymbol{G_i} = (\boldsymbol{x_i}\boldsymbol{1_K}^\top - \boldsymbol{X_i})^\top(\boldsymbol{x_i}\boldsymbol{1_K}^\top - \boldsymbol{X_i}) \in \mathbb{R}^{K\times K}$. The optimization problem is

$$\boldsymbol{W}^* = \arg\min_{\boldsymbol{W}} \sum_{i=1}^{N} \boldsymbol{w_i}^\top\boldsymbol{G_i}\boldsymbol{w_i} \quad \text{s.t. } \boldsymbol{W}\boldsymbol{1}_K = \boldsymbol{1}_N$$

The Lagrangian for this is

$$\begin{aligned} &\mathcal{L}(\boldsymbol{W}, \boldsymbol{\Lambda}) = \sum_{i=1}^{N} \boldsymbol{w_i}^\top\boldsymbol{G_i}\boldsymbol{w_i} - \sum_{i=1}^{N} \lambda_i(\boldsymbol{1}_K^\top\boldsymbol{w_i} - 1) \\ &\Rightarrow \begin{cases} \dfrac{\partial\mathcal{L}}{\partial\boldsymbol{w_i}} = 2\boldsymbol{G_i}\boldsymbol{w_i} - \lambda_i\boldsymbol{1}_K = \boldsymbol{0} \\ \dfrac{\partial\mathcal{L}}{\partial\lambda_i} = \boldsymbol{1}_K^\top\boldsymbol{w_i} - 1 = 0 \end{cases} \Rightarrow \begin{cases} \boldsymbol{w_i} = \dfrac{\lambda_i}{2}\boldsymbol{G_i^{-1}}\boldsymbol{1}_K \\ \boldsymbol{1}_K^\top\dfrac{\lambda_i}{2}\boldsymbol{G_i^{-1}}\boldsymbol{1}_K = 1 \end{cases} \\ &\Rightarrow \begin{cases} \boldsymbol{w_i} = \dfrac{\lambda_i}{2}\boldsymbol{G_i^{-1}}\boldsymbol{1}_K \\ \lambda_i = \dfrac{2}{\boldsymbol{1}_K^\top\boldsymbol{G_i^{-1}}\boldsymbol{1}_K} \end{cases} \Rightarrow \boldsymbol{w_i} = \frac{\boldsymbol{G_i^{-1}}\boldsymbol{1}_K}{\boldsymbol{1}_K^\top\boldsymbol{G_i^{-1}}\boldsymbol{1}_K} \end{aligned}$$

Second, for the derivation of **linear embedding,** our optimization problem is essentially

$$\min \operatorname{tr}(\boldsymbol{Z}\boldsymbol{\Phi}\boldsymbol{Z}^\top) \quad \text{s.t. } \frac{1}{N}\boldsymbol{Z}\boldsymbol{Z}^\top = \boldsymbol{I}_L$$

and therefore the Lagrangian for this is (**Important: Under optimal** $\boldsymbol{\Lambda} \in \mathbb{R}^{L\times L}$)

$$\begin{aligned}\mathcal{L}(\boldsymbol{Z},\boldsymbol{\Lambda}) &= \operatorname{tr}(\boldsymbol{Z}\boldsymbol{\Phi}\boldsymbol{Z}^\top) - \operatorname{tr}(\boldsymbol{\Lambda}^\top(\frac{1}{N}\boldsymbol{Z}\boldsymbol{Z}^\top - \boldsymbol{I}_L))\\ \Rightarrow \frac{\partial \mathcal{L}}{\partial \boldsymbol{Z}} &= 2\boldsymbol{Z}\boldsymbol{\Phi} - \frac{2}{N}\boldsymbol{\Lambda}\boldsymbol{Z} = 0 \Rightarrow \boldsymbol{\Phi}\boldsymbol{Z}^\top = \boldsymbol{Z}^\top(\frac{1}{N}\boldsymbol{\Lambda})\end{aligned}$$

Moreover, recall that our goal is to minimize

$$\operatorname{tr}(\boldsymbol{Z}\boldsymbol{\Phi}\boldsymbol{Z}^\top) = \operatorname{tr}(\boldsymbol{Z}\boldsymbol{Z}^\top\frac{1}{N}\boldsymbol{\Lambda}) = \operatorname{tr}(\frac{1}{N}\boldsymbol{\Lambda}) = \frac{1}{N}\sum_{i=1}^{N}\lambda_i$$

and EVD of $\boldsymbol{\Phi} := \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^\top$. This is means that **under optimal**, we **should pick** L **eigenvectors from** the eigenvectors of $\boldsymbol{\Phi}$ to compose the embedding $(\boldsymbol{Z}^\top)^* \in \mathbb{R}^{N\times L}$. After sorting the eigenvectors from smallest to largest eigenvalues, **we ignore the first eigenvector having zero eigenvalue** and take the L smallest eigenvectors of $\boldsymbol{U}$ with non-zero eigenvalues of $\boldsymbol{\Phi}$ as the embedding $(\boldsymbol{Z}^\top)^*$.

1.4 Laplacian Eigenmap

Laplacian Eigenmap

Given a set of data $\boldsymbol{X} = [\boldsymbol{x}_1, \cdots, \boldsymbol{x}_N] \in \mathbb{R}^{D\times N}$, we construct the similarity matrix $\boldsymbol{A} = [a(\boldsymbol{x_i}, \boldsymbol{x_j})] \in \mathbb{R}^{N\times N}$. A reasonable criterion to get the low-dimensional embedding $\boldsymbol{Z} = [\boldsymbol{z}_1, \cdots, \boldsymbol{z}_N] \in \mathbb{R}^{L\times N}$ is to minimize the following objective function

$$\min_{\boldsymbol{Z}} \sum_{m,n=1}^{N} \|\boldsymbol{z}_m - \boldsymbol{z}_n\|_2^2 a(\boldsymbol{x_m}, \boldsymbol{x_n})$$

because when distance $\|\boldsymbol{z}_m - \boldsymbol{z}_n\|_2^2$ is small, the similarity $a(\boldsymbol{x_m}, \boldsymbol{x_n})$ should be large.

Closed-form Solution of Laplacian Eigenmap

$$\begin{aligned}\boldsymbol{Z} &= \arg\min_{\boldsymbol{Z}} \sum_{m,n=1}^{N} \|\boldsymbol{z}_m - \boldsymbol{z}_n\|_2^2 a(\boldsymbol{x_m}, \boldsymbol{x_n}) = \arg\min_{\boldsymbol{Z}} \sum_{m,n=1}^{N} (\boldsymbol{z_m}^\top\boldsymbol{z_m} - 2\boldsymbol{z_m}^\top\boldsymbol{z_n} + \boldsymbol{z_n}^\top\boldsymbol{z_n})a_{mn}\\ &= \arg\min_{\boldsymbol{Z}} \sum_{m=1}^{N} \boldsymbol{z_m}^\top\boldsymbol{z_m}\left(\sum_{n=1}^{N} a_{mn}\right) + \sum_{n=1}^{N} \boldsymbol{z_n}^\top\boldsymbol{z_n}\left(\sum_{m=1}^{N} a_{mn}\right) - 2\sum_{m,n=1}^{N} \boldsymbol{z_m}^\top\boldsymbol{z_n}a_{mn}\\ &= \arg\min_{\boldsymbol{Z}} 2\operatorname{tr}(\boldsymbol{Z}^\top \operatorname{diag}(\boldsymbol{A}\boldsymbol{1_N})\boldsymbol{Z}) - 2\operatorname{tr}(\boldsymbol{Z}^\top\boldsymbol{A}\boldsymbol{Z})\\ &= \arg\min_{\boldsymbol{Z}} 2\operatorname{tr}(\boldsymbol{Z}^\top(\operatorname{diag}(\boldsymbol{A}\boldsymbol{1_N}) - \boldsymbol{A})\boldsymbol{Z})\\ &= \arg\min_{\boldsymbol{Z}} \operatorname{tr}(\boldsymbol{Z}^\top\boldsymbol{L}\boldsymbol{Z}) \quad \text{where } \boldsymbol{L} = \operatorname{diag}(\boldsymbol{A}\boldsymbol{1_N}) - \boldsymbol{A}\end{aligned}$$

In practice, the Laplacian matrix $\boldsymbol{L}$ is usually normalized by the degree matrix $\boldsymbol{D} = \operatorname{diag}(\boldsymbol{A}\boldsymbol{1_N})$:

$$\widehat{\boldsymbol{L}}_{\text{sym}} = \boldsymbol{D}^{-\frac{1}{2}}\boldsymbol{L}\boldsymbol{D}^{-\frac{1}{2}} = \boldsymbol{D}^{-\frac{1}{2}}(\boldsymbol{D}-\boldsymbol{A})\boldsymbol{D}^{-\frac{1}{2}} = \boldsymbol{I}_N - \boldsymbol{D}^{-\frac{1}{2}}\boldsymbol{A}\boldsymbol{D}^{-\frac{1}{2}} = \boldsymbol{I}_N - \widehat{\boldsymbol{A}}$$

By performing EVD on $\widehat{\boldsymbol{L}}_{\text{sym}} := \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^\top$, we can get the embedding $\boldsymbol{Z}^* = \boldsymbol{U_L} \in \mathbb{R}^{N\times L}$.

In construction of similarity matrix $\boldsymbol{A}$, we can apply the Gram matrix of kernel function such as the RBF kernel:

$$a(\boldsymbol{x_i}, \boldsymbol{x_j}) := K(\boldsymbol{x_i}, \boldsymbol{x_j}) = \exp(-\|\boldsymbol{x_i} - \boldsymbol{x_j}\|_2^2 / h)$$

2 Kernel Methods

Kernel PCA

Suppose our data $\boldsymbol{X} \in \mathbb{R}^{N \times D}$ is non-linearly separable. We can first map the data into a higher-dimensional space $\boldsymbol{\Phi}(\boldsymbol{X}) = [\phi(\boldsymbol{x_1}), \cdots, \phi(\boldsymbol{x_n})]^\top \in \mathbb{R}^{N \times \dim(F)}$ and then perform EVD on the Gram matrix $\boldsymbol{K} = \boldsymbol{\Phi}(\boldsymbol{X})\boldsymbol{\Phi}(\boldsymbol{X})^\top$.

$$\boldsymbol{K} = \boldsymbol{V}\boldsymbol{\Delta}\boldsymbol{V}^\top$$

The PCA corresponds to the top-L eigenvectors of $\boldsymbol{K}$: $\boldsymbol{Z}^* = \boldsymbol{V}_L \boldsymbol{\Delta}_L^{\frac{1}{2}} \in \mathbb{R}^{N \times L}$.

Revisiting MDS and ISOMAP, we can consider them as special cases of Kernel PCA.

- For MDS, $\boldsymbol{K} = -\frac{1}{2}\boldsymbol{C}(\boldsymbol{D} \odot \boldsymbol{D})\boldsymbol{C} = \boldsymbol{C}\boldsymbol{X}\boldsymbol{X}^\top\boldsymbol{C} = \tilde{\boldsymbol{X}}\tilde{\boldsymbol{X}}^\top$ (Linear Kernel)
- For ISOMAP, $\boldsymbol{K} = -\frac{1}{2}\boldsymbol{C}(\boldsymbol{D}_{geo} \odot \boldsymbol{D}_{geo})\boldsymbol{C}$ (Mercer Kernel)

2.1 t-Distributed Stochastic Neighbor Embedding (t-SNE)

t-SNE

Given a dataset $\boldsymbol{X} = [\boldsymbol{x}_1, \cdots, \boldsymbol{x}_N] \in \mathbb{R}^{D \times N}$, first we define a Probability p_{ij} that is proportional to the similarity between $\boldsymbol{x_i}$ and $\boldsymbol{x_j}$:

$$p_{ij} = \frac{p_{j|i} + p_{i|j}}{2N}, \quad p_{ii} = 0$$
$$p_{j|i} = \frac{\exp(-\|\boldsymbol{x_i} - \boldsymbol{x_j}\|_2^2 / 2\sigma_i^2)}{\sum_{k \neq i} \exp(-\|\boldsymbol{x_i} - \boldsymbol{x_k}\|_2^2 / 2\sigma_i^2)}$$

t-SNE aims to learn $\boldsymbol{Z} = [\boldsymbol{z}_1, \cdots, \boldsymbol{z}_N] \in \mathbb{R}^{K \times N}$ (usually $K = 2, 3$ for visualization purposes) that minimizes the KL divergence between p_{ij} and q_{ij}

$$\min_{\boldsymbol{Z}} \mathrm{KL}(P||Q) = \min_{\boldsymbol{Z}} \sum_{i \neq j} p_{ij} \log \frac{p_{ij}}{q_{ij}}$$

where q_{ij} is the similarity between $\boldsymbol{z_i}$ and $\boldsymbol{z_j}$:

$$q_{ij} = \frac{(1 + \|\boldsymbol{z_i} - \boldsymbol{z_j}\|_2^2)^{-1}}{\sum_{k \neq l} (1 + \|\boldsymbol{z_k} - \boldsymbol{z_l}\|_2^2)^{-1}}, \quad q_{ii} = 0$$

where $\{q_{ij}\}$ is the Student-t distribution with df=1. Optimization of KL divergence is done with SGD.

3 Autoencoding

First, let's revisit PCA from a viewpoint of **autoencoding**. Recall that PCA is the least-square data denoising under i.i.d. Gaussian noise,

$$\hat{\boldsymbol{X}} = \arg\min_{\boldsymbol{X}\in\Omega} \|\boldsymbol{X}_{\text{noisy}} - \boldsymbol{X}\|_F^2 = \boldsymbol{U}_L \boldsymbol{\Sigma}_L \boldsymbol{V}_L^\top, \text{ where } \boldsymbol{X}_{\text{noisy}} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top$$

referring to the construction of principal components and the corresponding reconstruction. This can be viewed as a special case of autoencoding where the encoder and decoder are linear transformations.

$$\begin{aligned} \text{Encoder: } & \boldsymbol{Z} = \boldsymbol{X}_{\text{noisy}} \boldsymbol{V}_L^\top \\ \text{Decoder: } & \boldsymbol{X}^* = \boldsymbol{X}_{\text{noisy}} \boldsymbol{V}_L^\top \boldsymbol{V}_L \end{aligned}$$

Here $\boldsymbol{V}_{\boldsymbol{L}}^\top$ and $\boldsymbol{V}_L$ are the encoder and decoder respectively.

Autoencoders

In general, a typical autocoder consists of

$$\begin{aligned} \text{Encoder: } & f : \mathcal{X} \to \mathcal{Z} \\ \text{Decoder: } & g : \mathcal{Z} \to \mathcal{X} \end{aligned}$$

Given a set of data $\boldsymbol{X} = \{\boldsymbol{x}_1, \cdots, \boldsymbol{x}_N\} \in \mathbb{R}^{D\times N}$, the autoencoder aims to learn the encoder and decoder that minimize the reconstruction error

$$\min_{f,g} \sum_{i=1}^{N} \text{loss}\left(\boldsymbol{x}_i - g(f(\boldsymbol{x}_i))\right) + \text{regularization}(q_{\boldsymbol{Z}|\boldsymbol{X}}, p_{\boldsymbol{Z}})$$

where $q_{\boldsymbol{Z}|\boldsymbol{X}}$ is the posterior distribution of latent space $\boldsymbol{Z}$ given dataset $\boldsymbol{X}$ and $p_{\boldsymbol{Z}}$ is the prior distribution of $\boldsymbol{Z}$.

References

- Multidimensional Scaling, Sammon Mapping, and Isomap: Tutorial and Survey
- Locally Linear Embedding and its Variants: Tutorial and Survey

Appendix

Classic MDS

The specific process of deriving $\boldsymbol{K} = \tilde{\boldsymbol{X}}\tilde{\boldsymbol{X}}^\top$ is as follows: Note that

$$\begin{aligned}
\boldsymbol{X} \odot \boldsymbol{X}\mathbf{1}_D\mathbf{1}_N &= \begin{bmatrix} x_{11}^2 & \cdots & x_{1D}^2 \\ \vdots & \ddots & \vdots \\ x_{N1}^2 & \cdots & x_{ND}^2 \end{bmatrix}_{N\times D} \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix}_{D\times 1} \begin{bmatrix} 1 & \cdots & 1 \end{bmatrix}_{1\times N} = \begin{bmatrix} \boldsymbol{x_1}^\top \boldsymbol{x_1} & \cdots & \boldsymbol{x_1}^\top \boldsymbol{x_1} \\ \vdots & \ddots & \vdots \\ \boldsymbol{x_N}^\top \boldsymbol{x_N} & \cdots & \boldsymbol{x_N}^\top \boldsymbol{x_N} \end{bmatrix}_{N\times N} \\
&= \begin{bmatrix} \langle \boldsymbol{x_1}, \boldsymbol{x_1} \rangle & \cdots & \langle \boldsymbol{x_1}, \boldsymbol{x_1} \rangle \\ \vdots & \ddots & \vdots \\ \langle \boldsymbol{x_N}, \boldsymbol{x_N} \rangle & \cdots & \langle \boldsymbol{x_N}, \boldsymbol{x_N} \rangle \end{bmatrix}_{N\times N}
\end{aligned}$$

and

$$\begin{aligned}
d_{ij} &= \|\boldsymbol{x_i} - \boldsymbol{x_j}\|_2^2 = (\boldsymbol{x_i} - \boldsymbol{x_j})^\top (\boldsymbol{x_i} - \boldsymbol{x_j}) = \boldsymbol{x_i}^\top \boldsymbol{x_i} - 2\boldsymbol{x_i}^\top \boldsymbol{x_j} + \boldsymbol{x_j}^\top \boldsymbol{x_j} \\
&= \langle \boldsymbol{x_i}, \boldsymbol{x_i} \rangle + \langle \boldsymbol{x_j}, \boldsymbol{x_j} \rangle - 2\langle \boldsymbol{x_i}, \boldsymbol{x_j} \rangle
\end{aligned}$$

We can decompose

$$\boldsymbol{D} \odot \boldsymbol{D} = \begin{bmatrix} d_{11}^2 & \cdots & d_{1N}^2 \\ \vdots & \ddots & \vdots \\ d_{N1}^2 & \cdots & d_{NN}^2 \end{bmatrix} = (\boldsymbol{X} \odot \boldsymbol{X}\mathbf{1}_D\mathbf{1}_N) + (\boldsymbol{X} \odot \boldsymbol{X}\mathbf{1}_D\mathbf{1}_N)^\top - 2\boldsymbol{X}\boldsymbol{X}^\top$$

$\boldsymbol{C}$ is essentially a **centering matrix** since

$$\begin{aligned}
\boldsymbol{C}\boldsymbol{X} &= (\boldsymbol{I}_N - \frac{1}{N}\mathbf{1}_{N\times N})\boldsymbol{X} = \boldsymbol{X} - \frac{1}{N}\mathbf{1}_N\mathbf{1}_N^\top \boldsymbol{X} \\
&= \begin{bmatrix} x_{11} & \cdots & x_{1D} \\ \vdots & \ddots & \vdots \\ x_{N1} & \cdots & x_{ND} \end{bmatrix} - \begin{bmatrix} \frac{x_{11} + \cdots + x_{N1}}{N} & \cdots & \frac{x_{1D} + \cdots + x_{ND}}{N} \\ \vdots & \ddots & \vdots \\ \frac{x_{11} + \cdots + x_{N1}}{N} & \cdots & \frac{x_{1D} + \cdots + x_{ND}}{N} \end{bmatrix} \\
&= \begin{bmatrix} x_{11} - \overline{x_1} & \cdots & x_{1D} - \overline{x_D} \\ \vdots & \ddots & \vdots \\ x_{N1} - \overline{x_1} & \cdots & x_{ND} - \overline{x_D} \end{bmatrix}
\end{aligned}$$

Therefore, $\tilde{\boldsymbol{X}} = \boldsymbol{C}\boldsymbol{X}$ is the zero-meaned data matrix of $\boldsymbol{X}$. One can verify that after **double centralizing**,

$$\boldsymbol{C}\left((\boldsymbol{X} \odot \boldsymbol{X}\mathbf{1}_D\mathbf{1}_N + (\boldsymbol{X} \odot \boldsymbol{X}\mathbf{1}_D\mathbf{1}_N)^\top)\right)\boldsymbol{C} = 0_{N\times N}$$

Therefore, the inner product data $\boldsymbol{K}$ is essentially

$$\begin{aligned}
\boldsymbol{K} &= -\frac{1}{2}\boldsymbol{C}(\boldsymbol{D} \odot \boldsymbol{D})\boldsymbol{C} = -\frac{1}{2}\boldsymbol{C}(\boldsymbol{X} \odot \boldsymbol{X}\mathbf{1}_D\mathbf{1}_N + (\boldsymbol{X} \odot \boldsymbol{X}\mathbf{1}_D\mathbf{1}_N)^\top - 2\boldsymbol{X}\boldsymbol{X}^\top)\boldsymbol{C} \\
&= \boldsymbol{C}\boldsymbol{X}\boldsymbol{X}^\top\boldsymbol{C} = \boldsymbol{C}\boldsymbol{X}(\boldsymbol{C}\boldsymbol{X})^\top = \tilde{\boldsymbol{X}}\tilde{\boldsymbol{X}}^\top
\end{aligned}$$