Mini-course on Manifold Learning

This lecture note grew out of my class at NTU. It is by no means original. Most materials came from the lecture note of Professor Hau-Tieng Wu. Sometime I copied word for word from his note. Please don't distribute it. This is only for teaching purpose.

Contents

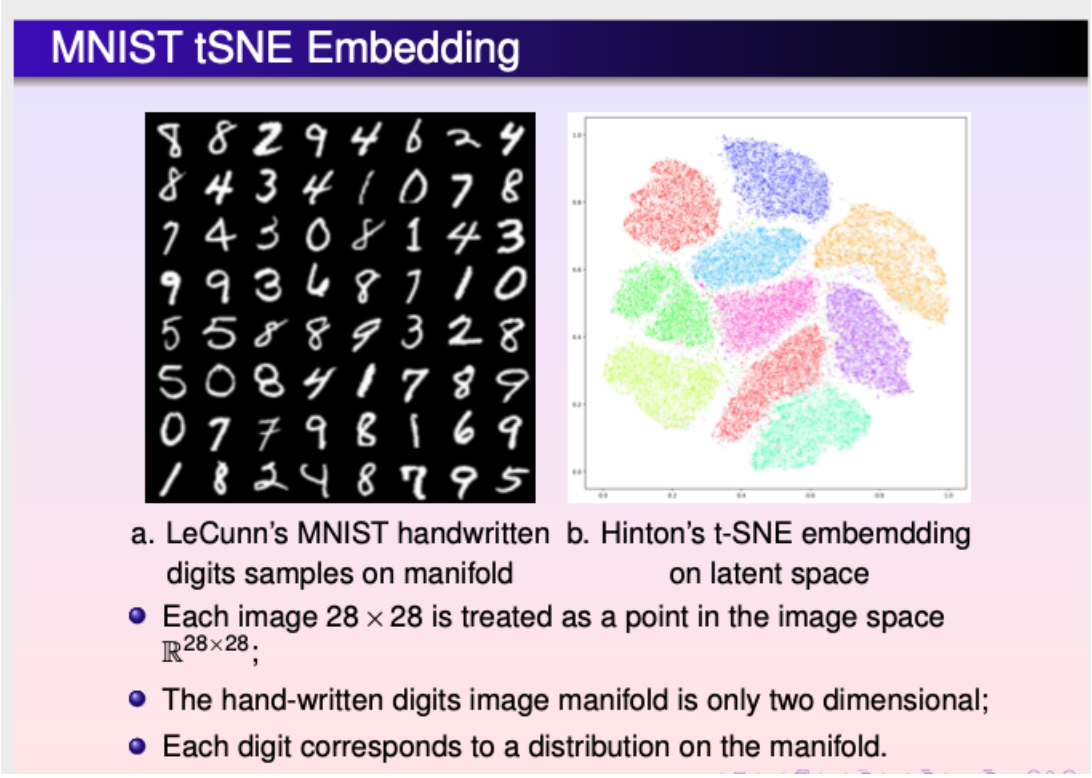
CHAPTER 1

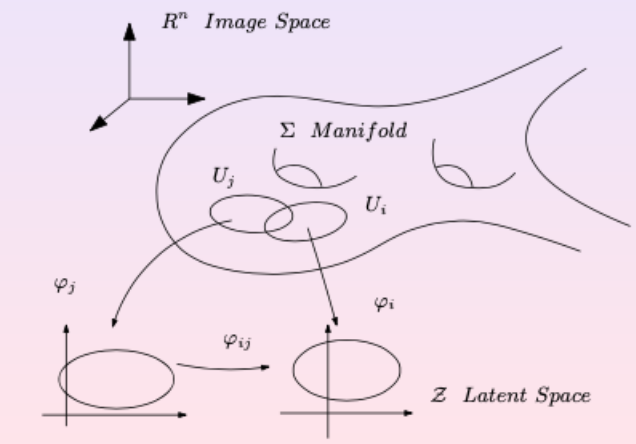
Introduction to Manifold Learning

1. Deep Learning

Deep learning is the mainstream technique for many machine learning tasks, including image recognition, machine translation, speech recognition, and so on. Despite its success, the theoretical understanding on how it works remains primitive. T he mathematical theories behind deep learning can be partially explained by the well accepted manifold distribution and the clustering distribution hypothesis:

- Manifold Distribution Natural high dimensional data concentrates close to a non-linear low-dimensional manifold.
- Clustering Distribution The distances among the probability distributions of subclasses on the manifold are far enough to discriminate them. Deep learning method can learn and represent the manifold structure, and transform the probability distributions.





- Ambient Space- image space $\mathbb{R}^n$
- manifold - Support of a distribution μ
- parameter domain - latent space $\mathbb{R}^m$
- coordinates map φ_i- encoding/decoding maps
- φ_{ij} controls the probability measure

The central tasks for Deep Learning are

1 Learn the manifold structure from the data;
2 Represent the manifold implicitly or explicitly.

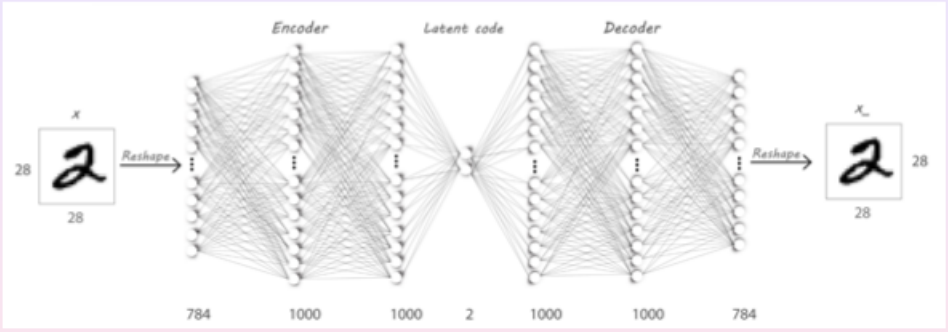


Figure: Auto-encoder architecture.

Ambient space $\mathcal{X}$, latent space $\mathcal{Z}$, encoding map $\varphi_\theta : \mathcal{X} \to \mathcal{Z}$, decoding map $\psi_\theta : \mathcal{Z} \to \mathcal{X}$.

2. Dimension Reduction and Manifold Learning

In real world applications, many objects can only be electronically represented with high-dimensional data—-speech signals, images, videos, text documents, hand- writing letters and numbers, fingerprints, and medical images, etc. We often need to analyze a large amount of data and process them. For instance, we often need to identify a persons fingerprint, to search text documents by keywords on the Internet, to find certain hidden patterns in images, to trace objects from videos, and so on. To complete these tasks, we develop systems to process data. However, due to the high dimension of data, a system directly processing them may be very complicated and unstable so that it is infeasible. In fact, many systems are only effective for relatively low dimensional data. When the dimensions of data are higher than the tolerance of such a system, they cannot be processed. Therefore, in order to process high-dimensional data in the systems, dimensionality reduction becomes necessary.

The task of manifold learning is to identify the low dimensional manifold structure of the the high dimensional data. Geometric harmonics provides a framework for taking data in high-dimensional measurement spaces and embedding them in low dimensional Euclidean space according to a similarity measure. Euclidean coordinates then characterize the ”manifold” on (or near) which the data live.

In Figure 3, we illustrate the organizational ability of the diffusion maps on a collection of images given in random order. The inputs are 2-D gray scale pictures of the object in 3D in various positions, each viewed as a $32 \times 32 = 1024$ dimensional vector. To calculate the embedding, one constructs the Markov matrix as above, and computes the first few eigenfunctions. The top two eigenfunctions reveal the orientation of 3D, and organize the data accordingly, see Figure 3.

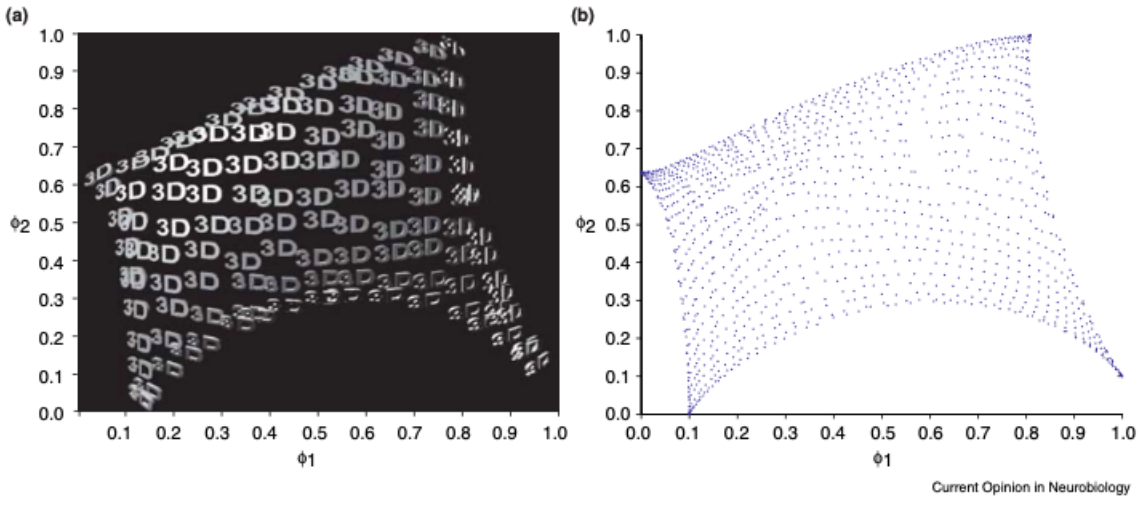


Figure 3

Diffusion embedding of a set of pictures of "3D". Organization emerging from a collection of images given in random order (data = $\{x_i\}$). **(a)** The images are displayed according to their location in the two-dimensional diffusion embedding $(\phi_1(x_i), \phi_2(x_i))$, displayed in **(b)**. The coordinates capture (perceive) the orientation of the picture in 3D.

The next example (Figure 5) represents an organization of the configuration space of lip images that arise from a single speaker. No structure is assumed. The local similarity between images, viewed as high- dimensional vectors, organizes them as above in the first three diffusion coordinates. Different locations in the diffusion plot correspond to different clusters of strongly related lip images.

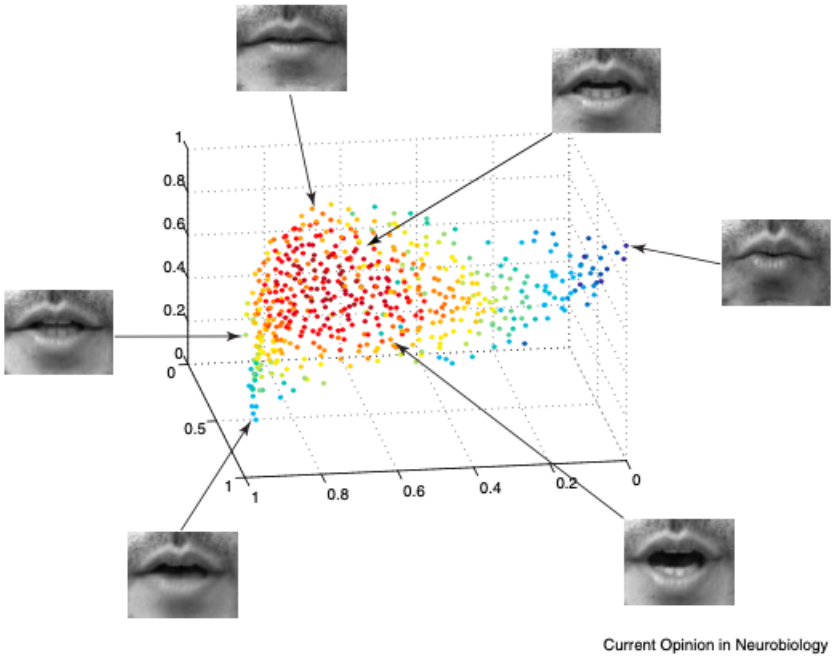


Figure 5
Diffusion embedding of images of lips. The lip alphabet is learnt from a set of pictures of the lips of a speaker. The manifold structure and its parameters are parametrized by the three top eigenfunctions (axes in the figure) of the diffusion, and this parametrization can be used to lip-read. An interpretation of the low order eigenfunctions is openness of the mouth and exposure of teeth.

In this lecture notes, we introduce three important techniques in manifold learning theories.

1 Principal Component Analysis (PCA) Given a set of points $\mathcal{X} = \{X_1, X_2, \cdots, X_n\} \subset \mathbf{R}^p$ where p is very large and $k << p$ (k is much less that p). How can one find a $k-$dimensional affine space $\mathcal{P}_k$ such that the sum of the distance square to $\mathcal{P}_k$ is minimum. Note that $\mathcal{P}_k$ is parametrized by the Grassmanian $G(k,p)$. Note that $G(k,p)$ is a compact manifold. Therefore the minimizer must exist. We will explain how to find it explicitly.

2 Multi-dimensional scaling The problem is that if we are given the distance square of n points then how we can determine the position of these data points in Euclidean space.

3 Diffusion maps If we are given a graph structure with a "metric" then how we can "embed "these data points in Euclidean spaces.

CHAPTER 2

Principal Component Analysis (PCA)

Principal Component Analysis (PCA) is a linear dimensionality reduction method dating back to Pearson (1901) [([F]) and it is one of the most useful techniques in exploratory data analysis. PCA can be applied to a data set comprising of n vectors $\{X_1, X_2, \cdots, X_n\} \in \mathbf{R}^p$ and in turn returns a new orthonormal basis to $\mathbf{R}^p$ whose elements are terms the principal components. It is important that the method is completely data-dependent, that is, the new basis is only a function of the data. There are two equivalent viewpoints of PCA:

- **Low-dimensional projection with maximum variability.** The goal in PCA is to find an orthogonal projection of the centered data to a lower dimensional space that captures the largest variability of the data. More precisely, we are looking for an orthogonal transformation which maps the centered data points to a new set called principal components so that the variance of the first component is as high as possible and the variance of each of the rest is the highest possible given the constraint that it is orthogonal to the previous ones.
- **Low-dimensional projection with minimum error.** The goal is to find a k-dimensional affine space in $\mathbf{R}^p$ that best approximates $\{X_1, X_2, \cdots, X_n\}$ in the least squares sense.

In the following, our convention is that we identify a vector in $\mathbf{R}^p$ with a column vector. We begin with the second viewpoint and along the way show it is equivalent to the first one. Take the data set $\mathcal{X} = \{X_1, X_2, \cdots, X_n\} \in \mathbf{R}^p$ in the Euclidean space and construct the data matrix

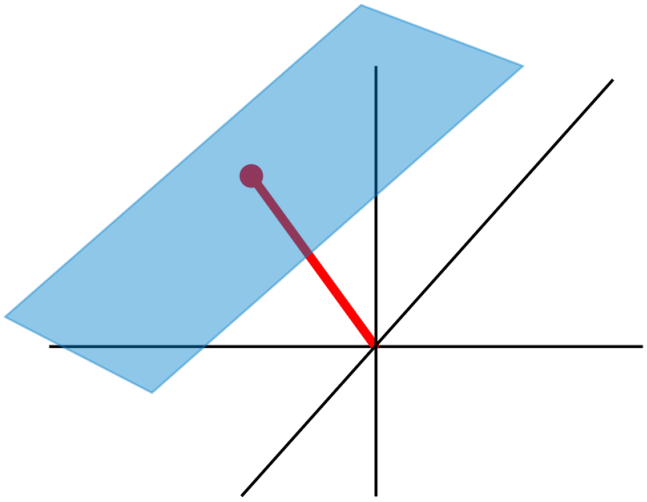
$$X = \begin{bmatrix} X_1 & X_2 & \cdots & X_{n-1} & X_n \end{bmatrix} \in \mathbf{R}^{p \times n}.$$

The goal is to find a k-dim affine subspace to best fit the data to achieve the dimension reduction purpose. Recall that the affine subspaces of a vector space V are the subsets of V of the form

$$p + S = \{p + s | s \in S\}$$

where $S \subset V$ is a subspace of V. Suppose the subspace S is spanned by the basis $\{s_1, \cdots, s_k\}$ Then

$$p + S = \{p + \sum_{i=1}^{k} a_k s_k | a_1, \cdots, a_k \in \mathbf{R}\} = \{p + Ta | T = [s_1, \cdots, s_k] \text{ and } a = [a_1, \cdots, a_k]^T\}.$$



We can parameterize an affine space as $\mu + U\beta$, where $\mu \in \mathbf{R}^p, U = \begin{bmatrix} U_1 & U_2 & \cdots & U_{k-1} & U_k \end{bmatrix} \in \mathbf{R}^{p\times k}$. is a matrix whose columns are orthonormal (i.e., $U^T U = I_{k\times k}$) and $\beta \in \mathbf{R}^k$ provides the reduced representation of points in the affine space.

In other words, we would like to find $U \in \mathbf{R}^{p\times k}, \mu \in \mathbf{R}^p$, and $\beta_1, \cdots, \beta_n \in \mathbf{R}^k$ such that

$$X_i \approx \mu + U\beta_i, i = 1, 2, \cdots, n$$

or in matrix notation

$$X \approx \mu \mathbf{1}^T + U\mathcal{B}$$

where $\mu \in \mathbf{R}^p, \mathbf{1} = \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} \in \mathbf{R}^n, U \in \mathbf{R}^{p\times k}, U^T U = I_{k\times k}, \mathcal{B} = \begin{bmatrix} \beta_1 & \cdots & \beta_n \end{bmatrix} \in \mathbf{R}^{k\times n}$.

We would like the approximation to minimize the l^2 norm, i.e.

$$\sum_{i=1}^{n} ||X_i - (\mu + U\beta_i)||^2_{\mathbf{R}^p}$$

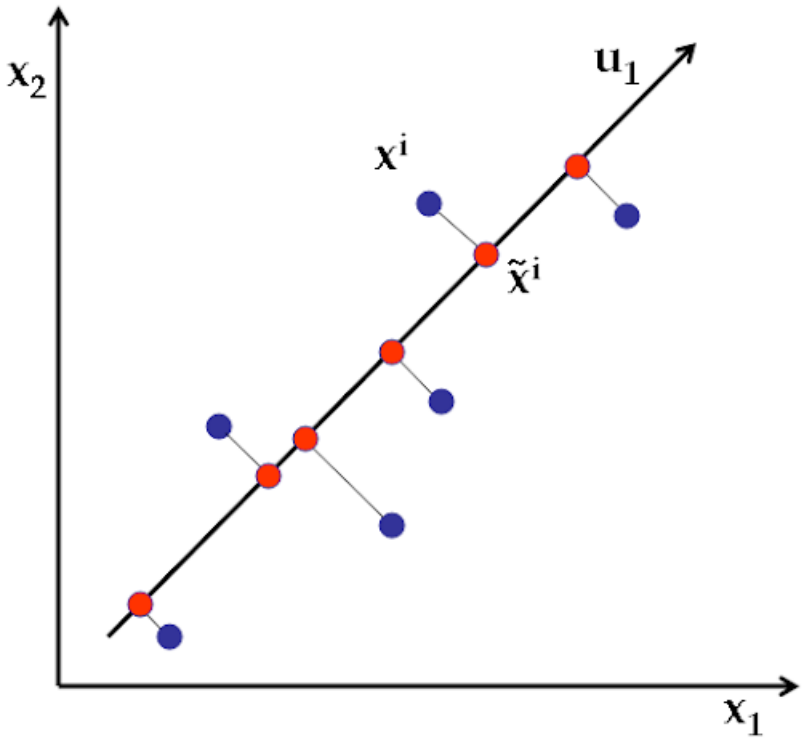
which is equivalient

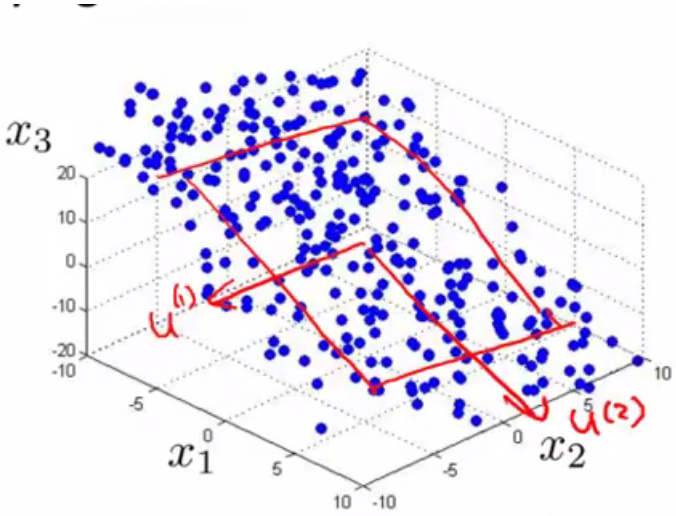
$$||X - (\mu \mathbf{1}^T + U\mathcal{B})||^2_F.$$

Note that the Frobenius now between two matrix $A, B \in \mathbf{R}^{p\times n}$ is

$$||A - B||^2_F = Trace(A - B)^T (A - B) = \sum_{i=1}^{n} ||A_i - B_i||^2_{\mathbf{R}^p}$$

where A_i, B_i are the column vectors of A and B.





Reduce data from 3D to 2D

We would like to have the mean of the data set $\{X_1, X_2, \cdots, X_n\}$ to be the same as the mean of the approximation set $\{\mu + U\beta_i\}_{i=1}^n$. Thus $\frac{\sum_{i=1}^n X_i}{n} = \frac{\sum_{i=1}^n \mu + U\beta_i}{n} = \mu + \frac{\sum_{i=1}^n U\beta_i}{n}$. We may choose $\mu = \frac{\sum_{i=1}^n X_i}{n}$ and impose the condition $U(\sum_{i=1}^n \beta_i) = 0$. Since the column vectors of U are independent , we have $\sum_{i=1}^n \beta_i = 0$, i.e. $\mathcal{B}\mathbf{1} = 0$.. However, we can determine the value

of μ with the constraint $\mathcal{B}\mathbf{1} = 0$. We consider the following optimization problem to solve the problem

$$\text{Minimize}_{\mathcal{B}\in\mathbf{R}^{k\times n},\mathcal{B}\mathbf{1}=0,\mu\in\mathcal{R}^n,U\in\mathbf{R}^{p\times k},U^TU=I_{k\times k}} I(\mathcal{B},\mu,U) = ||X-(\mu\mathbf{1}^T+U\mathcal{B})||_F^2.$$

To solve this minimization problem, we use the following steps.

Step 1: Determine μ, $\mu = \frac{\sum_{i=1}^n X_i}{n}$

Note that

$$||X-(\mu\mathbf{1}^T+U\mathcal{B})||_F^2 = \sum_{i=1}^n ||X_i-(\mu+U\beta_i)||_{\mathbf{R}^p}^2$$

where $\beta_i = \mathcal{B}e_i$.

First, we can rewrite

$$I = \sum_{i=1}^n ||\mu-(X_i-U\beta_i)||_{\mathbf{R}^p}^2 = \sum_{i=1}^n ||\mu||^2+||X_i-U\beta_i||^2-2\langle\mu,X_i-U\beta_i\rangle$$

To solve this optimization problem, we first note that when the minimization is achieved we have

$$0 = \nabla_\mu I = \sum_{i=1}^n 2(\mu-(X_i-U\beta_i))$$

This implies that $\sum_{i=1}^n(\mu-(X_i-U\beta_i)) = 0$ and $\mu = \frac{\sum_{i=1}^n(X_i+U\beta_i)}{n}$ Using $\sum_{i=1}^n\beta_i = 0$, we have $\mu = \frac{\sum_{i=1}^n X_i}{n}$.

Step 2: Determine $\mathcal{B}$, $\mathcal{B} = U^T(X-\bar{\mu}\mathbf{1}^T)$ where $\bar{\mu} = \frac{\sum_{i=1}^n X_i}{n}$

Let $\bar{\mu} = \frac{\sum_{i=1}^n X_i}{n}$, Now

$$\begin{aligned}
&I(\bar{\mu},U,\mathcal{B})\\
=&\sum_{i=1}^n ||(X_i-\bar{\mu})-U\beta_i)||_{\mathbf{R}^p}^2\\
=&\sum_{i=1}^n ||X_i-\bar{\mu}||^2+\beta_i^TU^TU\beta_i-2(X_i-\bar{\mu})^TU\beta_i\\
=&\sum_{i=1}^n ||X_i-\bar{\mu}||^2+||\beta_i||^2-2(X_i-\bar{\mu})^TU\beta_i\\
=&\sum_{i=1}^n ||X_i-\bar{\mu}||^2+||\beta_i||^2-2\langle\beta_i,U^T(X_i-\bar{\mu})\rangle
\end{aligned}$$

Minimize w.r.t. β_i: $\nabla_{\beta_i} I = 2\beta_i-2U^T(X_i-\bar{\mu}) = 0$. Then $\beta_i = U^T(X_i-\bar{\mu})$ and we can rewrite them as $\mathcal{B} = [\beta_1,\cdots,\beta_n] = U^T(X-\mu\mathbf{1}^T)$. Note that $\sum_{i=1}^n U^T(X_i-\bar{\mu}) = 0$. So it satisfies the constraint $\mathcal{B}\mathbf{1} = 0$.

Step 3: Determine U

Note that $\bar{\mu} = \frac{\sum_{i=1}^n X_i}{n} = \frac{1}{n}X\mathbf{1}$. Let $\hat{\mathcal{B}} = U^T(X-\bar{\mu}\mathbf{1}^T)$ and $Y = X-\bar{\mu}\mathbf{1}^T = X-\frac{1}{n}X\mathbf{1}\mathbf{1}^T = XH$ where $H = I-\frac{1}{n}\mathbf{1}\mathbf{1}^\mathbf{T}$. Now we can rewrite the functional as

$$I(\bar{\mu},\hat{\mathcal{B}},U) = ||X-(\bar{\mu}\mathbf{1}^T+UU^T(X-\bar{\mu}\mathbf{1}^T))||_F^2 = ||(X-\bar{\mu}\mathbf{1}^T)-UU^T(X-\bar{\mu}\mathbf{1}^T)||_F^2 = ||Y-UU^TY||_F^2$$

Thus

$$\text{Minimize}_{\mathcal{B}\in\mathbf{R}^{k\times n},\mathcal{B}\mathbf{1}=0,\mu\in\mathcal{R}^n,U\in\mathbf{R}^{p\times k},U^TU=I_{k\times k}} I(\mathcal{B},\mu,U) = \text{Minimize}_{U\in\mathbf{R}^{p\times k},U^TU=I_{k\times k}} ||Y-UU^TY||_F^2.$$

Let $P = UU^T \in \mathbf{R}^{p\times p}$. Then $P^T = P$ and $P^2 = UU^TUU^T = UU^T = P$. We have

$$\begin{aligned} ||Y-UU^TY||_F^2 &= ||Y-PY||_F^2 = \text{Trace}(Y-PY)^T(Y-PY) \\ &= \text{Trace}(Y^T-Y^TP)(Y-PY) = \text{Trace}(Y^TY-2Y^TPY+Y^TP^2Y) \\ &= \text{Trace}(Y^TY-Y^TPY) = \text{Trace}(Y^TY-Y^TUU^TY) \end{aligned}$$

Since $\text{Trace}(Y^TY) = \text{constant}$,

$$\text{argmin}_{U\in\mathbf{R}^{p\times k},U^TU=I_{k\times k}}\text{Trace}(Y^TY-Y^TUU^TY) = \text{argmax}_{U\in\mathbf{R}^{p\times k},U^TU=I_{k\times k}}\text{Trace}(Y^TUU^TY)$$

$$=\text{argmax}_{U\in\mathbf{R}^{p\times k},U^TU=I_{k\times k}}\text{Trace}(U^TYY^TU)$$

Note that we use the fact that $\text{Trace}(Y^TUU^TY) = \text{Trace}(U^TYY^TU)$. Now it is clear that the largest k orthonormal eigenvectors of YY^T maximize the rightmost sum above. Hence we know that the best approximation of the data set $\mathcal{X} = \{X_1,\cdots,X_n\}$ by a k-dimensional affine space is the affine space thru the mean $\bar{\mu} = \frac{\sum_{i=1}^n X_i}{n}$ spanned by $\{U_1,\cdots U_k\}$ of largest k orthonormal eigenvectors of $(X-\bar{\mu}\mathbf{1}^T)(X-\bar{\mu}\mathbf{1}^T)^T = (XH)^T(XH)$ where $H = I-\frac{1}{n}\mathbf{1}\mathbf{1}^T$.

Recall the best approximation of X_i is

$$\begin{aligned} &\bar{\mu}+U\beta_i \\ =&\bar{\mu}+UU^T(X_i-\bar{\mu}) \\ =&\bar{\mu}+\begin{bmatrix} U_1 & U_2 & \cdots & U_{k-1} & U_k \end{bmatrix}\begin{bmatrix} U_1^T \\ U_2^T \\ \vdots \\ U_{k-1}^T \\ U_k^T \end{bmatrix}(X_i-\bar{\mu}) \\ =&\bar{\mu}+\sum_{j=1}^k\langle X_i-\bar{\mu},U_j\rangle U_j \end{aligned}$$

Thus we have

$$X_i-\bar{\mu} \approx \sum_{j=1}^k\langle X_i-\bar{\mu},U_j\rangle U_j, i=1,2,\cdots,n.$$

This means that we approximate $X_i-\bar{\mu}$ by the orthogonal projection of $X_i-\bar{\mu}$ to its top k eigenspace spanned by $\{U_1,\cdots,U_k\}$. Next, we summarize the PCA algorithm.

PCA algorithm

Given the data set $\{X_1, X_2, \cdots, X_n\} \in \mathbf{R}^p$ in the Euclidean space.

$$\text{Let } X = \begin{bmatrix} X_1 & X_2 & \cdots & X_{n-1} & X_n \end{bmatrix} \in \mathbf{R}^{p\times n}.$$

Step 1 Find the mean of the data set $\bar{\mu} = \frac{\sum_{i=1}^n X_i}{n}$ and evaluate the covariance matrix Σ_n of X where $\Sigma_n = (X - \bar{\mu}\mathbf{1}^T)(X - \bar{\mu}\mathbf{1}^T)^T = (XH)(XH)^T$ where $H = I - \frac{1}{n}\mathbf{1}\mathbf{1}^T \in \mathbf{R}^{n\times n}$.

Step 2 Apply the spectral decomposition to Σ_n and get $\Sigma_n = U\Lambda U^T$ where

$$U = [U_1, U_2, \cdots, U_p] \in O(p), \Lambda = diag(\lambda_1, \cdots, \lambda_p)$$

and $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_p$ are the eigenvectors and eigenvalues of Σ_n respectively.

Step 3 Project X_i into the affine apace thru the mean $\bar{\mu}$ and spanned by the top k eigenvectors by

$$X_i \approx \bar{\mu} + \sum_{j=1}^{k} \langle X_i - \bar{\mu}, U_j\rangle U_j.$$

Here k is chosen based on a pre-defined criterion determined by the user. If the purpose is visualization, choose $k = 3$.

CHAPTER 3

Multidimensional scaling (MDS)

The MDS algorithm is aiming for isometric embedding. The question we would like to ask is the following.

Take a metric space (M, d) and a finite sample $X = \{X_i\} \subset M$. Suppose we are provided with pairwise distance $d(X_i, X_j)$. The mission is to find an embedding $i : X \mapsto \mathbf{R}^p$ such that $||\iota(X_i) - \iota(X_j)||_{\mathbf{R}^p} = d(X_i, X_j)$ or as close as possible. In other words, solve the following optimization problem:

$$\text{argmin} \sum_{1 \le i,j \le n} \omega_{ij}(||\iota(X_i) - \iota(X_j)|| - d(X_i, X_j))^2.$$

where $\omega_{ij} = 0$ when the pairwise information is missing, 1 otherwise.

There are several different ways to choose ω_{ij}. When $\omega_{ij} = 1$, the algorithm is called the Kruskal-Shephard scaling; when $\omega_{ij} = \frac{1}{d(X_i, X_j)}$, the algorithm is called the Sammon scaling. To solve these optimization problems, we could apply the gradient descent algorithm.

1. Classical MDS

In the following, we mainly discuss the classical MDS. Suppose we are given the distance square of n points in Euclidean space. How can we find n points in Euclidean space to realize the distance between n points. We will show that when $M = \mathbf{R}^p$, we could convert the problem so that it could be solved by the spectral method, and later it will be linked back to the graph Laplacian framework. In this setup, the algorithm is called the classical MDS algorithm. We introduce an important operator.

DEFINITION 3.1. The matrix $H = I - \frac{1}{n}\mathbf{1}\mathbf{1}^T \in \mathbf{R}^{n \times n}$, where I is a $n \times n$ identity matrix, is called the centering matrix.

The left multiplication of this operator is nothing but removing the mean from a dataset with n vectors Let $X = [X_1, X_2, \cdots, X_n]$ and $\bar{\mu} = \frac{\sum_{i=1}^n X_i}{n}$. Note that $X\mathbf{1} = \sum_{i=1}^n X_i$. Then

$$XH = X - \frac{1}{n}(X\mathbf{1})\mathbf{1}^T = X - \bar{\mu}\mathbf{1}^T = [X_1 - \bar{\mu}, X_2 - \bar{\mu}, \cdots X_n - \bar{\mu}].$$

Note that if we solve the problem by the classical MDS algorithm, the recovery is correct up to global rotation, translation and reflection. To study how accurate the classical MDS algorithm works and what it means, we need to following fact about the centering matrix.

PROPOSITION 3.2. *The centering matrix $H = I - \frac{1}{n}\mathbf{1}\mathbf{1}^T \in \mathbf{R}^{n \times n}$ satisfies the following properties.*
(a) H is idempotent, that is, $H^2 = H$;
(b) $H \ge 0$;

(c) $\mathbf{1} \in Ker(H)$ *and* $\mathbf{1}^T H = 0$*;*
(d) H *is a projection to the subspacespace perpendicular to the vector* $\mathbf{1}$*.*

PROOF. By direct computation and using $\mathbf{1}^T\mathbf{1} = n$,

$$H^2 = (I - \frac{1}{n}\mathbf{1}\mathbf{1}^T)(I - \frac{1}{n}\mathbf{1}\mathbf{1}^T) = I - 2\frac{1}{n}\mathbf{1}\mathbf{1}^T + \frac{1}{n^2}\cdot n\mathbf{1}\mathbf{1}^T = H.$$

Next, we have $q^t Hq = ||q||^2 - \frac{1}{n}\langle q, \mathbf{1}\rangle^2 = ||q||^2 - \langle q, \frac{1}{\sqrt{n}}\mathbf{1}\rangle^2 \geq 0$ by Cauchy-Schwartz inequality and the fact that $\frac{1}{\sqrt{n}}\mathbf{1}$ is a unit vector. Thus H is nonnegative definite.

$$H\mathbf{1} = (I - \frac{1}{n}\mathbf{1}\mathbf{1}^T)\mathbf{1} = \mathbf{1} - \mathbf{1} = 0.$$

So $\mathbf{1} \in Kek(H)$. Note that H is symmetric. We have $0 = (H\mathbf{1})^T = \mathbf{1}^T H^T = \mathbf{1}^T H$.

$$Hv = (I - \frac{1}{n}\mathbf{1}\mathbf{1}^T)v = v - \langle v, \frac{1}{\sqrt{n}}\mathbf{1}\rangle\frac{1}{\sqrt{n}}\mathbf{1}.$$

Note that $\frac{1}{\sqrt{n}}\mathbf{1}$ is a unit vector . So H is a projection to space perpendicular to 1.

□

We now turn our attention to why this algorithm works. We first have the following theorem describing the relationship between the inner product and distance.

THEOREM 3.3. *Suppose we have* $\{X_i\}_{i=1}^n \subset \mathbf{R}^p$ *and* $d(X_i, X_j) = ||X_i - X_j||_{\mathbf{R}^p}$*. Denote* $X = [X_1, X_2, \cdots, X_n] \in \mathbf{R}^{m\times n}$ *and* $\hat{X} = XH$ *where* $H = I - \frac{1}{n}\mathbf{1}\mathbf{1}^T$ *. Then we have*

$$\hat{X}^T\hat{X} = -\frac{1}{2}HDH$$

where $D_{ij} = d(X_i, X_j)^2$*.*

PROOF. Using H is symmetric, we have $\hat{X}^T\hat{X} = (XH)^T XH = HX^T XH$. $(X^T X)_{ij} = \langle X_i, X_j\rangle$. $D_{ij} = ||X_i - X_j||^2 = \langle X_i, X_i\rangle + \langle X_j, X_j\rangle - 2\langle X_i, X_j\rangle$. Let k be the column vector

$$k = \begin{bmatrix} ||X_1||^2 \\ ||X_2||^2 \\ \cdots \\ ||X_n||^2 \end{bmatrix}.$$

Note that $k_{i1} = ||X_i||^2$ and $(k^T)_{1j} = ||X_j||^2$. Then $(k\mathbf{1}^T)_{ij} = k_{i1}1_{1j}^T = k_{i1}\cdot 1 = ||X_i||^2$ and $(\mathbf{1}k^T)_{ij} = \mathbf{1}_{i1}k_{1j}^T = ||X_j||^2$. So $D_{ij} = (k\mathbf{1}^T + \mathbf{1}k^T - 2X^T X)_{ij}$ and $D = k\mathbf{1}^T + \mathbf{1}k^T - 2X^T X$. Thus $X^T X = \frac{1}{2}(k\mathbf{1}^T + \mathbf{1}k^T - D)$. Using $H^T = H$, $H\mathbf{1} = 0$ and $\mathbf{1}^T H = 0$, we have

$$\hat{X}^T\hat{X} = H^T X^T XH = HX^T XH = H\cdot\frac{1}{2}(k\mathbf{1}^T + \mathbf{1}k^T - D)\cdot H = -\frac{1}{2}HDH^T.$$

Note that we have used $H\mathbf{1} = 0$ and $\mathbf{1}^T H = 0$. □

Now we present the classical MDS algorithm.

MDS algorithm. Given the data set $D_{ij} = ||X_i - X_j||^2$ for $1 \le i, j \le n$. D_{ij} is the pairwise distance square between points X_i and X_j in the point cloud $\{X_j\}_{j=1}^n$. Here $\{X_j\}_{j=1}^n$ is unknown.

Step 1 Apply the spectral decomposition $-\frac{1}{2}HDH = U\Lambda U^T = \hat{U}^T\hat{U}$ where $U \in O(n)$, $\Lambda = diag(\lambda_1, \cdots, \lambda_n), \lambda_1 \ge \lambda_2 \ge \cdots \ge \lambda_n \ge 0$. Let $\hat{U} = \Lambda^{\frac{1}{2}}U^T = [\hat{U}_1, \cdots, \hat{U}_n]$. Remark: Here we assume that all eigenvalues of $-\frac{1}{2}HDH$ are nonnegative. If this is not the case, we replace the negative eigenvalues by zero.

Step 2 Solve the problem by estimating X_i by $X_i := \hat{U}_i \in \mathbf{R}^n$.

REMARK 3.4. Suppose there are p positive eigenvalue with

$$\lambda_1 \ge \lambda_2 \ge \cdots \ge \lambda_p > \lambda_{p+1} = \cdots = \lambda_n = 0.$$

Then $(\Lambda^{\frac{1}{2}}U^T)_{ij} = \sum_{k=1}^n \Lambda^{\frac{1}{2}}_{ik}U^T_{kj}$. Thus if $i \ge p + i$ then $\Lambda^{\frac{1}{2}}{}_{ik} = 0$ and $(\Lambda^{\frac{1}{2}}U^T)_{ij} = 0$ if $i \ge p + 1$. $X_i := \hat{U}_i \in \mathbf{R}^p$. This gives an embedding to $\mathbf{R}^p$. In MDS, the number of positive eigenvalues from $-\frac{1}{2}HDH$ gives us the dimension of the space where the data points come from.

Next we discuss the modified MDS.

Modified MDS algorithm. Given the data set D_{ij} for $1 \le i, j \le n$. D_{ij} is the "pairwise distance square" between points X_i and X_j in the point cloud $\{X_j\}_{j=1}^n$. Here $\{X_j\}_{j=1}^n$ is unknown. Here D_{ij} may not be realized as the pairwise distance square between n points. We can still run the mofified MDS algorithm.

Step 1 Apply the spectral decomposition $-\frac{1}{2}HDH = U\Lambda U^T = \hat{U}^T\hat{U}$ where $U \in O(n)$, $\Lambda = diag(\lambda_1, \cdots, \lambda_n), \lambda_1 \ge \lambda_2 \ge \cdots \ge \lambda_p > 0 \ge \lambda_{p+1} \ge \cdots \ge \lambda_n$. Take $\bar{\Lambda} = diag(\lambda_1, \cdots, \lambda_p, 0, \cdots 0)$. Here we only keep the positive eigenvalue. Let $\hat{U} = \bar{\Lambda}^{\frac{1}{2}}U^T = [\hat{U}_1, \cdots, \hat{U}_n]$.

Step 2 Solve the problem by estimating X_i by $X_i := \hat{U}_i \in \mathbf{R}^p$.

2. MDS in inner product space

PROOF. Using H is symmetric, we have $\hat{X}^T\hat{X} = (XH)^TXH = HX^TXH$. $(X^TQX)_{ij} = \langle X_i, X_j\rangle_Q$. $D_{ij} = \langle X_i - X_j, X_i - X_j, \rangle_Q = \langle X_i, X_i\rangle_Q + \langle X_j, X_j\rangle_Q - 2\langle X_i, X_j\rangle_Q$. Let k be the column vector

$$k = \begin{bmatrix} ||X_1||_Q^2 \\ ||X_2||_Q^2 \\ \cdots \\ ||X_n||_Q^2 \end{bmatrix}.$$

Note that $k_{i1} = ||X_i||_Q^2$ and $(k^T)_{1j} = ||X_j||_Q^2$. Then $(k\mathbf{1}^T)_{ij} = k_{i1}1^T_{1j} = k_{i1} \cdot 1 = ||X_i||_Q^2$ and $(\mathbf{1}k^T)_{ij} = \mathbf{1}_{i1}k^T_{1j} = ||X_j||_Q^2$. So $D_{ij} = (k\mathbf{1}^T + \mathbf{1}k^T - 2X^TX)_{ij}$ and $D = k\mathbf{1}^T + \mathbf{1}k^T - 2X^TQX$. Thus $X^TQX = \frac{1}{2}(k\mathbf{1}^T + \mathbf{1}k^T - D)$. Using $H^T = H$, $H\mathbf{1} = 0$ and $\mathbf{1}^TH = 0$, we have

$$\hat{X}^TQ\hat{X} = H^TX^TQXH = HX^TQXH = H \cdot \frac{1}{2}(k\mathbf{1}^T + \mathbf{1}k^T - D) \cdot H = -\frac{1}{2}HDH^T.$$

Note that we have used $H\mathbf{1} = 0$ and $\mathbf{1}^T H = 0$. □

3. ISOMAP

ISOMAP was proposed by Tanenbaum, de Silva and Langford in 2000 and published on Science. ISOMAP can be viewed as a nonlinear extension of MDS. Indeed, the pairwise Euclidean distance considered in MDS is replaced by the geodesic distance. The ISOMAP algorithm is shown in the following Algorithm.

ISOMAP algorithm. Given the data set $\mathcal{X} = \{X_j\}_{j=1}^n \subset \mathbf{R}^p$. Here $\{X_j\}_{j=1}^n$ is known.

Step 1 Find the geodesic distance square between any two points in $\mathcal{X}$by, for example, the Dijkstras algorithm or Floyds algorithm.

Step 2 Run Modified MDS with $D_{ij} =$ the geodesic distance square between X_i and X_j.

4. Relation between PCA and MDS

While classical MDS and PCA look quite different, they are indeed intimately related.

In PCA, we start with a data set $\{X_1, X_2, \cdots, X_n\} \in \mathbf{R}^p$ in the Euclidean space.

$$\text{Let } X = \begin{bmatrix} X_1 & X_2 & \cdots & X_{n-1} & X_n \end{bmatrix} \in \mathbf{R}^{p\times n}.$$

Then we find the mean of the data set $\bar{\mu} = \frac{\sum_{i=1}^n X_i}{n}$ and evaluate the covariance matrix Σ_n of X where $\Sigma_n = \frac{1}{n-1}(X - \bar{\mu}\mathbf{1}^T)(X - \bar{\mu}\mathbf{1}^T)^T = \frac{1}{n-1}(XH)(XH)^T$ where $H = I - \frac{1}{n}\mathbf{1}\mathbf{1}^T$. Next we apply the spectral decomposition to Σ_n and get $\Sigma_n = U\Lambda U^T$ and we use the eigenvectors as principal eigenvectors.

In MDS, we are given the data set $D_{ij} = ||X_i - X_j||^2$ for $1 \le i, j \le n$. D_{ij} is the pairwise distance square between points X_i and X_j in the point cloud $\{X_j\}_{j=1}^n \subset \mathbf{R}^p$. Here $\{X_j\}_{j=1}^n$ is unknown. We have $-\frac{1}{2}HDH = (XH)^T(XH)$. Then spectral decomposition $-\frac{1}{2}HDH$ to find the embedding map.

So PCA is closely related to $(XH)(XH)^T$ and MDS is closely related to $(XH)^T(XH)$. We can show that they have the same nonzero eigenvalues and their eigenvectors are related by the following proposition.

PROPOSITION 3.5. *If u_i and $\lambda_i \neq 0$ are eigenvector and eigenvalue of $(XH)(XH)^T$ then $(XH)^T u_i \neq 0$ is also an eigenvector of $(XH)^T(XH)$ with eigenvalue λ_i.*

If y_i and $\lambda_i \neq 0$ are eigenvector and eigenvalue of $(XH)^T(XH)$ then $(XH)y_i \neq 0$ is also an eigenvector of $(XH)(XH)^T$ with eigenvalue λ_i.

PROOF. Suppose u_i is an eigenvector of $(XH)(XH)^T$ with nonzero eigenvalue λ_i. Then $(XH)(XH)^T u_i = \lambda_i u_i$. We know that $y_i = (XH)^T u_i \neq 0$ (otherwise $u_i = \frac{(XH)y^i}{\lambda_i} = 0$). Now $(XH)^T(XH)y^i = \lambda_i(XH)^T u_i = \lambda_i y_i$. Thus $y_i = (XH)^T u_i$ is an eigenvector of $(XH)^T(XH)$ with eigenvalue λ_i.

Suppose y_i is an eigenvector of $(XH)^T(XH)$ with nonzero eigenvalue λ_i. Then $(XH)^T(XH)y_i = \lambda_i y_i$. We know that $u_i = (XH)y_i \neq 0$. Now $(XH)(XH)^T u^i = \lambda_i(XH)y_i = \lambda_i u_i$. Thus $u_i = (XH)y_i$ is an eigenvector of $(XH)(XH)^T$ with eigenvalue λ_i. □

CHAPTER 4

Diffusion Maps

In our world, many objects can only be electronically represented with high-dimensional data—-speech signals, images, videos, text documents, hand- writing letters and numbers, fingerprints, and medical images, etc. We often need to analyze a large amount of data and process them. For instance, we often need to identify a persons fingerprint, to search text documents by keywords on the Internet, to find certain hidden patterns in images, to trace objects from videos, and so on. To complete these tasks, we develop systems to process data. However, due to the high dimension of data, a system directly processing them may be very complicated and unstable so that it is infeasible. In fact, many systems are only effective for relatively low dimensional data. When the dimensions of data are higher than the tolerance of such a system, they cannot be processed. Therefore, in order to process high-dimensional data in the systems, dimensionality reduction becomes necessary.

In practice, high-dimensional data are often not truly high-dimensional. It is a consensus in the high-dimensional data analysis community that the points of high-dimensional data usually reside on a much low-dimensional manifold. Assume that a data set $\mathcal{X} = \{x_\alpha\}_{\alpha \in A}$ resides on an k–dimensional manifold M that is embedded in $\mathbf{R^d}$:$M \subset \mathbf{R^d}$. Then we call d the extrinsic dimension of $\mathcal{X}$ and d the intrinsic dimension of $\mathcal{X}$. From the viewpoint of statistics, $\mathcal{X}$ can be considered as a sample set of a random vector $\mathcal{X}$ in $\mathbf{R^d}$ .If $\mathcal{X}$ is governed by k–independent variables, that is, there are a random vector $Y \in \mathbf{R}^k$ and an invertible analytic function $f : \mathbf{R}^k \mapsto \mathbf{R}^d$ such that $f(Y) = \mathcal{X}$, then the random vector $\mathcal{X}$ is said to have intrinsic dimension d. The low intrinsic dimensionality of high-dimensional data is the key to the feasibility of dimensionality reduction. Due to the low intrinsic dimension of data, we can reduce the (extrinsic) dimension without losing much information for many types of real-life high-dimensional data, avoiding many of the curses of dimensionality. In one sense, dimensionality reduction is processing to find a certain parameterization of the manifold which the points of data reside on.

Examples for dimensionality reduction:

- Statistical mechanics: an ideal gas contains an Avogadro number of particles ($\approx 10^{23}$), the microscopic description of the system have an enormous number of degrees of freedom but the macroscopic equation of state is $PV = NRT$ where V is the volume of the gas, N is the number of particles in the gas, R is the gas constant, T is the Kelvin temperature, and P is the pressure (in pascals).

  We need only a few macroscopic variables to describe the system all other degrees of freedom can be viewed as noise.
- Image analysis: given natural images, medical imaging, hyperspectral imaging, and hand-written digits, we would like to be able to perform tasks such as classification, image segmentation, denoising, etc.

1. Dimensionality Reduction

If f is a mapping from $[0,1] \mapsto \mathbf{R}$, a reasonable numerical approximation of f could consist in dividing the segment $[0,1]$ into 100 grid points and evaluating f at these points. In this case, f is identified as a vector in $\mathbf{R}^{100}$.

Suppose now you want to approximate a function f of 1000 variables and each variable is defined on $[0.1]$. Similarly, if we dividing the segment $[0,1]$ into 100 grid points You will need 100^{1000} grid points for the same kind of approximation. The main issue is that no machine is currently able to handle such an exponential amount of data.

In many real life datasets we view the data as points in a high dimensional ambient space but the intrinsic dimensionality of the data can be much smaller. For example, consider a dataset consisting of face images, where every image corresponds to different rotation of the head. Every image is 112 by 92 pixels can be viewed as a point in $\mathbf{R}^{10304}$ ($112 \times 92 = 10304$). However, face images are far from being randomly distributed in that high dimensional Euclidean space. The rotation angle is the single physically meaningful parameter describing the images. We will say that the intrinsic dimension of the dataset $= 1$. The face images are samples from a one-dimensional curve embedded in $\mathbf{R}^{10304}$. Notice that this curve can have twists and folds so that linearly projecting it may confuse the natural ordering of the images (based on the intrinsic parameter). More generally, data can lie on or near a low dimensional Riemannian manifold embedded in the ambient space. This calls for non-linear dimensionality reduction methods and for tools for parameterization of the data manifold.

Fig. 1.1 Left: set of images randomly permuted. This is the input of the algorithm. Right: output of the algorithm, the sequence is recorded with respect to the angle of rotation of the head (the sequence is to be read from left to right, and top down). Source: S. Lafon dissertation, Yale University 2004; see also Graham and Allinson 1998

2. Motivation

Given n data points $\{x_1, \cdots, x_n\}$ in $\mathbf{R}^d$. Consider the problem of mapping these data points to the real line.

Let's say that we want to assign a single coordinate to every point on the data, which we'll write as f, with $f(x_i) = f_i \in \mathbf{R}$ being the coordinate for x_i in $\mathbf{R}$.

Because there are n data points, instead of treating f as a function, we can also treat it as an $n \times 1$ matrix, which for simplicity I'll also write f.

Let's say that we want to minimize the following energy

$$E(f) = \sum_{i,j=1}^{n} A_{ij}(f_i - f_j)^2$$

For simplicity, we assume that A is symmetric and A is a transition matrix with $A_{ij} \geq 0$, i.e. $\sum_{j=1}^{n} A_{ij} = \sum_{i=1}^{n} A_{ij} = 1$, This should force the coordinates of points which are very similar $A_{ij} \approx 1$ to be close to each other, but let the coordinates of deeply dissimilar points $A_{ij} \approx 0$ vary freely.

Let us look at the quadratic form $\sum_{i,j=1}^{n} A_{ij}(f_i - f_j)^2$. Using $A_{ij} = A_{ji}$, $\sum_{j=1}^{n} A_{ij} = \sum_{i=1}^{n} A_{ij} = 1$, we can rewrite it as

$$\begin{aligned}
\sum_{i,j=1}^{n} A_{ij}(f_i - f_j)^2 &= \sum_{i=1}^{n} f_i^2 (\sum_{j=1}^{n} A_{ij}) - 2 \sum_{i,j=1}^{n} f_i A_{ij} f_j + \sum_{j=1}^{n} f_j^2 (\sum_{i=1}^{n} A_{ij}) \\
&= \sum_{i=1}^{n} f_i^2 - 2 \sum_{i,j=1}^{n} f_i A_{ij} f_j + \sum_{j=1}^{n} f_j^2 \\
&= 2 f^T (I - A) f
\end{aligned}$$

where I is the $n \times n$ identity matrix.

Since $A_{ij} \geq 0$ for $1 \leq i, j \leq n$, we have $E(f) = \sum_{i,j=1}^{n} A_{ij}(f_i - f_j)^2 \geq 0$. The global minimum of $E(f)$ is clearly 0. To avoid the uninteresting minimum of $E(f)$ at $f = 0$, we impose the constraint $f^T f = 1$ to minimize $E(f) = 2 f^T (I - A) f$.

The minimizer f is the first eigenvector of the symmetric matrix $L = I - A$ corresponding to the smallest eigenvalue, i.e $Lf = \lambda f$ and the minimum of E is 2λ where λ is the first (smallest) eigenvalue of L. Note that $(I - A)f = \lambda f$ implies $Af = (1 - \lambda)f$. This implies $1 - \lambda$ is the largest eigenvalue of A

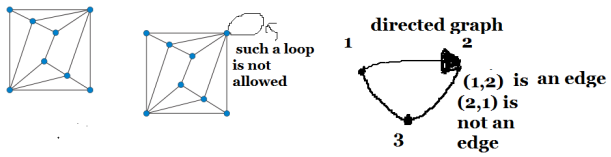
Since A is symmetric with $A_{ij} \geq 0$ and $\sum_{j=1}^{n} A_{ij} = 1$, we know that $||A||_\infty = 1$ ($||A||_\infty = max_{1 \leq i \leq n} \sum_{j=1}^{b} |A_{ij}|$) a matrix norm) and the absolute value of the eigenvalues of A are all less or equal to 1. Let $\mathbf{1} = [1, \cdots, 1]^T$ be the column vector that is 1 in each entry. Then , $A\mathbf{1} = \mathbf{1}$. So the largest eigenvalue of A is 1.

This implies that the smallest eigenvalue of $L = I - A$ is 0 with $f = \frac{\mathbf{1}}{\sqrt{n}}$, So this is not an interesting case.

To find an interesting coordinate function, we should look the second eigenvalue of L and second eigenfunction of $L = I - A$ which to minimize the energy

$$Minimize \quad f^T (I - A) f \text{ subject to } \quad f^t f = 1 \text{ and } f^T \mathbf{1} = 0.$$

Next, we explain how to construct A from the graph associated with data points.



3. Affinity Graph and affinity matrix

We start from the introducing the general framework for the high dimensional data analysis.

DEFINITION 4.1. A graph is a pair $G = (V, E)$, where $V = \{x_1, \cdots, x_n\}$ is a set of vertices and $E \subset V \times V$ is the set of edges. A vertex x_i is called isolated if there is no $(x_i, x_j) \in E$ or $(x_j, x_i) \in E$, where $j \neq i$. The graph G is undirected if E is symmetric, that is, for all $(x_i, x_j) \in E$, $(x_j, x_i) \in E$, and is directed otherwise.

Note that we allow the existence of "loops", that is, (x_i, x_i) could be an edge. To ease the notation, when there is no danger of confusion, we use i to denote the vertex x_i.

DEFINITION 4.2. An affinity graph is a triple $G = (V, E, \omega)$, where (V, E) is a graph and $\omega : E \mapsto R_+ = \{x \in \mathbf{R} | x > 0\}$. We call ω the affinity function. An affinity graph is undirected if (V, E) is undirected and ω is symmetric; that is, $\omega(i, j) = \omega(j, i)$ for all edges $(i, j) \in E$, and is directed otherwise.

Suppose $(i, j) \in E$. $\omega(i, j)$ can regarded as some kind of "distance function" between two vertices. It is clear that we can convert a graph into an affinity graph by setting a function ω defined on E so that $\omega(i, j) = 1$ for all $(i, j) \in E$. And it is obvious that an affinity graph becomes a graph if we ignore ω. Next, we introduce an equivalent way to represent a given graph or an affinity graph.

DEFINITION 4.3. (Adjacency matrix). Given a graph $G = (V, E)$ so that $|V| = n$, the adjacency matrix of G is the matrix $W \in \mathbf{R}^{n \times n}$ defined by $W_{i,j} = \begin{cases} 1 & if\ i, j \in E \\ 0 & otherwise \end{cases}$

DEFINITION 4.4. (Affnity matrix). Given a affinity graph $G = (V, E, \omega)$ so that $V = \{x_i\}_{i=1}^n$, the affinity matrix of G is the matrix $W \in \mathbb{R}^{n \times n}$ defined by $W_{i,j} = \begin{cases} \omega_{i,j} & if\ i, j \in E \\ 0 & otherwise \end{cases}$

Note that an adjacency matrix is nothing but an affinity matrix if we convert a graph into an affinity graph. Note that there is an one to one relationship between the affinity matrix and the affinity graph. In the future, when there is no danger of confusion, we will use the term affinity matrix and affinity graph interchangeably.

We have an equivalent way to represent a function defined on the graph.

DEFINITION 4.5. A function defined on the vertex set $f : V \mapsto \mathbf{R}$ is a $|V|$–dim vector $v \in R^{|V|}$ so that $v_i = f(i)$ for all $i \in V$; a function de

ned on the edge set $g : E \mapsto \mathbf{R}$ is a $|E|$–dim vector $u \in \mathbf{R}^{|E|}$.

DEFINITION 4.6. (Degree function). Let G be an affinity graph. The degree function is $d : V \mapsto \mathbb{R}_+$ defined by

$$d(i) = \sum_{(i,j)\in E} W_{i,j}.$$

Define the degree matrix $D \in \mathbb{R}^{n\times n}$ to be the diagonal matrix with $D_{i,i} = d(i)$.

Note that when the graph is undirected and there is no isolated vertex, the degree function d is positive; that is, $d(i) > 0$ for all $i \in V$ and the degree matrix D is invertible. Since G has no isolated vertices, we can define an invertible $n \times n$ diagonal matrix

$$D = (D_{ij})_{1\le i,j\le n} \text{ where } D_{ij} = d_i\delta_{ij},$$

,i.e.

$$D = \begin{pmatrix} d_1 & 0 & \cdots & 0 \\ 0 & d_2 & \cdots & 0 \\ 0 & \vdots & \ddots & 0 \\ 0 & 0 & \cdots & d_n \end{pmatrix}.$$

4. Graph Laplacian and random walk on the graph

In this section, we focus on the undirected graph $G = (V, E)$ with n vertices.

DEFINITION 4.7. Let $G = (V, E, \omega)$ be an undirected affinity graph with n vertices. The unnormalized graph Laplacian (GL) is defined as $\tilde{L} := D - W$. When there is no isolated vertex, the normalized graph Laplacian (NGL) is defined as $L := I_n - D^{-1}W$. where I_n is the $n \times n$ identity matrix, and the symmetrized normalized graph Laplacian is defined as

$$\mathcal{L} := I_n - D^{-\frac{1}{2}}WD^{-\frac{1}{2}}.$$

We know that $\tilde{L}$ and $\mathcal{L}$ are symmetric while L is not. The normalized and unnormalized versions are related by $\tilde{L} = DL$. Furthermore, L is similar to $\mathcal{L}$ via

$$\mathcal{L} = D^{\frac{1}{2}}LD^{-\frac{1}{2}}.$$

To simplify the discussion, from now on we will assume that all graphs we consider do not have an isolated vertex. Note that this is not a stringent assumption. Indeed, we could freely remove the isolated vertex when we analyze the data.

DEFINITION 4.8. Let G be an affinity graph with $|V| = n$. Define the transition matrix of the random walk on the graph as

$$A := D^{-1}W.$$

Note that $D^{-1}W$ is not symmetric. But it similar to $D^{\frac{1}{2}}AD^{-\frac{1}{2}} = D^{-\frac{1}{2}}WD^{-\frac{1}{2}}$ which is symmetric and also a transition matrix.

Then the transition matrix is defined to be $A = D^{-1}W$. Note that $\sum_{j=1}^n A_{ij} = \sum_{j,k=1}^n \frac{1}{d_i}\delta_{ik}W_{kj} = \sum_{j=1}^n \frac{1}{d_i}W_{ij} = 1$. There is a probability interpretation of the transition matrix A - the entry $A_{i;j}$ can be thought of as the probability of moving from i to j in one step of a random walk on G; that is, A is the transition matrix of a finite Markov process on G. Similarly, $(A^k)_{i;j}$ describes

the probability moving from i to j in k steps.
Note that

$$\begin{aligned}\sum_{j=1}^{n}(A^k)_{ij} &= \sum_{j_1,\cdots,j_k,j=1}^{n} A_{ij_1}A_{j_1j_2}\cdots A_{j_{k-1}j_k}A_{j_kj}\\ &= \sum_{j_1=1}^{n} A_{ij_1}\sum_{j_2=1}^{n} A_{j_1j_2}\cdots\sum_{j_k=1}^{n} A_{j_{k-1}j_k}\sum_{j=1}^{n} A_{j_kj}\\ &=1.\end{aligned}$$

5. Some spectral properties of the graph Laplacian

In this subsection, we provide some basic spectral properties of the GL so that we could introduce another algorithm for high dimensional data analysis, the "diffusion maps".

Notation:Denote $\sigma(M)$ to be the spectrum of a given matrix M and $\rho(M)$ to be the associated spectral radius, i.e $\rho(M) = max_{f\neq 0}\frac{||f^TMf||}{f^Tf} = max\{|\lambda| \quad | \quad \lambda \text{ is an eigenvalue of } M\}$. To fix the notation, denote $L\phi_l = \lambda_l\phi_l$, where $\lambda_1, \lambda_2, \cdots$. Recall the following definition

DEFINITION 4.9. The Rayleigh quotient of a matrix $M \in R^{n\times n}$ is defined as

$$RM(v) := \frac{\langle v, Mv\rangle}{\langle v, v\rangle}$$

where v is a non-zero n-dim vector.

We start from studying the spectral behavior of the un-normalized Laplacian $\widetilde{L}$.

PROPOSITION 4.10. *The unnormalized graph Laplacian $\widetilde{L} = D - W$ is nonnegative definite and $\sigma(\tilde{L}) \subset [0, 2\rho(D)]$.*

PROOF. Let $f \in R^n$. Using $d_i = \sum_{j=1}^{n} W_{ij}$, then

$$\begin{aligned}f^T\widetilde{L}f = f^T(D-W)f &= \sum_{i,j=1}^{n} f_i(d_i\delta_{ij} - W_{ij})f_j = \sum_{i=1}^{n} d_if_i^2 - \sum_{i,j=1}^{n} f_iW_{ij}f_j\\ &= \sum_{i=1}^{n}\sum_{j=1}^{n} W_{ij}f_i^2 - \sum_{i,j=1}^{n} f_iW_{ij}f_j\\ &= \frac{1}{2}\Big(\sum_{i=1}^{n}\sum_{j=1}^{n} W_{ij}f_i^2 + \sum_{j=1}^{n}\sum_{i=1}^{n} W_{ji}f_j^2 - \sum_{i,j=1}^{n} 2f_iW_{ij}f_j\Big)\end{aligned}$$

Recall that W is symmetric and $W_{ij} \geq 0$, we have $W_{ij} = W_{ji}$ and previous equation can be written as a complete square

$$f^T\widetilde{L}f = \frac{1}{2}\sum_{i=1,j=1}^{n} W_{ij}(f_i - f_j)^2 \geq 0.$$

Note that this is zero if $f_1 = f_2 = \cdots = f_n$. This implies that the smallest eigenvalue of $\widetilde{L}$ is 0 with eigenvector $\mathbf{1} = (1, 1, \cdots, 1)^T$.

Note that $(f_i - f_j)^2 \leq 2f_i^2 + 2f_j^2$ and $\rho(D) = max\{d_i\}_{i=1}^n$.. So

$$\begin{aligned} & f^T\widetilde{L}f \\ =&\frac{1}{2}\sum_{i=1,j=1}^{n} W_{ij}(f_i - f_j)^2 \leq \sum_{i=1,j=1}^{n} W_{ij}(f_i^2 + f_j^2) \\ =&\sum_{i=1,j=1}^{n} W_{ij}f_i^2 + \sum_{i=1,j=1}^{n} W_{ij}f_j^2 \\ =&2\sum_i d_i f_i^2 \leq 2\rho(D)f^T f \end{aligned}$$

So $\rho(\widetilde{L}) \leq 2\rho(D)$. □

Remark: From this computation, we can observe that $\mathbf{1}^T\widetilde{L}\mathbf{1} = \frac{1}{2}\sum_{i=1,j=1}^n W_{ij}(\mathbf{1}_i - \mathbf{1}_j)^2 = 0$ where $\mathbf{1} = (1, 1, \cdots, 1)^T$ and $(\widetilde{L}\mathbf{1})_i = \sum_{j=1}^n (D - W)_{ij}\mathbf{1}_j = \sum_{j=1}^n d_i\delta_{ij} - W_{ij} = d_i - d_i = 0$. Thus $\widetilde{L}\mathbf{1} = 0$.

Recall that $\tilde{L} = DL$. In the case where D is invertible (G has no isolated vertices), then $L\mathbf{1} = 0$, $(I - D^{-1}W)\mathbf{1} = 0$ and $(D^{-1}W)\mathbf{1} = \mathbf{1}$. Thus 1 is an eigenvalue of A

Now we turn to an analysis of the transition matrix $A = D^{-1}W$. Note that A is not necessary symmetric. But A is similar to $D^{\frac{1}{2}}AD^{-\frac{1}{2}} = D^{\frac{1}{2}}(D^{-1}W)D^{-\frac{1}{2}} = D^{-\frac{1}{2}}WD^{-\frac{1}{2}}$, which is symmetric. Recall that $\mathcal{L} := I_n - D^{-\frac{1}{2}}WD^{-\frac{1}{2}}$. Recall $L := I_n - D^{-1}W$. Note that L and $\mathcal{L}$ have the same eigenvalue. Also if λ is an eigenvalue of $A = D^{-1}W$ then $1 - \lambda$ is an eigenvalue of L and $\overline{L}$.

LEMMA 4.11. *$\rho(A) = 1$, $\sigma(A) \subset [-1, 1]$ and $\sigma(L) = \sigma(\bar{L}) \subset [0, 2]$.*

PROOF. Recall that $A = D^{-1}W$, $A_{ij} \geq 0$ and $\sum_{j=1}^n A_{ij} = 1$.
$||Ax||_\infty = \max_{1\leq i\leq n}|\sum_{j=1}^n A_{ij}x_j| = \max_{1\leq i\leq n}(\sum_{j=1}^n A_{ij})||x||_\infty = ||x||_\infty$. Thus $||A||_\infty \leq 1$
If $Ax = \lambda x$. then $|\lambda| \leq ||A||_\infty$ and $\rho(A) \leq ||A||_\infty = 1$. On the other hand, 1 is an eigenvalue of A. So $1 \leq \rho(A)$. Hence $\rho(A) = 1$. In particular, $\sigma(A) \subset [-1, 1]$. This implies that $\sigma(L) = \sigma(\bar{L}) \subset [0, 2]$. □

Next, we study the eigenvectors of the normalized graph Laplacian $L = I - A = I - D^{-1}W$, which is the same as those of the transition matrix $A = D^{-1}W$ which is not symmetric. But A is similar to a symmetric matrix $D^{-\frac{1}{2}}WD^{-\frac{1}{2}}$.

Thus we can diagonalize $D^{-\frac{1}{2}}WD^{-\frac{1}{2}}$ by an orthogonal matrix $O \in O(n)$ with

$$D^{-\frac{1}{2}}WD^{-\frac{1}{2}} = O\Lambda O^T$$

where $O^TO = I_n$ and $\Lambda = diag(\mu_1, \cdots, \mu_n)$.
We assume that the eigenvalues of $D^{-\frac{1}{2}}WD^{-\frac{1}{2}}$ are ordered such that $1 = |\mu_1| \geq |\mu_2| \geq \cdots \geq |\mu_n|$.

Also the eigenvalue of L and $\mathcal{L}$ are $\{\lambda_1 = 1 - \mu_1, \lambda_2 = 1 - \mu_2, \cdots, \lambda_n = 1 - \mu_n\}$.

From $D^{-\frac{1}{2}}WD^{-\frac{1}{2}} = O\Lambda O^T$, we have

$$D^{-1}W = D^{-\frac{1}{2}}O\Lambda O^T D^{\frac{1}{2}} = U\Lambda V^T$$

where $U := D^{-\frac{1}{2}}O$ and $V := D^{\frac{1}{2}}O$.

Not that $UV^T = VU^T = I_n$. From $A = U\Lambda V^T$ and $V^TU = I_n$, we have $AU = (U\Lambda V^T)U = U\Lambda$.
Similarly $V^TA = V^T(U\Lambda V^T) = \Lambda V^T$. Thus the column vectors of U and V are the right eigenvectors and left eigenvectors of A. We denote the $i-th$ column of U and V by u_i and v_i respectively. We have the following proposition.

PROPOSITION 4.12. *The right eigenvectors and left eigenvectors of L and A satisfy*
(1) $U^TV = V^TU = I_n$,
(2) Denote $u = \frac{\mathbf{1}}{n}$ *and* $v = \frac{1}{\sum_{l=1}^n d_l}(d_1, \cdots, d_n)^T$. *Then* $Au = u$ *and* $v^TA = v^T$. *We normalize u and v so* $||u||_1 = ||v||_1 = 1$.

PROOF. We have proved everything except $v^TA = v^T$.

This follows from $(v^tA)_j = \sum_i v_iA_{ij} = \frac{\sum_i d_i \frac{W_{ij}}{d_i}}{\sum_l d_l} = \frac{\sum_i W_{ij}}{\sum_l d_l} = \frac{d_j}{\sum_{l=1}^n d_l} = v_j$. □

With the above preparation, we are ready to introduce "Eigenmap". It is first proposed in the paper "M. Belkin and P. Niyogi, "Laplacian Eigenmaps for Dimensionality Reduction and Data Representation," in Neural Computation, vol. 15, no. 6, pp. 1373-1396, 1 June 2003."

DEFINITION 4.13. Let $A = D^{-1}W = U\Lambda V^T$, as above with $\Lambda = diag(\mu_1, \cdots, \mu_n), \mu_1 = 1 \geq \mu_2 \cdots \geq \mu_n$ Let $m+1 \leq n$. The m-dimensional eigenmap is defined as

$$Eig_m(i) = (u_2(i), u_3(i), \cdots, u_{m+1}(i))^T.$$

Note that

$$\begin{pmatrix} Eig_m(1)^T \\ Eig_m(2)^T \\ \vdots \\ Eig_m(n-1)^T \\ Eig_m(n)^T \end{pmatrix} = \begin{pmatrix} u_2(1) & u_3(1) & \cdots & u_{m+1}(1) \\ u_2(2) & u_3(2) & \cdots & u_{m+1}(2) \\ \vdots & & & \\ u_2(n-1) & u_3(n-1) & \cdots & u_{m+1}(n-1) \\ u_2(n) & u_3(n) & \cdots & u_{m+1}(n) \end{pmatrix} = [u_2\ u_3\ \cdots\ u_{m+1}].$$

Now we are ready to introduce "Diffusion map". It is first proposed in the paper "R. Coifman and S. Lafon. Diffusion maps. Appl. Comput. Harmon. Anal., 21:5-30, 2006."

DEFINITION 4.14. Let $A = D^{-1}W = U\Lambda V^T$, with $\Lambda = diag(\mu_1, \cdots, \mu_n), \mu_1 = 1 \geq |\mu_2| \cdots \geq |\mu_n|$, and take a diffusion time $t > 0$. The diffusion map (DM) is

$$\Phi_t : \mathrm{V} \mapsto C^{n-1}$$

, defined by

$$\Phi_t(i) = (\mu_2^t u_2(i), \mu_3^t u_3(i), \cdots, \mu_n^t u_n(i))^T,$$

which is located in a $(n-1)$-dim real Euclidean subspace in C^{n-1}. When all eigenvalues are non-negative, then the embedding is into R^{n-1}.

We did not include $\mu_1 = 1$ and $u_1 = \frac{\mathbf{1}}{\sqrt{n}}$ in the diffusion map.

Note that when $\mu_i < 0$ for some i, then μ_i^t in general is a complex number. Note that when some eigenvalues have multiplicities greater than 1, U and V are not unique. In this case, the DM is defined for a chosen U , and we could use the notation Φ_t^U to emphasize the chosen eigenvectors U. In practice, we often need to consider truncated versions of the diffusion map.

DEFINITION 4.15. Let $A = D^{-1}W = U\Lambda V^T$, as above, and take a diffusion time $t > 0$. Fix $\delta > 0$ as the threshold. The truncated diffusion map (tDM) with time t and threshold δ is a map

$$\Phi_t^\delta : \mathrm{V} \mapsto C^{m(t,\delta)-1}$$

, defined by

$$\Phi_t^\delta(i) = (\mu_2^t u_2(i), \mu_3^t u_3(i), \cdots, \mu_{m(t,\delta)}^t u_{m(t,\delta)}(i))^T,$$

where

$$m(t,\delta) := max\{i : |\mu_i|^t > \delta|\mu_2|^t\}$$

which is located in a $m(t,\delta) - 1$-dim real Euclidean subspace in $C^{m(t,\delta)-1}$. When all eigenvalues are non-negative, then the embedding is into $R^{m(t,\delta)-1}$.

In other words, the map is defined to be the projection of the diffusion map Φ_t onto its first $m - 1$. We truncate those terms that are too small compared with $\delta|\mu_2|^t$. coordinates.

DEFINITION 4.16. The diffusion distance (DD) between $i, j \in \mathrm{V}$, with diffusion time $t > 0$, is defined to be

$$D_t(i,j) = ||\Phi_t(i) - \Phi_t(j)||_{l^2} = \sqrt{\sum_{k=2}^{n} (\mu_k^t u_k(i) - \mu_k^t u_k(j))^2}.$$

The truncated diffusion distance (tDD) is analogously defined as

$$D_t^\delta(i,j) = ||\Phi_t^\delta(i) - \Phi_t^\delta(j)||_{l^2}.$$

REMARK 4.17. Note that $\mu_1 = 1$, $u_1(l) = \frac{1}{\sqrt{n}}$ for $l = 1, \cdots, n$ and $\mu_1^t u_1(i) - \mu_1^t u_k(j) = 0$ for all $1 \leq i, j \leq n$. So

$$D_t(i,j) = \sqrt{\sum_{k=1}^{n} (\mu_k^t u_k(i) - \mu_k^t u_k(j))^2}. \tag{4.1}$$

These distances give us a new metric on the graph G, and hence on our data sets.

Recall that $A = U\Lambda V^T$. We define $A^t = U\Lambda^t V^T$ where

$$\Lambda^t = diag(\mu_1^t, \cdots, \mu_n^t)$$

with $\mu_1 = 1 \geq |\mu_2| \geq |\mu_3| \geq \cdots \geq |\mu_2| \geq |\mu_n|$.

DEFINITION 4.18. The probability cloud of $i \in \mathrm{V}$ at time $t > 0$ is

$$A_i^t = (A_{i1}^t, \cdots, A_{in}^t)^T \in R^n.$$

This is the $i-$th row vectors of A^t

The probability cloud of i describes the behaviour of random walks on G at time t. A_{ij}^t is the probability that i moves to j at time $t > 0$.

DEFINITION 4.19. Let $x \in R^n$. We define

$$||x||_{l^2(D^{-1})} = \sqrt{\sum_{i=1}^{n} \frac{x_i^2}{d_i}}$$

where $D = diag(d_1, d_2, \cdots, d_n)$ is the degree matrix.

We have the following interesting results.

PROPOSITION 4.20. *(1)*

$$D_t(i,j) = ||e_i^T U\Lambda^t - e_j^T U\Lambda^t||_{\mathbf{R^n}}$$

(2)We have

$$D_t(i,j) = ||A_i^t - A_j^t||_{l^2(D^{-1})}$$

.

PROOF. Recall that $\mu_1 = 1 \geq |\mu_2| \geq \cdots |\mu_n|$ and $u_1 = \frac{1}{\sqrt{n}} \begin{pmatrix} 1 \\ 1 \\ \vdots \\ 1 \\ 1 \end{pmatrix}$. Note

$U\Lambda^t = [u_1\, u_2\, \cdots u_n] diag(\mu_1^t, \mu_2^t, \cdots, \mu_n^t] = [\mu_1^t u_1\, \mu_2^t u_2\, \cdots\, \mu_n^t u_n] = [\mu_1^t u_1\, \mu_2^t u_2\, \cdots\, \mu_n^t u_n]$ and

$$e_i^T U\Lambda^t = [\mu_1^t u_1(i)\, \mu_2^t u_2(i)\, \cdots\, \mu_n^t u_n(i)].$$

Thus

$$||e_i^T U\Lambda^t - e_j^T U\Lambda^t||_{\mathbf{R}^n} = D_t(i,j)$$

from equation (4.1).

Since O is an orthogonal matrix, we have $D_t(i,j) = ||e_i^T U\Lambda^t O^T - e_j^T U\Lambda^t O^T||$. Note that $||u|| = ||uD^{\frac{1}{2}}||_{l^2(D^{-1})}$. We have $D_t(i,j) = ||e_i^T U\Lambda^t O^T D^{\frac{1}{2}} - e_j^T U\Lambda^t O^T D^{\frac{1}{2}}||_{l^2(D^{-1})} = ||e_i^T U\Lambda^t V^T - e_j^T U\Lambda^t V^T||_{l^2(D^{-1})} = ||e_i^T A^t - e_j^T A^t||_{l^2(D^{-1})} = ||A_i^t - A_j^t||_{l^2(D^{-1})}$. Note that we use $V = OD^{\frac{1}{2}}$, $V^T = D^{\frac{1}{2}} O^T$, and $A^t = U\Lambda^t V^T$. □

PROPOSITION 4.21. *The tDD satisfies*

$$||\Phi_t(i) - \Phi_t(j)||_{l^2}^2 - \frac{2\delta^2|\mu_2|^{2t}}{d_{min}}(1-\delta_{ij}) \leq D_t^\delta(i,j)^2 \leq ||\Phi_t(i) - \Phi_t(j)||_{l^2}^2,$$

where $d_{min} = min\{d_1, \cdots, d_n\}$.

PROOF. The inequality $D_t^\delta(i,j)^2 \leq ||\Phi_t(i) - \Phi_t(j)||_{l^2}^2$ is immediate from the definition. If $k \geq m(t,\delta)$, we have

$$\begin{aligned} ||\Phi_t(i) - \Phi_t(j)||_{l^2}^2 - D_t^\delta(i,j)^2 &= \sum_{k>m(t,\delta)} |\mu_k|^{2t}(u_k(i) - u_k(j))^2 \\ &\leq \delta^2|\mu_2|^{2t} \sum_{k>m(t,\delta)} (u_k(i) - u_k(j))^2 = \delta^2|\mu_2|^{2t} \sum_{k=1}^{n} u_k(i)^2 + u_k(j)^2 - 2u_k(i)u_k(j) \end{aligned} \tag{4.2}$$

Recall that $U = D^{-\frac{1}{2}}O$ and $O \in O(n)$. We have $UU^T = D^{-\frac{1}{2}}OO^TD^{-\frac{1}{2}} = D^{-1}$. This implies that $\frac{\delta_{ij}}{d_i} = \sum_{k=1}^n U_{ik}U_{kj}^T = \sum_{k=1}^n U_{ik}U_{jk} = \sum_{k=1}^n u_k(i)u_k(j)$, then $\sum_{k=1}^n u_k(i)u_k(i) = \frac{1}{d_i}$, $\sum_{k=1}^n u_k(j)u_k(j) = \frac{1}{d_j}$ and $\sum_{k=1}^n u_k(i)u_k(j) = \frac{1}{d_i}$. Here we have used the convention$U_{ik} = u_k(i)$. Hence,

$$||\Phi_t(i) - \Phi_t(j)||_{l^2}^2 - D_t^\delta(i,j)^2 \leq \delta^2|\mu_2|^{2t}(\frac{1}{d_i} + \frac{1}{d_j}) \leq \frac{2\delta^2|\mu_2|^{2t}}{d_{min}}.$$

where $d_{min} = min\{d_1, \cdots, d_n\}$.

□

We now claim that under suitable conditions, the DD is really a metric defined on the affinity graph.

PROPOSITION 4.22. *If* $\mu_i \neq 0$ *for all* i *, then* D_t *is a distance function on the graph.*

PROOF. It is clear from the definition of D_t that $D_t : V \times V \mapsto \mathbf{R}^+$ Note D_t is a metric on V iff
(1) $D_t(i,j) = D_t(j,i)$
(2) $D_t(i,j) = 0$ iff $i = j$
(3) $D_t(i,k) \leq D_t(i,j) + D_t(j,k)$.

(1) and (3) are obvious from $D_t(i,j) = ||e_i^T U\Lambda^t - e_j^T U\Lambda^t||_{\mathbf{R^n}}$.

If $D_t(i,j) = 0$ then $e_i^T U\Lambda^t = e_j^T U\Lambda^t$ and $[\mu_1^t u_1(i)\ \mu_2^t u_2(i)\ \cdots\ \mu_n^t u_n(i)] = [\mu_1^t u_1(j)\ \mu_2^t u_2(j)\ \cdots\ \mu_n^t u_n(j)]$. Since $|\mu_l| > 0$ for $1 \leq l \leq n$, we have $[u_1(i)\ u_2(i)\ \cdots\ u_n(i)] = [u_1(j)\ u_2(i)\ \cdots\ u_n(j)]$. If $i \neq j$ then the $i-th$ row vector and the $j-th$ row vector in the matrix U are the same and $det(U) = 0$. It contradicts with the fact that U is a nonsingular(invertible) matrix. □

6. Examples

EXAMPLE 4.23. Let $V = \{1, \cdots, n\}$, and put undirected edges between k and $k+1$ for all k, as well as between n and 1. In this example we will consider the graph whose affinity matrix is the adjacent matrix or, equivalently, an affinity graph with the affinity function $\omega(i,j) = 1$ for all $(i,j) \in E$. Then

$$W = \begin{bmatrix} 0 & 1 & 0 & \dots & 0 & 1 \\ 1 & 0 & 1 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 1 & 0 & 0 & \dots & 1 & 0 \end{bmatrix}, \quad D = \begin{bmatrix} 2 & 0 & 0 & \dots & 0 & 0 \\ 0 & 2 & 0 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & 0 & 2 \end{bmatrix} = 2I_n.$$

and

$$A = D^{-1}W = \frac{1}{2}W.$$

Let $v = \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{bmatrix}$. Then $Av = \frac{1}{2}\begin{bmatrix} v_n + v_2 \\ v_1 + v_3 \\ v_2 + v_4 \\ \vdots \\ v_{n-1} + v_1 \end{bmatrix}$. For simplicity, we define $v_0 = v_n$ and $v_{n+1} = v_1$. Thus $(Av)_k = \frac{1}{2}(v_{k-1} + v_{k+1})$

First, let us look at the case when $n = 3$ where $W = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix}$ and $A = \frac{1}{2}\begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix}$. The eigenvalues of A and W are the same. $det(\lambda I - A) = det\begin{bmatrix} \lambda & -\frac{1}{2} & -\frac{1}{2} \\ -\frac{1}{2} & \lambda & -\frac{1}{2} \\ -\frac{1}{2} & -\frac{1}{2} & \lambda \end{bmatrix} = \lambda^3 - \frac{1}{4} - \frac{3}{4}\lambda$. Solving $\lambda^3 - \frac{1}{4} - \frac{3}{4}\lambda = 0$, we have $1/4(\lambda - 1)(2\lambda + 1)^2 = 0$ and $\lambda = 1, -\frac{1}{2}$. Note that $\lambda = \cos(0), \cos(2\pi/3)$.

Next, we discuss the general case. We claim that for each $l = 1, \cdots, n$, the vectors $\phi_l = \begin{bmatrix} e^{i\frac{2\pi(l-1)}{n}} \\ e^{i\frac{4\pi(l-1)}{n}} \\ e^{i\frac{6\pi(l-1)}{n}} \\ \vdots \\ e^{i\cdot 2\pi(l-1)} \end{bmatrix}$ is a complex eigenvectors of A with eigenvalues $\mu_l = \cos(\frac{2\pi(l-1)}{n})$, i.e. $A\phi_l =$

$\mu_l\phi_l$. Note that $\phi_l(k) = e^{i\frac{2\pi(l-1)k}{n}}$ for $k = 1, \cdots, n$. This can be verified by computing

$$A\phi_l(k) = \frac{1}{2}(\phi_l(k-1) + \phi_l(k+1)) = \frac{1}{2}(e^{i\frac{2\pi(l-1)(k-1)}{n}} + e^{i\frac{2\pi(l-1)(k+1)}{n}})$$
$$= \frac{1}{2}(e^{i\frac{2\pi(l-1)k}{n}}(e^{i\frac{2\pi(l-1)(-1)}{n}} + e^{i\frac{2\pi(l-1)}{n}})) = \cos(\frac{2\pi(l-1)}{n})e^{i\frac{2\pi(l-1)k}{n}} = \mu_l\phi_l(k).$$

Now $A\phi_l = \mu_l\phi_l$ and $A\phi_l = \mu_l\bar{\phi}_l$. We have $A(\frac{\phi_l+\bar{\phi}_l}{2}) = \mu_l(\frac{\phi_l+\bar{\phi}_l}{2})$ and $A(\frac{\phi_l-\bar{\phi}_l}{2i}) = \mu_l(\frac{\phi_l-\bar{\phi}_l}{2i})$.

Since $e^{-i\frac{2\pi(l-1)k}{n}} = e^{i\frac{2\pi(n-l+1)k}{n}}$, we have $\bar{\phi}_l(k) = \phi_{n-l+2}(k)$ and $\mu_l = \mu_{n-l+2}$. When n is an even value, that is, $n = 2m$, the $(m+1)-$th eigenvalue is $\mu_{m+1} = \cos(\frac{2\pi m}{2m}) = \cos(\pi) = -1$ and its associated eigenvector is composed of alternating 1 and -1 ($\phi_{m+1}(k) = e^{i\frac{2\pi mk}{2m}} = e^{ik\pi} = (-1)^k$). Also, we have $\mu_1 = 1$, $\phi_1(k) = e^{i\frac{2\pi 0\cdot k}{n}} = 1$ and $\phi_1 = \mathbf{1}$. Thus ϕ_1 and ϕ_{m+1} are real eigenvectors.

When $n = 2m$, we have $m+1$ eigenvalues, that is, $\mu_1 = 1$ with multiplicity $\mu_1 = 1$ with multiplicity 1, $\mu_l = \cos(2\pi(l-1)/n)$ with multiplicity 2, $l = 2, \cdots, m$ and $\mu_{m+1} = 1$ with multiplicity 1.

When $n = 2m+1$ is odd, we have $m+1$ eigenvalues, that is, $\mu_1 = 1$ with multiplicity 1 and $\mu_l = \cos(2\pi(l-1)/n)$ with multiplicity 2, $l = 2, \cdots, m+1$.

In particular, $\mu_2 = \cos(\frac{2\pi}{n})$, $\phi_2 == \begin{bmatrix} e^{i\frac{2\pi}{n}} \\ e^{i\frac{4\pi}{n}} \\ e^{i\frac{6\pi}{n}} \\ \vdots \\ e^{i\cdot 2\pi)} \end{bmatrix} = \begin{bmatrix} \cos(\frac{2\pi}{n}) \\ \cos(\frac{4\pi}{n}) \\ \vdots \\ 1 \end{bmatrix} + i \begin{bmatrix} \sin(\frac{2\pi}{n}) \\ \sin(\frac{4\pi}{n}) \\ \vdots \\ 0 \end{bmatrix} = u_2 + iu_3.$

where u_2 and u_3 are the second and third eigenvectors of A. Thus

$$[u_2, u_3] = \begin{bmatrix} \cos(\frac{2\pi}{n}), & \sin(\frac{2\pi}{n}) \\ \cos(\frac{4\pi}{n}), & \sin(\frac{4\pi}{n}) \\ \vdots & \\ 1, & 0 \end{bmatrix}.$$

Thus the embedding of the second and third eigenfunction of A is just the dividing of the unit circle into a regular n-gons.

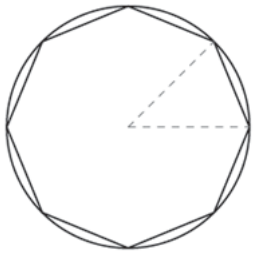


Figure 1: The eigenmap embedding when $n = 8$.

Bibliography

[F] Karl Pearson F.R.S., *Liii. on lines and planes of closest fit to systems of points in space*, The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science **2** (1901), no. 11, 559–572, available at https://doi.org/10.1080/14786440109462720.