
Laplacian-Based Dimensionality Reduction Including Spectral Clustering, Laplacian Eigenmap, Locality Preserving Projection, Graph Embedding, and Diffusion Map: Tutorial and Survey

Benyamin Ghojogh

BGHOJOGH@UWATERLOO.CA

Department of Electrical and Computer Engineering,
Machine Learning Laboratory, University of Waterloo, Waterloo, ON, Canada

Ali Ghodsi

ALI.GHODSI@UWATERLOO.CA

Department of Statistics and Actuarial Science & David R. Cheriton School of Computer Science,
Data Analytics Laboratory, University of Waterloo, Waterloo, ON, Canada

Fakhri Karray

KARRAY@UWATERLOO.CA

Department of Electrical and Computer Engineering,
Centre for Pattern Analysis and Machine Intelligence, University of Waterloo, Waterloo, ON, Canada

Mark Crowley

MCROWLEY@UWATERLOO.CA

Department of Electrical and Computer Engineering,
Machine Learning Laboratory, University of Waterloo, Waterloo, ON, Canada

Abstract

This is a tutorial and survey paper for **nonlinear dimensionality and feature extraction methods** which are based on the Laplacian of graph of data. We first introduce adjacency matrix, definition of Laplacian matrix, and the interpretation of Laplacian. Then, we cover the cuts of graph and spectral clustering which applies clustering in a subspace of data. Different optimization variants of Laplacian eigenmap and its **out-of-sample extension** are explained. Thereafter, we introduce the locality preserving projection and its kernel variant as **linear special cases** of Laplacian eigenmap. Versions of graph embedding are then explained which are generalized versions of Laplacian eigenmap and locality preserving projection. Finally, diffusion map is introduced which is a method based on Laplacian of data and random walks on the data graph.

extraction or data visualization (Ghojogh et al., 2019d). Dimensionality reduction methods belong to **three broad categories, which are spectral, probabilistic, and neural network-based methods** (Ghojogh, 2021). The spectral methods deal with the graph and geometry of data and usually reduce to an **eigenvalue or generalized eigenvalue problem** (Ghojogh et al., 2019a). A sub-family of the spectral dimensionality reduction methods is based on the Laplacian of graph of data (Merris, 1994). The importance of Laplacian was probably noticed more when researchers gradually proposed spectral clustering methods which made use of Laplacian (Weiss, 1999; Ng et al., 2001). **Spectral clustering transforms data onto a low-dimensional subspace to hopefully separate the clusters better**. Then, in that **discriminating** subspace, a simple clustering algorithm is applied. Researchers adopted the idea of spectral clustering to use it for the goal of dimensionality reduction. The result was the **Laplacian eigenmap** (Belkin & Niyogi, 2001; 2003). In the meantime, **out-of-sample extension techniques were also proposed for spectral clustering and Laplacian eigenmap** (Bengio et al., 2003a; Gisbrecht et al., 2012; Bunte et al., 2012; Gisbrecht et al., 2015). Later on, people decided to **linearize the nonlinear Laplacian eigenmap using linear projection**. Hence, Locality Preserving Projection (LPP) was proposed (He & Niyogi, 2004). A kernel version of LPP was also proposed (Cheng et al., 2005; Li et al., 2008) to apply LPP in the feature space rather than input space. Afterwards, researchers

1. Introduction

Nonlinear dimensionality reduction methods find the **manifold of data** in a nonlinear way for the sake of feature

found out that many spectral methods belong to a family of methods so they tried to propose a **generalized spectral method using the Laplacian matrix**. Therefore, **graph embedding**, with its various variants, was proposed (Yan et al., 2005; 2006). It was around that time that another method, which made use of Laplacian and kernel of data, was proposed which **revealed the underlying nonlinear manifold of data in a diffusion process**. This method is known as the diffusion map (Lafon, 2004; Coifman et al., 2005; Coifman & Lafon, 2006). In this paper, we explain the theory and intuitions of these Laplacian-based methods in detail. Note that although Locally Linear Embedding (LLE) can also be considered as a Laplacian-based method, we have provided a separate tutorial for it and its variants (Ghojogh et al., 2020a). Note that there **exist some surveys**, such as (Wiskott & Schönfeld, 2019), on the Laplacian-based methods.

The remainder of this paper is as follows. We introduce the adjacency matrix, Laplacian matrix, and their interpretations in Section 2. Sections 3 introduces the **cuts and clustering** for two and multiple number of classes. Laplacian eigenmap and its out-of-sample extension are explained in Section 4. We introduce LPP and kernel LPP in Section 5. Direct, linearized, and kernelized graph embedding are also detailed in Section 6. We introduce the **diffusion map** in Section 7. Finally, Section 8 concludes the paper.

Required Background for the Reader

This paper assumes that the reader has general knowledge of calculus, linear algebra, and basics of optimization.

2. Laplacian Matrix and its Interpretation

Most of the methods in this tutorial use the **Laplacian matrix of data graph**. In this section, we introduce the Laplacian matrix and its interpretation.

2.1. Adjacency Matrix

Assume we have a dataset $\{\mathbf{x}_i\}_{i=1}^n$. In spectral clustering and related algorithms, we first construct a nearest neighbors graph. There are two versions of constructing the nearest neighbors graph (Belkin & Niyogi, 2001):

1. graph using ϵ -neighborhoods: nodes i and j are connected if $\|\mathbf{x}_i - \mathbf{x}_j\|_2^2 \leq \epsilon$
2. k -Nearest Neighbors (k NN) graph: node j is connected to node i are connected if \mathbf{x}_j is among the k nearest neighbors of \mathbf{x}_i

Using the nearest neighbors graph, we construct an adjacency matrix, also called **weight matrix or affinity matrix**, which is a weighted version of nearest neighbors graph. The weights of adjacency matrix represent the **similarity of nodes**. The more similar two nodes are, the larger their

weight is. Let $\mathbf{W} \in \mathbb{R}^{n \times n}$ denote the adjacency or weight matrix whose (i, j) -th element is (Ng et al., 2001):

$$\mathbf{W}(i, j) := \begin{cases} w_{ij} & \text{if } \mathbf{x}_j \in \text{nearest neighbors of } \mathbf{x}_i, \\ w_{ij} = 0 & \text{if } i = j, \\ w_{ij} = 0 & \text{if } \mathbf{x}_j \notin \text{nearest neighbors of } \mathbf{x}_i. \end{cases} \quad (1)$$

Note that one can **choose a fully connected nearest neighbors graph** where all points are among the neighbors of each other. In that case, the third above condition can be omitted (Ng et al., 2001).

An example of the weight function for the edge between \mathbf{x}_i and \mathbf{x}_j is the Radial Basis Function (RBF) or heat kernel (Belkin & Niyogi, 2001):

$$w_{ij} = \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|_2^2}{2\sigma^2}\right), \quad (2)$$

whose parameter σ^2 can be set to 0.5 or one for simplicity. This weight function gets larger, or closer to one, when \mathbf{x}_i and \mathbf{x}_j are closer to each other.

Another approach for choosing w_{ij} is the simple-minded approach (Belkin & Niyogi, 2001):

$$w_{ij} := \begin{cases} 1 & \text{if } i \text{ and } j \text{ are connected} \\ & \text{in the nearest neighbors graph,} \\ 0 & \text{otherwise.} \end{cases} \quad (3)$$

2.2. Laplacian Matrix

We define the Laplacian matrix of the adjacency matrix, denoted by $\mathbf{L} \in \mathbb{R}^{n \times n}$, as (Merris, 1994):

$$\mathbf{L} := \mathbf{D} - \mathbf{W}, \quad (4)$$

where $\mathbb{R}^{n \times n} \ni \mathbf{D} = \text{diag}([d_1, \dots, d_n]^\top)$ is the degree matrix with diagonal elements as:

$$d_i = \sum_{j=1}^n w_{ij}, \quad (5)$$

which is the summation of its corresponding row. Therefore, the row summation of Laplacian matrix is zero:

$$\sum_{j=1}^n L_{ij} = 0. \quad (6)$$

It is noteworthy that there exist some other variants of **Laplacian matrix** such as (Weiss, 1999; Ng et al., 2001):

$$\mathbf{L} := \mathbf{D}^{-\alpha} \mathbf{W} \mathbf{D}^{-\alpha}, \quad (7)$$

where $\alpha \geq 0$ is a parameter. A common example is $\alpha = 0.5$. Some papers have used (Weiss, 1999; Ng et al., 2001):

$$\mathbf{L} := \mathbf{D}^{-1/2} \mathbf{W} \mathbf{D}^{-1/2}, \quad (8)$$

rather than Eq. (4) for the definition of Laplacian. According to (Weiss, 1999), Eq. (8) leads to having the (i, j) -th element of Laplacian as:

$$L(i, j) := \frac{W(i, j)}{\sqrt{D(i, i) D(j, j)}}. \quad (9)$$

Note that Eq. (9) is very similar to one of the techniques for **kernel normalization** (Ah-Pine, 2010). Also, we know that adjacency matrix is a kernel, such as RBF kernel in Eq. (2). Hence, in the literature, sometimes Laplacian is referred to as the **normalized kernel matrix** (e.g., see the literature of diffusion map (Coifman & Lafon, 2006; Nadler et al., 2006b)).

Note that Eq. (7) has an opposite effect compared to Eq. (4) because the matrix W is not negated in it. According to the spectral graph theory (Chung, 1997), we can use $(I - L)$ or Eqs. (8) and (9) rather than L in optimization problems if we change minimization to maximization.

It is also good to note that Eq. (7) is very similar to the **symmetric normalized Laplacian matrix** defined as (Chung, 1997, p. 2):

$$L \leftarrow D^{-(1/2)} L D^{-(1/2)}. \quad (10)$$

2.3. Interpretation of Laplacian

Let B denote the incidence matrix of the data graph where the rows and columns of B correspond to edges and vertices of the graph, respectively. The elements of this incidence matrix **determine if a vertex is connected to an edge (it is zero if they are not connected)**. Therefore, this matrix shows if every two points (vertices) are connected or not.

Lemma 1 ((Strang, 2014, Section 5.6)). *The incidence matrix of a directed graph can be seen as a difference matrix (of graph nodes).*

Corollary 1. *As gradient, denoted by ∇ , is a measure of small difference, we can say from Lemma 1 that the incidence matrix is analogous to gradient:*

$$B \propto \nabla. \quad (11)$$

Lemma 2 ((Kelner, 2007, Lemma 10)). *We can factorize the Laplacian matrix of a graph using its incidence matrix:*

$$L = B^\top B. \quad (12)$$

This also explains why the Laplacian matrix is positive semi-definite.

Corollary 2. *From Eqs. (11) and (12), we can say that:*

$$L = \nabla^\top \nabla. \quad (13)$$

This operator, which is the inner produce of gradients, is also referred to as the Laplace operator.

Hence, the *Laplace operator* or *Laplacian* is a scalar operator which is the inner product of two gradient vectors:

$$\begin{aligned} L &:= \nabla \cdot \nabla = \nabla^2 = \left[\frac{\partial}{\partial x_1}, \dots, \frac{\partial}{\partial x_d} \right] \begin{bmatrix} \frac{\partial}{\partial x_1} \\ \vdots \\ \frac{\partial}{\partial x_d} \end{bmatrix} \\ &= \frac{\partial^2}{\partial x_1 \partial x_1} + \dots + \frac{\partial^2}{\partial x_d \partial x_d} = \sum_{i=1}^d \frac{\partial^2}{\partial x_i \partial x_i}. \end{aligned} \quad (14)$$

This shows that Laplacian is the summation of diagonal of the Hessian matrix (the Jacobian of the gradient). Therefore, we can say that Laplacian is the **trace of the Hessian matrix**. Thus, it can also be seen as **the summation of eigenvalues of the Hessian matrix**. Laplacian captures **the amount of divergence from a point**. Therefore, Laplacian measures whether the neighboring points of a point are larger or smaller than it or are on a line. In other words, it **shows how much linear the neighborhood of a point is**. If a point is a local minimum/maximum, its Laplacian is positive/negative because its diverging neighbors are upper/lower than it. In case, the point is on linear function, its Laplacian is zero. Therefore, **Laplacian shows how bumpy the neighborhood of a point is**¹.

The above explanation can be stated in mathematics as divergence:

$$L \stackrel{(13)}{=} \nabla^\top \nabla = -\text{div} \nabla, \quad (15)$$

where div denotes the divergence operator.

From the above discussion, we see that Laplacian measures how bumpy the neighbors of every point are with respect to it in the nearest neighbors graph of data. Therefore, using Laplacian in spectral clustering and related algorithms is because Laplacian is a measure of relation of neighbor points; hence, it can be used for **inserting the information of structure of data into the formulation of algorithm**. Similar to this interpretation, as well as some connections to heat equations, can be seen in (Belkin & Niyogi, 2005).

2.4. Eigenvalues of Laplacian Matrix

It is well-known in linear algebra and graph theory that if a graph has k disjoint connected parts, its Laplacian matrix has k zero eigenvalues (see (Anderson Jr & Morley, 1985, Theorem 1), (Marsden, 2013, Theorem 3.10), and (Polito & Perona, 2002; Ahmadizadeh et al., 2017)). We assume that the nearest neighbors graph of data is **a connected graph**. L has one zero eigenvalue whose eigenvector is $\mathbf{1} = [1, 1, \dots, 1]^\top$. In spectral clustering and related methods, we ignore the eigenvector corresponding to the zero eigenvalue.

¹We thank Ming Miao for explaining this interpretation of Laplace operator to the authors.

2.5. Convergence of Laplacian

The Laplacian-based methods usually use either the Laplacian matrix in the Euclidean space or the Laplace–Beltrami operator on the Riemannian manifold (Hein et al., 2005). It is shown in several works, such as (Burago et al., 2015), (Trillos et al., 2020), and (Dunson et al., 2021), that the eigenfunctions and eigenvalues of the Laplace–Beltrami operator can be approximated by eigenvalues and eigenvectors of Laplacian of a weight matrix, respectively. Moreover, it is shown in (Hein et al., 2005; 2007) that the kernel-based averaging operators can approximate the weighted Laplacian point-wise, for epsilon-graph (i.e., not fully-connected) weight matrices.

3. Spectral Clustering

Spectral clustering (Weiss, 1999; Ng et al., 2001) is a clustering method whose solution is reduced to an eigenvalue problem. Refer to (Weiss, 1999) for a survey on initial work on spectral clustering approaches. A list of some initial works developing to result in the modern spectral clustering is (Scott & Longuet-Higgins, 1990; Costeira & Kanade, 1995; Perona & Freeman, 1998; Shi & Malik, 1997; 2000). The general idea of spectral clustering is as follows. First we transform data from its input space to another low-dimensional subspace so that the clusters of data become more separated hopefully. Then, we apply a simple clustering algorithm in that discriminating subspace. In the following, we explain the theory of spectral clustering.

3.1. Spectral Clustering

3.1.1. ADJACENCY MATRIX

As was explained in Section 2, we compute the adjacency matrix and its Laplacian. These matrices are used in later sections for the spectral clustering algorithm.

3.1.2. THE CUT

We denote the graph of dataset by:

$$\mathcal{G} := (\mathcal{X}, \mathcal{W}), \quad (16)$$

where the data indices $\mathcal{X} := \{i\}_{i=1}^n$ are the vertices of graph and the weights $\mathcal{W} := \{w_{ij}\}_{i,j=1}^n$ are the weights for edges of graph.

Let the data points be divided into two disjoint clusters \mathcal{A} and \mathcal{A}' :

$$\mathcal{X} = \mathcal{A} \cup \mathcal{A}', \quad \mathcal{A} \cap \mathcal{A}' = \emptyset. \quad (17)$$

We define the cut between two sub-graphs as (Shi & Malik, 1997; 2000):

$$\text{cut}(\mathcal{A}, \mathcal{A}') := \sum_{i \in \mathcal{A}} \sum_{j \in \mathcal{A}'} w_{ij}. \quad (18)$$

As the summations can be exchanged, we have:

$$\text{cut}(\mathcal{A}, \mathcal{A}') = \text{cut}(\mathcal{A}', \mathcal{A}). \quad (19)$$

3.1.3. OPTIMIZATION OF SPECTRAL CLUSTERING

If the two clusters or the sub-graphs are very different or far from each other, the summation of weights between their points is small so their cut is small. Hence, for finding the separated clusters, we can find the sub-graphs which minimize the cut:

$$\underset{\mathcal{A}, \mathcal{A}'}{\text{minimize}} \quad \text{cut}(\mathcal{A}, \mathcal{A}') = \underset{\mathcal{A}, \mathcal{A}'}{\text{minimize}} \quad \sum_{i \in \mathcal{A}} \sum_{j \in \mathcal{A}'} w_{ij}. \quad (20)$$

However, this minimization is sensitive to outliers. This is because an outlier point can be considered as a cluster itself in this optimization problem as it is very far from other points. To overcome this issue, we use a normalized version of cut, named ratio cut, which is defined as (Shi & Malik, 1997; 2000):

$$\begin{aligned} \text{ratioCut}(\mathcal{A}, \mathcal{A}') &= \frac{\text{cut}(\mathcal{A}, \mathcal{A}')}{|\mathcal{A}|} + \frac{\text{cut}(\mathcal{A}', \mathcal{A})}{|\mathcal{A}'|} \\ &\stackrel{(19)}{=} \frac{\text{cut}(\mathcal{A}, \mathcal{A}')}{|\mathcal{A}|} + \frac{\text{cut}(\mathcal{A}, \mathcal{A}')}{|\mathcal{A}'|}, \end{aligned} \quad (21)$$

where $|\mathcal{A}|$ denotes the cardinality of set \mathcal{A} . Minimizing the ratio cut rather than cut resolves the problem with outliers because the cluster of outlier(s) is a small cluster and its cardinality is small' hence, the denominator of one of the fractions in ratio cut gets very small and the ratio cut becomes large. Therefore, we minimize the ratio cut for clustering:

$$\underset{\mathcal{A}, \mathcal{A}'}{\text{minimize}} \quad \text{ratioCut}(\mathcal{A}, \mathcal{A}'). \quad (22)$$

Proposition 1 ((Shi & Malik, 1997)). *Minimizing the ratio cut, i.e. Eq. (22), is equivalent to:*

$$\underset{\mathbf{f}}{\text{minimize}} \quad \sum_{i=1}^n \sum_{j=1}^n w_{ij} (f_i - f_j)^2, \quad (23)$$

where $\mathbf{f} := [f_1, \dots, f_n]^\top \in \mathbb{R}^n$ and:

$$f_i := \begin{cases} \sqrt{\frac{|\mathcal{A}'|}{|\mathcal{A}|}} & \text{if } i \in \mathcal{A}, \\ -\sqrt{\frac{|\mathcal{A}|}{|\mathcal{A}'|}} & \text{if } i \in \mathcal{A}'. \end{cases} \quad (24)$$

Proof.

$$\begin{aligned}
\sum_{i=1}^n \sum_{j=1}^n w_{ij} (f_i - f_j)^2 &= \sum_{i \in \mathcal{A}} \sum_{j \in \mathcal{A}} w_{ij} \underbrace{\left(\sqrt{\frac{|\mathcal{A}'|}{|\mathcal{A}|}} - \sqrt{\frac{|\mathcal{A}'|}{|\mathcal{A}|}} \right)^2}_{=0} \\
&+ \sum_{i \in \mathcal{A}'} \sum_{j \in \mathcal{A}'} w_{ij} \underbrace{\left(-\sqrt{\frac{|\mathcal{A}|}{|\mathcal{A}'|}} + \sqrt{\frac{|\mathcal{A}|}{|\mathcal{A}'|}} \right)^2}_{=0} \\
&+ \sum_{i \in \mathcal{A}} \sum_{j \in \mathcal{A}'} w_{ij} \left(\sqrt{\frac{|\mathcal{A}'|}{|\mathcal{A}|}} + \sqrt{\frac{|\mathcal{A}'|}{|\mathcal{A}|}} \right)^2 \\
&+ \sum_{i \in \mathcal{A}'} \sum_{j \in \mathcal{A}} w_{ij} \left(-\sqrt{\frac{|\mathcal{A}|}{|\mathcal{A}'|}} - \sqrt{\frac{|\mathcal{A}|}{|\mathcal{A}'|}} \right)^2 \\
&= \sum_{i \in \mathcal{A}} \sum_{j \in \mathcal{A}'} w_{ij} \left(\sqrt{\frac{|\mathcal{A}'|}{|\mathcal{A}|}} + \sqrt{\frac{|\mathcal{A}'|}{|\mathcal{A}|}} \right)^2 \\
&+ \sum_{i \in \mathcal{A}'} \sum_{j \in \mathcal{A}} w_{ij} \left(\sqrt{\frac{|\mathcal{A}|}{|\mathcal{A}'|}} + \sqrt{\frac{|\mathcal{A}|}{|\mathcal{A}'|}} \right)^2 \\
&= \sum_{i \in \mathcal{A}} \sum_{j \in \mathcal{A}'} w_{ij} \left(\frac{|\mathcal{A}'|}{|\mathcal{A}|} + \frac{|\mathcal{A}|}{|\mathcal{A}'|} + 2 \right) \\
&+ \sum_{i \in \mathcal{A}'} \sum_{j \in \mathcal{A}} w_{ij} \left(\frac{|\mathcal{A}|}{|\mathcal{A}'|} + \frac{|\mathcal{A}'|}{|\mathcal{A}|} + 2 \right) \\
&= \left(\frac{|\mathcal{A}|}{|\mathcal{A}'|} + \frac{|\mathcal{A}'|}{|\mathcal{A}|} + 2 \right) \left(\sum_{i \in \mathcal{A}} \sum_{j \in \mathcal{A}'} w_{ij} + \sum_{i \in \mathcal{A}'} \sum_{j \in \mathcal{A}} w_{ij} \right) \\
&\stackrel{(18)}{=} \left(\frac{|\mathcal{A}|}{|\mathcal{A}'|} + \frac{|\mathcal{A}'|}{|\mathcal{A}|} + 2 \right) (\text{cut}(\mathcal{A}, \mathcal{A}') + \text{cut}(\mathcal{A}', \mathcal{A})) \\
&\stackrel{(19)}{=} 2 \left(\frac{|\mathcal{A}|}{|\mathcal{A}'|} + \frac{|\mathcal{A}'|}{|\mathcal{A}|} + 2 \right) \text{cut}(\mathcal{A}, \mathcal{A}') \\
&= 2 \left(\frac{|\mathcal{A}|}{|\mathcal{A}'|} + \frac{|\mathcal{A}'|}{|\mathcal{A}|} + \underbrace{\frac{|\mathcal{A}|}{|\mathcal{A}|} + \frac{|\mathcal{A}'|}{|\mathcal{A}'|}}_{=2} \right) \text{cut}(\mathcal{A}, \mathcal{A}') \\
&= 2 \left(\frac{|\mathcal{A}| + |\mathcal{A}'|}{|\mathcal{A}'|} + \frac{|\mathcal{A}'| + |\mathcal{A}|}{|\mathcal{A}|} \right) \text{cut}(\mathcal{A}, \mathcal{A}') \\
&= 2 \left(\frac{n}{|\mathcal{A}'|} + \frac{n}{|\mathcal{A}|} \right) \text{cut}(\mathcal{A}, \mathcal{A}') \\
&= 2n \left(\frac{\text{cut}(\mathcal{A}, \mathcal{A}')}{|\mathcal{A}'|} + \frac{\text{cut}(\mathcal{A}, \mathcal{A}')}{|\mathcal{A}|} \right) \stackrel{(21)}{=} 2n \text{ratioCut}(\mathcal{A}, \mathcal{A}').
\end{aligned}$$

Q.E.D. \square

Proposition 2 ((Belkin & Niyogi, 2001)). *We have:*

$$\frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n w_{ij} (f_i - f_j)^2 = \mathbf{f}^\top \mathbf{L} \mathbf{f}, \quad (25)$$

where $\mathbf{L} \in \mathbb{R}^{n \times n}$ is the Laplacian matrix of the weight matrix.

Proof.

$$\begin{aligned}
\mathbf{f}^\top \mathbf{L} \mathbf{f} &\stackrel{(4)}{=} \mathbf{f}^\top (\mathbf{D} - \mathbf{W}) \mathbf{f} = \mathbf{f}^\top \mathbf{D} \mathbf{f} - \mathbf{f}^\top \mathbf{W} \mathbf{f} \\
&\stackrel{(a)}{=} \sum_{i=1}^n d_i f_i^2 - \sum_{i=1}^n \sum_{j=1}^n w_{ij} f_i f_j \\
&= \frac{1}{2} \left(2 \sum_{i=1}^n d_i f_i^2 - 2 \sum_{i=1}^n \sum_{j=1}^n w_{ij} f_i f_j \right) \\
&\stackrel{(b)}{=} \frac{1}{2} \left(\sum_{i=1}^n d_i f_i^2 + \sum_{j=1}^n d_j f_j^2 - 2 \sum_{i=1}^n \sum_{j=1}^n w_{ij} f_i f_j \right) \\
&\stackrel{(5)}{=} \frac{1}{2} \left(\sum_{i=1}^n f_i^2 \sum_{j=1}^n w_{ij} + \sum_{j=1}^n f_j^2 \sum_{i=1}^n w_{ji} \right. \\
&\quad \left. - 2 \sum_{i=1}^n \sum_{j=1}^n w_{ij} f_i f_j \right) \\
&\stackrel{(c)}{=} \frac{1}{2} \left(\sum_{i=1}^n \sum_{j=1}^n f_i^2 w_{ij} + \sum_{i=1}^n \sum_{j=1}^n f_j^2 w_{ij} \right. \\
&\quad \left. - 2 \sum_{i=1}^n \sum_{j=1}^n w_{ij} f_i f_j \right) \\
&= \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n w_{ij} (f_i - f_j)^2,
\end{aligned}$$

where (a) notices that \mathbf{D} is a diagonal matrix, (b) is because we can replace the dummy index i with j , and (c) notices that $w_{ij} = w_{ji}$ as we usually use symmetric similarity measures. Q.E.D. \square

Therefore, we can minimize $\mathbf{f}^\top \mathbf{L} \mathbf{f}$ for spectral clustering. Hence, the optimization in spectral clustering is:

$$\begin{aligned}
&\underset{\mathbf{f}}{\text{minimize}} \quad \mathbf{f}^\top \mathbf{L} \mathbf{f} \\
&\text{subject to} \quad f_i := \begin{cases} \sqrt{\frac{|\mathcal{A}'|}{|\mathcal{A}|}} & \text{if } i \in \mathcal{A}, \\ -\sqrt{\frac{|\mathcal{A}|}{|\mathcal{A}'|}} & \text{if } i \in \mathcal{A}'. \end{cases}, \forall i. \quad (26)
\end{aligned}$$

This optimization problem is very complicated to solve. We can relax the constraint to having orthonormal vector \mathbf{f} . Recall that according to Eq. (24) or the constraint, if $f_i > 0$, it is in the first cluster or \mathcal{A} and if $f_i < 0$, it is in the second cluster or \mathcal{A}' . Here, in the modified constraint, we keep the sign characteristic of Eq. (24) as:

$$f_i := \begin{cases} > 0 & \text{if } i \in \mathcal{A}, \\ < 0 & \text{if } i \in \mathcal{A}'. \end{cases} \quad (27)$$

This modification satisfies $\sum_{i=1}^n (f_i \times f_i) = 1$. Hence, the optimization is relaxed to:

$$\begin{aligned}
&\underset{\mathbf{f}}{\text{minimize}} \quad \mathbf{f}^\top \mathbf{L} \mathbf{f} \\
&\text{subject to} \quad \mathbf{f}^\top \mathbf{f} = 1. \quad (28)
\end{aligned}$$

3.1.4. SOLUTION TO OPTIMIZATION

The Lagrangian of Eq. (28) is (Boyd et al., 2004):

$$\mathcal{L} = \mathbf{f}^\top \mathbf{L} \mathbf{f} - \lambda(\mathbf{f}^\top \mathbf{f} - 1),$$

with λ as the dual variable. Setting the derivative of Lagrangian to zero gives:

$$\mathbb{R}^n \ni \frac{\partial \mathcal{L}}{\partial \mathbf{f}} = 2\mathbf{L}\mathbf{f} - 2\lambda \mathbf{f} \stackrel{\text{set}}{=} \mathbf{0} \implies \mathbf{L}\mathbf{f} = \lambda \mathbf{f}, \quad (29)$$

which is the eigenvalue problem for the Laplacian matrix \mathbf{L} (Ghojogh et al., 2019a).

As Eq. (28) is a minimization problem, the eigenvectors should be sorted from the corresponding smallest to largest eigenvalues. Moreover, note that \mathbf{L} is the Laplacian matrix for the weights \mathbf{W} . Hence, according to Section 2.4, after sorting the eigenvectors from smallest to largest eigenvalues, we ignore the first eigenvector having zero eigenvalue and take the smallest eigenvector of \mathbf{L} with non-zero eigenvalue as the solution $\mathbf{f} \in \mathbb{R}^n$.

According to Eq. (27), after finding $\mathbf{f} = [f_1, \dots, f_n]^\top$, we determine the cluster of each point as:

$$i := \begin{cases} \in \mathcal{A} & \text{if } f_i > 0, \\ \in \mathcal{A}' & \text{if } f_i \leq 0. \end{cases} \quad (30)$$

3.1.5. EXTENSION OF SPECTRAL CLUSTERING TO MULTIPLE CLUSTERS

If we have more than binary clusters in data, the optimization (28):

$$\begin{aligned} & \underset{\mathbf{F}}{\text{minimize}} && \text{tr}(\mathbf{F}^\top \mathbf{L} \mathbf{F}) \\ & \text{subject to} && \mathbf{F}^\top \mathbf{F} = \mathbf{I}, \end{aligned} \quad (31)$$

where $\text{tr}(\cdot)$ denotes the trace of matrix and \mathbf{I} is the identity matrix. The Lagrangian is (Boyd et al., 2004):

$$\mathcal{L} = \mathbf{F}^\top \mathbf{L} \mathbf{F} - \text{tr}(\mathbf{\Lambda}^\top (\mathbf{F}^\top \mathbf{F} - \mathbf{I})),$$

where $\mathbf{\Lambda}$ is the diagonal matrix with the dual variables. Setting the derivative of Lagrangian to zero gives:

$$\mathbb{R}^{n \times n} \ni \frac{\partial \mathcal{L}}{\partial \mathbf{F}} = 2\mathbf{L}\mathbf{F} - 2\mathbf{F}\mathbf{\Lambda} \stackrel{\text{set}}{=} \mathbf{0} \implies \mathbf{L}\mathbf{F} = \mathbf{F}\mathbf{\Lambda}, \quad (32)$$

which is the eigenvalue problem for the Laplacian matrix \mathbf{L} where columns of \mathbf{F} are the eigenvectors (Ghojogh et al., 2019a). As the problem is minimization, the columns of \mathbf{F} are sorted from corresponding smallest to largest eigenvalues. Also, as it is eigenvalue problem for the Laplacian matrix, we ignore the smallest eigenvector with eigenvalue zero (see Section 2.4).

If we have c number of clusters, we truncate the sorted \mathbf{F} to have the c smallest eigenvectors with non-zero eigenvalues. Hence, we will have $\mathbf{F} \in \mathbb{R}^{n \times c}$.

After finding the matrix $\mathbf{F} \in \mathbb{R}^{n \times c}$, we treat the rows of \mathbf{F} as new data points in a c -dimensional space. We apply another clustering algorithm, such as K-means, on the rows of \mathbf{F} in order to cluster the n points into c clusters (Ng et al., 2001). Note that regular clustering methods, such as K-means, are applied on the input space (space of \mathbf{X}) while spectral clustering is applied on the embedding space (space of \mathbf{Y}). In other words, spectral clustering first extracts features for better discrimination of clusters and then applies clustering. This usually results in better clustering because the extracted features are supposed to discriminate the data clusters more.

Recall that as was mentioned in Section 2.2, some papers use Eq. (8) for Laplacian and they use maximization instead of minimization.

3.1.6. OPTIMIZATION APPROACH 2

In the original work of spectral clustering, the following optimization is used rather than Eq. (31) (Shi & Malik, 1997):

$$\begin{aligned} & \underset{\mathbf{F}}{\text{minimize}} && \text{tr}(\mathbf{F}^\top \mathbf{L} \mathbf{F}) \\ & \text{subject to} && \mathbf{F}^\top \mathbf{D} \mathbf{F} = \mathbf{I}, \end{aligned} \quad (33)$$

which makes the rotated \mathbf{F} , by the degree matrix, orthogonal. Note that, according to Eq. (5), the degree matrix is a natural measure for the data points in the way that the larger d_i corresponds to the more important f_i because it has more similarity (weights) with its neighbors (He & Niyogi, 2004). That is why having \mathbf{D} in the constraint makes sense as it gives more weight to the more important f_i 's.

The Lagrangian is (Boyd et al., 2004):

$$\mathcal{L} = \mathbf{F}^\top \mathbf{L} \mathbf{F} - \text{tr}(\mathbf{\Lambda}^\top (\mathbf{F}^\top \mathbf{D} \mathbf{F} - \mathbf{I})),$$

where $\mathbf{\Lambda}$ is the diagonal matrix with the dual variables. Setting the derivative of Lagrangian to zero gives:

$$\mathbb{R}^{n \times n} \ni \frac{\partial \mathcal{L}}{\partial \mathbf{F}} = 2\mathbf{L}\mathbf{F} - 2\mathbf{D}\mathbf{F}\mathbf{\Lambda} \stackrel{\text{set}}{=} \mathbf{0} \implies \mathbf{L}\mathbf{F} = \mathbf{D}\mathbf{F}\mathbf{\Lambda}, \quad (34)$$

which is the generalized eigenvalue problem (\mathbf{L}, \mathbf{D}) where columns of \mathbf{F} are the eigenvectors (Ghojogh et al., 2019a). Sorting the matrix \mathbf{F} , ignoring the eigenvector with zero eigenvalue (see Section 2.4), and the rest of algorithm are done as explained before.

3.2. Other Improvements over Spectral Clustering

There exist many different improvements over the basic spectral clustering which was explained. Some of these developments are distributed spectral clustering (Chen et al., 2010), consistency spectral clustering (Von Luxburg et al., 2008), correctional spectral clustering (Blaschko & Lampert, 2008), spectral clustering by autoencoder (Banijamali

& Ghodsi, 2017), multi-view spectral clustering (Kumar et al., 2011; Kumar & Daumé, 2011; Yin et al., 2019), self-tuning spectral clustering (Zelnik-Manor & Perona, 2004), and fuzzy spectral clustering (Yang & Wang, 2016). Some existing surveys and tutorials on spectral clustering are (Von Luxburg, 2007; Guan-zhong & Li-bin, 2008; Nascimento & De Carvalho, 2011; Guo et al., 2012). The optimization set-up of spectral clustering has also been successfully used in the field of hashing. Spectral hashing (Weiss et al., 2008) is a successful example. Moreover, spectral clustering has had various applications, e.g., in image segmentation (Yang et al., 2016).

4. Laplacian Eigenmap

4.1. Laplacian Eigenmap

Laplacian eigenmap (Belkin & Niyogi, 2001; 2003) is a *nonlinear* dimensionality reduction method which is related to spectral clustering. As was explained in Section 3.1.5, spectral clustering extracts features for better discriminative clusters and then applies clustering in the embedding space. Therefore, spectral clustering has some dimensionality reduction and feature extraction task inside it. This guided researchers to propose Laplacian eigenmap based on the concepts of spectral clustering.

Consider the dataset matrix $\mathbf{X} := [\mathbf{x}_1, \dots, \mathbf{x}_n] \in \mathbb{R}^{d \times n}$. We desire p -dimensional embeddings of data points $\mathbf{Y} := [\mathbf{y}_1, \dots, \mathbf{y}_n] \in \mathbb{R}^{n \times p}$ where $p \leq d$ and usually $p \ll d$. Note that in Laplacian eigenmap, for simplicity of algebra, we put the embedding vectors row-wise in the matrix \mathbf{Y} .

4.1.1. ADJACENCY MATRIX

Similar to spectral clustering, we create an adjacency matrix using Eq. (1). The elements w_{ij} can be determined by the RBF kernel, i.e. Eq. (2), or the simple-minded approach, i.e. Eq. (3).

4.1.2. INTERPRETATION OF LAPLACIAN EIGENMAP

Inspired by Eq. (23) in spectral clustering, the optimization problem of Laplacian eigenmap is:

$$\underset{\mathbf{Y}}{\text{minimize}} \quad \sum_{i=1}^n \sum_{j=1}^n w_{ij} \|\mathbf{y}_i - \mathbf{y}_j\|_2^2. \quad (35)$$

Eq. (35) can be interpreted as the following (Belkin & Niyogi, 2003):

- If \mathbf{x}_i and \mathbf{x}_j are close to each other, w_{ij} is large. Hence, for minimizing the objective function, the term $\|\mathbf{y}_i - \mathbf{y}_j\|_2^2$ should be minimized which results in close \mathbf{y}_i and \mathbf{y}_j . This is expected because \mathbf{x}_i and \mathbf{x}_j are close and their embeddings should be close as well.
- If \mathbf{x}_i and \mathbf{x}_j are far apart, w_{ij} is small. Hence, the objective function is small because of being multiplied

by the small weight w_{ij} . Hence, we do not care about \mathbf{y}_i and \mathbf{y}_j much as the objective is already small.

According to the above two notes, we see that Laplacian eigenmap fits data locally because it cares more about the nearby points and hopes that the global structure is preserved by this local fitting (Saul & Roweis, 2003). In other words, we only care about nearby points and hope for correct relative embedding of far points. This might be considered as a weakness of Laplacian eigenmap. In another word, Laplacian eigenmap preserves locality of data pattern. This connects this method to the locality preserving projection which is explained later in Section 5.

4.1.3. OPTIMIZATION APPROACH 1

According to Proposition 2, we restate Eq. (35) with adding a constraint inspired by Eq. (31) as:

$$\begin{aligned} & \underset{\mathbf{Y}}{\text{minimize}} \quad \text{tr}(\mathbf{Y}^\top \mathbf{L} \mathbf{Y}) \\ & \text{subject to} \quad \mathbf{Y}^\top \mathbf{Y} = \mathbf{I}, \end{aligned} \quad (36)$$

where \mathbf{L} is the Laplacian of the adjacency matrix.

The Lagrangian for this optimization problem is (Boyd et al., 2004):

$$\mathcal{L} = \mathbf{Y}^\top \mathbf{L} \mathbf{Y} - \text{tr}(\mathbf{\Lambda}^\top (\mathbf{Y}^\top \mathbf{Y} - \mathbf{I})),$$

where $\mathbf{\Lambda}$ is the diagonal matrix with the dual variables. Setting the derivative of Lagrangian to zero gives:

$$\mathbb{R}^{n \times n} \ni \frac{\partial \mathcal{L}}{\partial \mathbf{Y}} = 2\mathbf{L}\mathbf{Y} - 2\mathbf{Y}\mathbf{\Lambda} \stackrel{\text{set}}{=} \mathbf{0} \implies \mathbf{L}\mathbf{Y} = \mathbf{Y}\mathbf{\Lambda}, \quad (37)$$

which is the eigenvalue problem for the Laplacian matrix \mathbf{L} where columns of \mathbf{Y} are the eigenvectors (Ghojogh et al., 2019a). As the problem is minimization, the columns of \mathbf{Y} are sorted from corresponding smallest to largest eigenvalues. Also, as it is eigenvalue problem for the Laplacian matrix, we ignore the smallest eigenvector with eigenvalue zero (see Section 2.4). For a p -dimensional embedding space, we truncate the sorted \mathbf{Y} to have the p smallest eigenvectors with non-zero eigenvalues. Hence, we will have $\mathbf{Y} \in \mathbb{R}^{n \times p}$. The rows of \mathbf{Y} are the p -dimensional embedding vectors.

Note that as the adjacency matrix can use any kernel, such as the RBF kernel of Eq. (2), some papers have named it the *kernel Laplacian eigenmap* (Guo et al., 2006).

4.1.4. OPTIMIZATION APPROACH 2

Another optimization approach for Laplacian eigenmap is to use the following optimization rather than Eq. (36) (Belkin & Niyogi, 2001):

$$\begin{aligned} & \underset{\mathbf{Y}}{\text{minimize}} \quad \text{tr}(\mathbf{Y}^\top \mathbf{L} \mathbf{Y}) \\ & \text{subject to} \quad \mathbf{Y}^\top \mathbf{D} \mathbf{Y} = \mathbf{I}, \end{aligned} \quad (38)$$

where D is the degree matrix defined by Eq. (5). In this optimization, rather than having orthonormal embeddings, we want the rotated embeddings by the degree matrix to be orthonormal. Note that, according to Eq. (5), the degree matrix is a natural measure for the data points in the way that the larger d_i corresponds to the more important x_i because it has more similarity (weights) with its neighbors (He & Niyogi, 2004). That is why having D in the constraint makes sense as it gives more weight to the more important y_i 's.

The Lagrangian for this optimization problem is (Boyd et al., 2004):

$$\mathcal{L} = Y^\top LY - \text{tr}(\Lambda^\top (Y^\top DY - I)),$$

where Λ is the diagonal matrix with the dual variables. Setting the derivative of Lagrangian to zero gives:

$$\begin{aligned} \mathbb{R}^{n \times n} \ni \frac{\partial \mathcal{L}}{\partial Y} &= 2LY - 2DY\Lambda \stackrel{\text{set}}{=} 0 \\ \implies LY &= DY\Lambda, \end{aligned} \quad (39)$$

which is the generalized eigenvalue problem (L, D) where columns of Y are the eigenvectors (Ghojogh et al., 2019a). Sorting the matrix Y and ignoring the eigenvector with zero eigenvalue (see Section 2.4) is done as explained before.

It is shown in (Weiss, 1999, Normalization Lemma 1) that the solution to Eq. (39) is equivalent to the solution of eigenvalue problem for Eq. (9). In other words, the eigenvectors of Eq. (9) are equivalent to generalized eigenvectors of Eq. (39).

4.2. Out-of-sample Extension for Laplacian Eigenmap

So far, we embedded the training dataset $\{x_i \in \mathbb{R}^d\}_{i=1}^n$ or $X = [x_1, \dots, x_n] \in \mathbb{R}^{d \times n}$ to have their embedding $\{y_i \in \mathbb{R}^p\}_{i=1}^n$ or $Y = [y_1, \dots, y_n]^\top \in \mathbb{R}^{n \times p}$. Assume we have some out-of-sample (test data), denoted by $\{x_i^{(t)} \in \mathbb{R}^d\}_{i=1}^{n_t}$ or $X_t = [x_1^{(t)}, \dots, x_{n_t}^{(t)}] \in \mathbb{R}^{d \times n_t}$. We want to find their embedding $\{y_i^{(t)} \in \mathbb{R}^p\}_{i=1}^{n_t}$ or $Y_t = [y_1^{(t)}, \dots, y_{n_t}^{(t)}]^\top \in \mathbb{R}^{n_t \times p}$ after the training phase.

4.2.1. EIGENFUNCTIONS

Consider a Hilbert space \mathcal{H}_p of functions with the inner product $\langle f, g \rangle = \int f(x)g(x)p(x)dx$ with density function $p(x)$. In this space, we can consider the kernel function K_p :

$$(K_p f)(x) = \int K(x, y) f(y) p(y) dy, \quad (40)$$

where the density function can be approximated empirically. The *eigenfunction decomposition* is defined to be (Bengio et al., 2004a;b):

$$(K_p f_k)(x) = \delta'_k f_k(x), \quad (41)$$

where $f_k(x)$ is the k -th *eigenfunction* and δ'_k is the corresponding eigenvalue. If we have the eigenvalue decomposition (Ghojogh et al., 2019a) for the kernel matrix K , we have $K v_k = \delta_k v_k$ where v_k is the k -th eigenvector and δ_k is the corresponding eigenvalue. According to (Bengio et al., 2004b, Proposition 1), we have $\delta'_k = (1/n)\delta_k$.

4.2.2. EMBEDDING USING EIGENFUNCTIONS

Proposition 3. If v_{ki} is the i -th element of the n -dimensional vector v_k and $k(x, x_i)$ is the kernel between vectors x and x_i , the eigenfunction for the point x and the i -th training point x_i are:

$$f_k(x) = \frac{\sqrt{n}}{\delta_k} \sum_{i=1}^n v_{ki} \check{k}_t(x_i, x), \quad (42)$$

$$f_k(x_i) = \sqrt{n} v_{ki}, \quad (43)$$

respectively, where $\check{k}_t(x_i, x)$ is the centered kernel between training set and the out-of-sample point x .

Let the embedding of the point x be $\mathbb{R}^p \ni y(x) = [y_1(x), \dots, y_p(x)]^\top$. The k -th dimension of this embedding is:

$$y_k(x) = \sqrt{\delta_k} \frac{f_k(x)}{\sqrt{n}} = \frac{1}{\sqrt{\delta_k}} \sum_{i=1}^n v_{ki} \check{k}_t(x_i, x). \quad (44)$$

Proof. This proposition is taken from (Bengio et al., 2004b, Proposition 1). For proof, refer to (Bengio et al., 2004a, Proposition 1), (Bengio et al., 2006, Proposition 1), and (Bengio et al., 2003c, Proposition 1 and Theorem 1). More complete proofs can be found in (Bengio et al., 2003b). \square

We can see the adjacency matrix as a kernel (e.g. see Eq. (2) where RBF kernel is used for adjacency matrix). This fact was also explained in Section 2.2 where we said why Laplacian can be seen as a normalized kernel matrix. Hence, here, kernel $k_t(x_i, x)$ refers to the adjacency matrix.

According to Eq. (9) and Eq. (32), the solution of spectral clustering is the eigenvectors of Laplacian in Eq. (9). Moreover, as was explained in Section 4.1.4, the solution of Laplacian eigenmap is also the eigenvectors of Laplacian in Eq. (9). Therefore, eigenfunctions can be used for out-of-sample extension.

If we have a set of n_t out-of-sample data points, we can compute the kernel $K_t \in \mathbb{R}^{n \times n_t}$ between training and out-of-sample data. We can normalize the kernel matrix (Ah-Pine, 2010) as follows. The $\check{k}_t(x_i, x)$, used in Eq. (44), is an element of the normalized kernel (adjacency) matrix (Bengio et al., 2003a):

$$\check{k}_t(x_i, x) := \frac{1}{n} \frac{k_t(x_i, x)}{\sqrt{\mathbb{E}[K_t(x_i, \cdot)] \mathbb{E}[K_t(\cdot, x)]}}, \quad (45)$$

where the expectations are computed as:

$$\mathbb{E}[\mathbf{K}_t(\mathbf{x}_i, \cdot)] \approx \frac{1}{n_t} \sum_{j=1}^{n_t} k_t(\mathbf{x}_i, \mathbf{x}_j^{(t)}), \quad (46)$$

$$\mathbb{E}[\mathbf{K}_t(\cdot, \mathbf{x})] \approx \frac{1}{n} \sum_{i=1}^n k_t(\mathbf{x}_i, \mathbf{x}). \quad (47)$$

Noticing the Eq. (5) for \mathbf{D} and comparing Eq. (9) to Eqs. (45), (46), and (47) show the approximate equivalency of Eqs. (9) and (45), particularly if we assume $k_t \approx k$ and so $n_t \approx n$. Hence, Eq. (45) is valid to be used in Eq. (44).

4.2.3. OUT-OF-SAMPLE EMBEDDING

One can use Eq. (44) to embed the i -th out-of-sample data point $\mathbf{x}_i^{(t)}$. For this purpose, $\mathbf{x}_i^{(t)}$ should be used in place of \mathbf{x} in Eq. (44). This out-of-sample extension technique can be used for both Laplacian eigenmap and spectral clustering (Bengio et al., 2004b). In addition to this technique, there exist some other methods for out-of-sample extension of Laplacian eigenmap which we pass by in this paper. For example, two other extension methods are kernel mapping (Gisbrecht et al., 2012; 2015) and through optimization (Bunte et al., 2012).

4.3. Other Improvements over Laplacian Eigenmap

There have been recent developments over the basic Laplacian eigenmap. Some examples are Laplacian Eigenmaps Latent Variable Model (LELVM) (Carreira-Perpinán & Lu, 2007), robust Laplacian eigenmap (Roychowdhury, 2009), and Laplacian forest (Lombaert et al., 2014). Convergence analysis of Laplacian eigenmap methods can be found in several papers (Belkin & Niyogi, 2006; Singer, 2006). Inspired by eigenfaces (Turk & Pentland, 1991) and Fisherfaces (Belhumeur et al., 1997) which have been proposed based on principal component analysis (Ghojogh & Crowley, 2019) and Fisher discriminant analysis (Ghojogh et al., 2019b), respectively, *Laplacianfaces* (He et al., 2005) has been proposed for face recognition using Laplacian eigenmaps. Finally, a recent survey on Laplacian eigenmap is (Li et al., 2019; Wiskott & Schönfeld, 2019).

5. Locality Preserving Projection

Locality Preserving Projection (LPP) (He & Niyogi, 2004) is a linear dimensionality reduction method. It is a linear approximation of Laplacian eigenmap (Belkin & Niyogi, 2001; 2003). It uses linear projection in the formulation of Laplacian eigenmap. Hence, in the following, we first introduce the concept of linear projection. Note that this method is called locality preserving projection because it approximates the Laplacian eigenmap by linear projection where, as was explained in Section 4.1.2, the Laplacian eigenmap fits data locally and preserves the local structure of data in the embedding space.

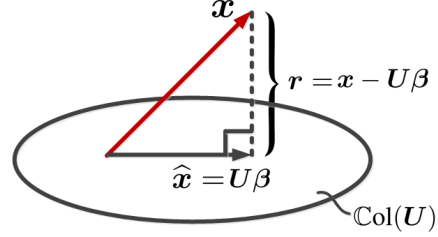


Figure 1. The residual and projection onto the column space of U . Credit of image is for (Ghojogh & Crowley, 2019).

5.1. Linear Projection

5.1.1. PROJECTION FORMULATION

Assume we have a data point $\mathbf{x} \in \mathbb{R}^d$. We want to project this data point onto the vector space spanned by p vectors $\{\mathbf{u}_1, \dots, \mathbf{u}_p\}$ where each vector is d -dimensional and usually $p \ll d$. We stack these vectors column-wise in matrix $U = [\mathbf{u}_1, \dots, \mathbf{u}_p] \in \mathbb{R}^{d \times p}$. In other words, we want to project \mathbf{x} onto the column space of U , denoted by $\text{Col}(U)$. The projection of $\mathbf{x} \in \mathbb{R}^d$ onto $\text{Col}(U) \in \mathbb{R}^p$ and then its representation in the \mathbb{R}^d (its reconstruction) can be seen as a linear system of equations:

$$\mathbb{R}^d \ni \hat{\mathbf{x}} := U\beta, \quad (48)$$

where we should find the unknown coefficients $\beta \in \mathbb{R}^p$.

If the \mathbf{x} lies in the $\text{Col}(U)$ or $\text{span}\{\mathbf{u}_1, \dots, \mathbf{u}_p\}$, this linear system has exact solution, so $\hat{\mathbf{x}} = \mathbf{x} = U\beta$. However, if \mathbf{x} does not lie in this space, there is no any solution β for $\mathbf{x} = U\beta$ and we should solve for projection of \mathbf{x} onto $\text{Col}(U)$ or $\text{span}\{\mathbf{u}_1, \dots, \mathbf{u}_p\}$ and then its reconstruction. In other words, we should solve for Eq. (48). In this case, $\hat{\mathbf{x}}$ and \mathbf{x} are different and we have a residual:

$$\mathbf{r} = \mathbf{x} - \hat{\mathbf{x}} = \mathbf{x} - U\beta, \quad (49)$$

which we want to be small. As can be seen in Fig. 1, the smallest residual vector is orthogonal to $\text{Col}(U)$; therefore:

$$\begin{aligned} \mathbf{x} - U\beta \perp U &\implies U^\top(\mathbf{x} - U\beta) = 0, \\ &\implies \beta = (U^\top U)^{-1} U^\top \mathbf{x}. \end{aligned} \quad (50)$$

It is noteworthy that the Eq. (50) is also the formula of coefficients in linear regression (Friedman et al., 2009) where the input data are the rows of U and the labels are \mathbf{x} ; however, our goal here is different.

Plugging Eq. (50) in Eq. (48) gives us:

$$\hat{\mathbf{x}} = U(U^\top U)^{-1} U^\top \mathbf{x}.$$

We define:

$$\mathbb{R}^{d \times d} \ni \Pi := U(U^\top U)^{-1} U^\top, \quad (51)$$

as “projection matrix” because it projects \mathbf{x} onto $\text{Col}(\mathbf{U})$ (and reconstructs back). Note that $\mathbf{\Pi}$ is also referred to as the “hat matrix” in the literature because it puts a hat on top of \mathbf{x} .

If the vectors $\{\mathbf{u}_1, \dots, \mathbf{u}_p\}$ are orthonormal (the matrix \mathbf{U} is orthogonal), we have $\mathbf{U}^\top = \mathbf{U}^{-1}$ and thus $\mathbf{U}^\top \mathbf{U} = \mathbf{I}$. Therefore, Eq. (51) is simplified:

$$\mathbf{\Pi} = \mathbf{U}\mathbf{U}^\top. \quad (52)$$

So, we have:

$$\hat{\mathbf{x}} = \mathbf{\Pi} \mathbf{x} = \mathbf{U}\mathbf{U}^\top \mathbf{x}. \quad (53)$$

5.1.2. PROJECTION ONTO A SUBSPACE

In subspace learning, the projection of a vector $\mathbf{x} \in \mathbb{R}^d$ onto the column space of $\mathbf{U} \in \mathbb{R}^{d \times p}$ (a p -dimensional subspace spanned by $\{\mathbf{u}_j\}_{j=1}^p$ where $\mathbf{u}_j \in \mathbb{R}^d$) is defined as:

$$\mathbb{R}^p \ni \tilde{\mathbf{x}} := \mathbf{U}^\top \mathbf{x}, \quad (54)$$

$$\mathbb{R}^d \ni \hat{\mathbf{x}} := \mathbf{U}\mathbf{U}^\top \mathbf{x} = \mathbf{U}\tilde{\mathbf{x}}, \quad (55)$$

where $\tilde{\mathbf{x}}$ and $\hat{\mathbf{x}}$ denote the projection and reconstruction of \mathbf{x} , respectively.

If we have n data points, $\{\mathbf{x}_i\}_{i=1}^n$, which can be stored column-wise in a matrix $\mathbf{X} \in \mathbb{R}^{d \times n}$, the projection and reconstruction of \mathbf{X} are defined as:

$$\mathbb{R}^{p \times n} \ni \tilde{\mathbf{X}} := \mathbf{U}^\top \mathbf{X}, \quad (56)$$

$$\mathbb{R}^{d \times n} \ni \hat{\mathbf{X}} := \mathbf{U}\mathbf{U}^\top \mathbf{X} = \mathbf{U}\tilde{\mathbf{X}}, \quad (57)$$

respectively.

If we have an out-of-sample data point \mathbf{x}_t , which was not used in calculation of \mathbf{U} , the projection and reconstruction of it are defined as:

$$\text{out of sample } \mathbb{R}^p \ni \tilde{\mathbf{x}}_t := \mathbf{U}^\top \mathbf{x}_t, \quad (58)$$

$$\mathbb{R}^d \ni \hat{\mathbf{x}}_t := \mathbf{U}\mathbf{U}^\top \mathbf{x}_t = \mathbf{U}\tilde{\mathbf{x}}_t, \quad (59)$$

respectively.

In case we have n_t out-of-sample data points, $\{\mathbf{x}_{t,i}\}_{i=1}^{n_t}$, which can be stored column-wise in a matrix $\mathbf{X}_t \in \mathbb{R}^{d \times n_t}$, the projection and reconstruction of \mathbf{X}_t are defined as:

$$\mathbb{R}^{p \times n_t} \ni \tilde{\mathbf{X}}_t := \mathbf{U}^\top \mathbf{X}_t, \quad (60)$$

$$\mathbb{R}^{d \times n_t} \ni \hat{\mathbf{X}}_t := \mathbf{U}\mathbf{U}^\top \mathbf{X}_t = \mathbf{U}\tilde{\mathbf{X}}_t, \quad (61)$$

respectively.

For the properties of the projection matrix \mathbf{U} , refer to (Ghosh & Crowley, 2019).

5.2. Locality Preserving Projection

LPP assumes that the embeddings of data points are obtained by a linear projection. It approximates Laplacian eigenmap linearly. We want to find the embeddings of data points $\mathbf{X} = [\mathbf{x}_1, \dots, \mathbf{x}_n] \in \mathbb{R}^{d \times n}$ using linear projection.

5.2.1. ONE-DIMENSIONAL SUBSPACE

For derivation of LPP optimization, we assume $p = 1$ so the projection is onto a line with projection vector $\mathbf{u} \in \mathbb{R}^d$. Later, we will extend to higher values for p . In this case, the vector of projections of all points is:

$$\mathbb{R}^{1 \times n} \ni \mathbf{Y} = [y_1, y_2, \dots, y_n] := \mathbf{u}^\top \mathbf{X}. \quad (62)$$

Hence, the one-dimensional embedding of every point is:

$$\mathbb{R} \ni y_i = \mathbf{u}^\top \mathbf{x}_i, \quad \forall i \in \{1, \dots, n\}. \quad (63)$$

Inspired by Eq. (35) for Laplacian eigenmap, the optimization for obtaining the one-dimensional embedding is:

$$\underset{\mathbf{Y}}{\text{minimize}} \quad \sum_{i=1}^n \sum_{j=1}^n w_{ij} (y_i - y_j)^2, \quad (64)$$

for the same interpretation that we had in Laplacian eigenmap.

Proposition 4 ((He & Niyogi, 2004)). *We have:*

$$\frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n w_{ij} (y_i - y_j)^2 = \mathbf{u}^\top \mathbf{X} \mathbf{L} \mathbf{X}^\top \mathbf{u}. \quad (65)$$

Proof. The proof is very similar to the proof of Proposition 2. We have:

$$\begin{aligned} \mathbf{u}^\top \mathbf{X} \mathbf{L} \mathbf{X}^\top \mathbf{u} &\stackrel{(4)}{=} \mathbf{u}^\top \mathbf{X} (\mathbf{D} - \mathbf{W}) \mathbf{X}^\top \mathbf{u} \\ &= \mathbf{u}^\top \mathbf{X} \mathbf{D} \mathbf{X}^\top \mathbf{u} - \mathbf{u}^\top \mathbf{X} \mathbf{W} \mathbf{X}^\top \mathbf{u} \\ &\stackrel{(a)}{=} \sum_{i=1}^n \mathbf{u}^\top \mathbf{x}_i d_i \mathbf{x}_i^\top \mathbf{u} - \sum_{i=1}^n \sum_{j=1}^n \mathbf{u}^\top \mathbf{x}_i w_{ij} \mathbf{x}_j^\top \mathbf{u} \\ &= \frac{1}{2} \left(2 \sum_{i=1}^n \mathbf{u}^\top \mathbf{x}_i d_i \mathbf{x}_i^\top \mathbf{u} - 2 \sum_{i=1}^n \sum_{j=1}^n \mathbf{u}^\top \mathbf{x}_i w_{ij} \mathbf{x}_j^\top \mathbf{u} \right) \\ &\stackrel{(b)}{=} \frac{1}{2} \left(2 \sum_{i=1}^n (\mathbf{u}^\top \mathbf{x}_i)^2 d_i - 2 \sum_{i=1}^n \sum_{j=1}^n (\mathbf{u}^\top \mathbf{x}_i) (\mathbf{u}^\top \mathbf{x}_j) w_{ij} \right) \\ &\stackrel{(c)}{=} \frac{1}{2} \left(\sum_{i=1}^n (\mathbf{u}^\top \mathbf{x}_i)^2 d_i + \sum_{j=1}^n (\mathbf{u}^\top \mathbf{x}_j)^2 d_j \right. \\ &\quad \left. - 2 \sum_{i=1}^n \sum_{j=1}^n (\mathbf{u}^\top \mathbf{x}_i) (\mathbf{u}^\top \mathbf{x}_j) w_{ij} \right) \end{aligned}$$

$$\begin{aligned}
& \stackrel{(5)}{=} \frac{1}{2} \left(\sum_{i=1}^n (\mathbf{u}^\top \mathbf{x}_i)^2 \sum_{j=1}^n w_{ij} + \sum_{j=1}^n (\mathbf{u}^\top \mathbf{x}_j)^2 \sum_{i=1}^n w_{ji} \right. \\
& \quad \left. - 2 \sum_{i=1}^n \sum_{j=1}^n (\mathbf{u}^\top \mathbf{x}_i)(\mathbf{u}^\top \mathbf{x}_j) w_{ij} \right) \\
& \stackrel{(d)}{=} \frac{1}{2} \left(\sum_{i=1}^n (\mathbf{u}^\top \mathbf{x}_i)^2 \sum_{j=1}^n w_{ij} + \sum_{j=1}^n (\mathbf{u}^\top \mathbf{x}_j)^2 \sum_{i=1}^n w_{ij} \right. \\
& \quad \left. - 2 \sum_{i=1}^n \sum_{j=1}^n (\mathbf{u}^\top \mathbf{x}_i)(\mathbf{u}^\top \mathbf{x}_j) w_{ij} \right) \\
& = \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n w_{ij} (\mathbf{u}^\top \mathbf{x}_i - \mathbf{u}^\top \mathbf{x}_j)^2 \\
& \stackrel{(63)}{=} \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n w_{ij} (y_i - y_j)^2,
\end{aligned}$$

where (a) notices that \mathbf{D} is a diagonal matrix, (b) is because $\mathbf{u}^\top \mathbf{x}_i = \mathbf{x}_i^\top \mathbf{u}_i \in \mathbb{R}$, (c) is because we can replace the dummy index i with j , and (d) notices that $w_{ij} = w_{ji}$ as we usually use symmetric similarity measures. Q.E.D. \square

According to Proposition 4, we restate Eq. (64) with adding a constraint inspired by Eq. (38) as:

$$\begin{aligned}
& \underset{\mathbf{u}}{\text{minimize}} \quad \mathbf{u}^\top \mathbf{X} \mathbf{L} \mathbf{X}^\top \mathbf{u} \\
& \text{subject to} \quad \mathbf{u}^\top \mathbf{X} \mathbf{D} \mathbf{X}^\top \mathbf{u} = 1,
\end{aligned} \tag{66}$$

where \mathbf{L} and \mathbf{D} are the Laplacian and degree matrix of the adjacency matrix, respectively. Note that the objective variable has been changed to the projection vector \mathbf{u} .

The Lagrangian for this optimization problem is (Boyd et al., 2004):

$$\mathcal{L} = \mathbf{u}^\top \mathbf{X} \mathbf{L} \mathbf{X}^\top \mathbf{u} - \lambda^\top (\mathbf{u}^\top \mathbf{X} \mathbf{D} \mathbf{X}^\top \mathbf{u} - 1),$$

where λ is the dual variable. Setting the derivative of Lagrangian to zero gives:

$$\begin{aligned}
\mathbb{R}^d \ni \frac{\partial \mathcal{L}}{\partial \mathbf{u}} &= 2\mathbf{X} \mathbf{L} \mathbf{X}^\top \mathbf{u} - 2\lambda \mathbf{X} \mathbf{D} \mathbf{X}^\top \mathbf{u} \stackrel{\text{set}}{=} \mathbf{0} \\
&\implies \mathbf{X} \mathbf{L} \mathbf{X}^\top \mathbf{u} = \lambda \mathbf{X} \mathbf{D} \mathbf{X}^\top \mathbf{u},
\end{aligned} \tag{67}$$

which is the generalized eigenvalue problem $(\mathbf{X} \mathbf{L} \mathbf{X}^\top, \mathbf{X} \mathbf{D} \mathbf{X}^\top)$ where \mathbf{u} is the eigenvector (Ghojogh et al., 2019a). The embedded points are obtained using Eq. (62).

5.2.2. MULTI-DIMENSIONAL SUBSPACE

Now, we extend LPP to linear projection onto a span of p basis vectors. In other words, we extend it to have p -dimensional subspace. The embedding of \mathbf{x}_i is denoted by

$\mathbf{y}_i \in \mathbb{R}^p$. In LPP, we put the embedding vectors column-wise in matrix $\mathbf{Y} = [\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_n] \in \mathbb{R}^{p \times n}$. In this case, the projection is:

$$\mathbb{R}^{p \times n} \ni \mathbf{Y} := \mathbf{U}^\top \mathbf{X}, \tag{68}$$

where $\mathbf{U} = [\mathbf{u}_1, \dots, \mathbf{u}_p] \in \mathbb{R}^{d \times p}$ is the projection matrix onto its column space.

The Eq. (36) can be extended to multi-dimensional subspace as:

$$\begin{aligned}
& \underset{\mathbf{U}}{\text{minimize}} \quad \text{tr}(\mathbf{U}^\top \mathbf{X} \mathbf{L} \mathbf{X}^\top \mathbf{U}) \\
& \text{subject to} \quad \mathbf{U}^\top \mathbf{X} \mathbf{D} \mathbf{X}^\top \mathbf{U} = \mathbf{I}.
\end{aligned} \tag{69}$$

In optimization, it is assumed that the dimensionality of \mathbf{U} is $d \times d$ but we will truncate it to $d \times p$ after solving the optimization.

The Lagrangian for this optimization problem is (Boyd et al., 2004):

$$\mathcal{L} = \text{tr}(\mathbf{U}^\top \mathbf{X} \mathbf{L} \mathbf{X}^\top \mathbf{U}) - \text{tr}(\mathbf{\Lambda}^\top (\mathbf{U}^\top \mathbf{X} \mathbf{D} \mathbf{X}^\top \mathbf{U} - \mathbf{I})),$$

where $\mathbf{\Lambda}$ is the diagonal matrix with the dual variables. Setting the derivative of Lagrangian to zero gives:

$$\begin{aligned}
\mathbb{R}^{d \times d} \ni \frac{\partial \mathcal{L}}{\partial \mathbf{D}} &= 2\mathbf{X} \mathbf{L} \mathbf{X}^\top \mathbf{U} - 2\mathbf{X} \mathbf{D} \mathbf{X}^\top \mathbf{U} \mathbf{\Lambda} \stackrel{\text{set}}{=} \mathbf{0} \\
&\implies \mathbf{X} \mathbf{L} \mathbf{X}^\top \mathbf{U} = \mathbf{X} \mathbf{D} \mathbf{X}^\top \mathbf{U} \mathbf{\Lambda},
\end{aligned} \tag{70}$$

which is the generalized eigenvalue problem $(\mathbf{X} \mathbf{L} \mathbf{X}^\top, \mathbf{X} \mathbf{D} \mathbf{X}^\top)$ where columns of \mathbf{U} are the eigenvectors (Ghojogh et al., 2019a). As it is a minimization problem, the columns of \mathbf{U} , which are the eigenvectors, are sorted from the corresponding smallest to largest eigenvalues. Moreover, as we have Laplacian matrix, we have one eigenvector with eigenvalue zero which should be ignored (see Section 2.4). The embedded points are obtained using Eq. (68).

5.3. Kernel Locality Preserving Projection

Kernel LPP, also called kernel supervised LPP, (Cheng et al., 2005; Li et al., 2008) performs LPP in the feature space. Assume $\phi(\cdot)$ denotes the pulling function to the feature space. According to representation theory (Alperin, 1993), any pulled solution (direction) $\phi(\mathbf{u}) \in \mathcal{H}$ must lie in the span of all the training vectors pulled to \mathcal{H} , i.e., $\Phi(\mathbf{X}) = [\phi(\mathbf{x}_1), \dots, \phi(\mathbf{x}_n)] \in \mathbb{R}^{t \times n}$. Hence, $\mathbb{R}^t \ni \phi(\mathbf{u}) = \sum_{i=1}^n \theta_i \phi(\mathbf{x}_i) = \Phi(\mathbf{X}) \boldsymbol{\theta}$ where $\mathbb{R}^n \ni \boldsymbol{\theta} = [\theta_1, \dots, \theta_n]^\top$ is the unknown vector of coefficients. Considering multiple projection directions, we have:

$$\mathbb{R}^{t \times p} \ni \Phi(\mathbf{U}) = \Phi(\mathbf{X}) \boldsymbol{\Theta}, \tag{71}$$

where $\mathbb{R}^{n \times p} \ni \Theta = [\theta_1, \dots, \theta_p]$. Therefore, the objective function in Eq. (69) is converted to:

$$\begin{aligned} & \text{tr}(\Phi(U)^\top \Phi(X) L \Phi(X)^\top \Phi(U)) \\ & \stackrel{(71)}{=} \text{tr}(\Theta^\top \Phi(X)^\top \Phi(X) L \Phi(X)^\top \Phi(X) \Theta) \\ & = \text{tr}(\Theta^\top K_x L K_x \Theta), \end{aligned}$$

where:

$$\mathbb{R}^{n \times n} \ni K_x := \Phi(X)^\top \Phi(X), \quad (72)$$

is the kernel of data (Hofmann et al., 2008). Similarly, the constraint of Eq. (69) is converted to $\Theta^\top K_x D K_x \Theta$. Finally, the Eq. (69) is changed to:

$$\begin{aligned} & \underset{\Theta}{\text{minimize}} \quad \text{tr}(\Theta^\top K_x L K_x \Theta) \\ & \text{subject to} \quad \Theta^\top K_x D K_x \Theta = I, \end{aligned} \quad (73)$$

where Θ is the optimization variable. The solution to this problem is similar to the solution of Eq. (69). So, its solution is the **generalized eigenvalue problem** $(K_x L K_x^\top, K_x D K_x^\top)$ where columns of Θ are the eigenvectors (Ghojogh et al., 2019a). Sorting the matrix Θ and ignoring the eigenvector with zero eigenvalue (see Section 2.4) is done as explained before. The p smallest eigenvectors form $\Theta \in \mathbb{R}^{n \times p}$. **The embedded data are obtained as:**

$$\begin{aligned} \mathbb{R}^{p \times n} \ni Y & \stackrel{(68)}{=} \Phi(U)^\top \Phi(X) \stackrel{(71)}{=} \Theta^\top \Phi(X)^\top \Phi(X) \\ & \stackrel{(72)}{=} \Theta^\top K_x. \end{aligned} \quad (74)$$

5.4. Other Improvements over Locality Preserving Projection

There have been many improvements over the basic LPP. Some examples of these developments are supervised LPP (Wong & Zhao, 2012), extended LPP (Shikkenawis & Mitra, 2012), multiview uncorrelated LPP (Yin & Sun, 2019), graph-optimized LPP (Zhang et al., 2010), and LPP for Grassmann manifold (Wang et al., 2017). **For image recognition purposes, a two-dimensional (2D) LPP** (Chen et al., 2007) is proposed whose robust version is (Chen et al., 2019) and some of its applications are (Hu et al., 2007; Xu et al., 2009). A survey on LPP variants can be found in (Shikkenawis & Mitra, 2016). LPP has been shown to be effective for different applications such as document clustering (Cai et al., 2005), face recognition (He et al., 2003; Yu et al., 2006; Yang et al., 2017), and speech recognition (Tang & Rose, 2008).

6. Graph Embedding

Graph Embedding (GE) (Yan et al., 2005; 2006) is a generalized dimensionality reduction method where the graph of data is embedded in the lower dimensional space (Chang,

2014). Note that graph embedding is referred to as two different families of methods in the literature. In the first sense, including GE (Yan et al., 2005; 2006), it refers to **embedding the graph of data in the lower dimensional space**. The second sense, however, **refers to embedding every node of data graph in a subspace** (Goyal & Ferrara, 2018b). Here, our focus is on embedding the whole data graph. Note that there exist some other generalized subspace learning methods, such as Roweis discriminant analysis (Ghojogh et al., 2020c), in the literature.

6.1. Direct Graph Embedding

Consider the high dimensional data points $\{x_i \in \mathbb{R}^d\}_{i=1}^n$. We want to reduce the dimensionality of data from d to $p \leq d$ to have $\{y_i \in \mathbb{R}^p\}_{i=1}^n$, where usually $p \ll d$. We denote $X := [x_1, \dots, x_n] \in \mathbb{R}^{d \times n}$ and $Y := [y_1, \dots, y_n] \in \mathbb{R}^{p \times n}$. Similar to spectral clustering, we create an adjacency matrix using Eq. (1). The elements w_{ij} can be determined by the RBF kernel, i.e. Eq. (2), or the simple-minded approach, i.e. Eq. (3).

The objective function of optimization in **direct GE** (Yan et al., 2005; 2006) is inspired by Eq. (23) or Eq. (33) in spectral clustering:

$$\begin{aligned} & \underset{Y}{\text{minimize}} \quad \sum_{i=1}^n \sum_{j=1}^n w_{ij} \|y_i - y_j\|_2^2 \\ & \text{subject to} \quad Y^\top B Y = I, \end{aligned} \quad (75)$$

where $B \succeq 0$ is called the constraint matrix. According to Proposition 2, this optimization problem is equivalent to:

$$\begin{aligned} & \underset{Y}{\text{minimize}} \quad Y^\top L Y \\ & \text{subject to} \quad Y^\top B Y = I, \end{aligned} \quad (76)$$

where L is the Laplacian of graph of data.

The Lagrangian for this optimization problem is (Boyd et al., 2004):

$$\mathcal{L} = Y^\top L Y - \text{tr}(\Lambda^\top (Y^\top B Y - I)),$$

where Λ is the diagonal matrix with the dual variables. Setting the derivative of Lagrangian to zero gives:

$$\begin{aligned} \mathbb{R}^{n \times n} \ni \frac{\partial \mathcal{L}}{\partial Y} & = 2LY - 2BY\Lambda \stackrel{\text{set}}{=} 0 \\ \implies LY & = BY\Lambda, \end{aligned} \quad (77)$$

which is the **generalized eigenvalue problem** (L, B) where columns of Y are the eigenvectors (Ghojogh et al., 2019a). Sorting the matrix Y and ignoring the eigenvector with zero eigenvalue (see Section 2.4) is done as explained before.

6.2. Linearized Graph Embedding

As we had for LPP, linearized GE (Yan et al., 2005; 2006) assumes that the embedded data can be obtained by a linear projection onto the column space of a projection matrix $U \in \mathbb{R}^{d \times p}$. Hence, linearized GE uses similar optimization to Eq. (69):

$$\begin{aligned} & \underset{U}{\text{minimize}} \quad \text{tr}(U^\top X L X^\top U) \\ & \text{subject to} \quad U^\top X B X^\top U = I, \end{aligned} \quad (78)$$

whose solution is the generalized eigenvalue problem $(X L X^\top, X B X^\top)$, similar to Eq. (70).

Another approach for linearized GE is a slight revision to Eq. (78) which is (Yan et al., 2005):

$$\begin{aligned} & \underset{U}{\text{minimize}} \quad \text{tr}(U^\top X L X^\top U) \\ & \text{subject to} \quad U^\top U = I, \end{aligned} \quad (79)$$

whose solution is the eigenvalue decomposition of $X L X^\top$.

6.3. Kernelized Graph Embedding

Kernelized GE (Yan et al., 2005; 2006) performs the linearized GE in the feature space. It uses a similar optimization problem as Eq. (73):

$$\begin{aligned} & \underset{\Theta}{\text{minimize}} \quad \text{tr}(\Theta^\top K_x L K_x \Theta) \\ & \text{subject to} \quad \Theta^\top K_x B K_x \Theta = I, \end{aligned} \quad (80)$$

whose solution is the generalized eigenvalue problem $(K_x L K_x, K_x B K_x)$ where columns of Θ are the eigenvectors (Ghojogh et al., 2019a). Sorting the matrix Θ and ignoring the eigenvector with zero eigenvalue (see Section 2.4) is done as explained before. The embedding is then obtained using Eq. (74).

Another approach for linearized GE is a slight revision to Eq. (80). This approach is the kernelized version of Eq. (79) and is (Yan et al., 2005):

$$\begin{aligned} & \underset{\Theta}{\text{minimize}} \quad \text{tr}(\Theta^\top K_x L K_x \Theta) \\ & \text{subject to} \quad \Theta^\top K_x \Theta = I, \end{aligned} \quad (81)$$

whose solution is the generalized eigenvalue problem $(K_x L K_x, K_x)$.

6.4. Special Cases of Graph Embedding

GE is a generalized subspace learning method whose special cases are some of the well-known dimensionality reduction methods. These special cases are cases of either direct, or linearized, or kernelized GE. Some of the special cases are Laplacian eigenmap, LPP, kernel LPP, Principal Component Analysis (PCA), kernel PCA, Fisher Discriminant Analysis (FDA), kernel FDA, Multidimensional Scaling (MDS), Isomap, and Locally Linear Embedding (LLE).

6.4.1. LAPLACIAN EIGENMAP

Comparing Eq. (36) of Laplacian eigenmap with Eq. (76), we see that $B = I$. Also comparing Eq. (38) as the second approach of Laplacian eigenmap, with Eq. (76) shows that $B = D$. Hence, Laplacian eigenmap is a special case of direct GE.

6.4.2. LPP AND KERNEL LPP

Comparing Eq. (69) with Eq. (78) and also comparing Eq. (73) with Eq. (80) show that LPP and kernel LPP are special cases of linearized GE and kernelized GE, respectively.

6.4.3. PCA AND KERNEL PCA

The optimization of PCA is (Ghojogh & Crowley, 2019):

$$\begin{aligned} & \underset{U}{\text{minimize}} \quad \text{tr}(U^\top S U) \stackrel{(a)}{=} \text{tr}(U^\top X X^\top U) \\ & \text{subject to} \quad U^\top U = I, \end{aligned} \quad (82)$$

where U is the projection matrix, S is the covariance matrix of data, and (a) is because of $S = X X^\top$ assuming that data X are already centered. Moreover, the solution of kernel PCA is the eigenvalue decomposition of kernel of data (Ghojogh & Crowley, 2019); hence, optimization of kernel PCA is the kernelized version of Eq. (82). Comparing Eq. (82) with Eq. (79) shows that PCA is a special case of linearized GE with $L = I$. Moreover, kernel PCA is a special case of kernelized GE with $L = I$.

6.4.4. FDA AND KERNEL FDA

The optimization of FDA is (Ghojogh et al., 2019b; Fisher, 1936):

$$\begin{aligned} & \underset{U}{\text{maximize}} \quad \text{tr}(U^\top S_B U) \\ & \text{subject to} \quad U^\top S_W U = I, \end{aligned} \quad (83)$$

where S_B and S_W are the between- and within-class scatterers, respectively. We also have (Ye, 2007):

$$S = S_B + S_W \implies S_B = S - S_W, \quad (84)$$

where S denotes the total covariance of data. Hence, the Fisher criterion, to be maximized, can be restated as:

$$\frac{\text{tr}(U^\top S_B U)}{\text{tr}(U^\top S_W U)} = \frac{\text{tr}(U^\top S U)}{\text{tr}(U^\top S_W U)} - 1, \quad (85)$$

whose second constant term can be ignored in maximization. Therefore, Eq. (83) is restated to (Yan et al., 2005; Ghojogh et al., 2020c):

$$\begin{aligned} & \underset{U}{\text{maximize}} \quad \text{tr}(U^\top S U) \stackrel{(a)}{=} \text{tr}(U^\top X X^\top U) \\ & \text{subject to} \quad U^\top S_W U = I, \end{aligned} \quad (86)$$

where (a) is because of $S = X X^\top$ assuming that data X are already centered.

Let c_i denote the class to which the point x_i belongs and μ_{c_i} be the mean of class c_i . Let c denote the number of classes and n_j be the sample size of class j . For every class j , we define $e_j = [e_{j1}, \dots, e_{jn}]^\top \in \mathbb{R}^n$ as:

$$e_{ji} = \begin{cases} 1 & \text{if } c_i = j, \\ 0 & \text{otherwise.} \end{cases} \quad (87)$$

Note that the within scatter can be obtained as (Yan et al., 2005):

$$\begin{aligned} S_W &= \sum_{i=1}^n (x_i - \mu_{c_i})(x_i - \mu_{c_i})^\top \\ &= X(I - \sum_{j=1}^c \frac{1}{n_j} e_j e_j^\top) X^\top. \end{aligned} \quad (88)$$

Comparing Eq. (86) with Eq. (78) and noticing Eq. (88) show that FDA is a special case of linearized GE with $L = I$ and $B = I - \sum_{j=1}^c (1/n_j) e_j e_j^\top$. Kernel FDA (Ghojogh et al., 2019b; Mika et al., 1999) is the kernelized version of Eq. (86) so it is a special case of kernelized GE with the mentioned L and B matrices.

6.4.5. MDS AND ISOMAP

Kernel classical MDS uses a kernel as (Ghojogh et al., 2020b):

$$\mathbb{R}^{n \times n} \ni K = -\frac{1}{2} H D H, \quad (89)$$

where $D \in \mathbb{R}^{n \times n}$ is a matrix with squared Euclidean distance between points and $\mathbb{R}^{n \times n} \ni H := I - (1/n) \mathbf{1} \mathbf{1}^\top$ is the centering matrix. Isomap also applies multidimensional scaling with a geodesic kernel which uses piece-wise Euclidean distance for computing D (Tenenbaum et al., 2000; Ghojogh et al., 2020b). The row summation of this kernel matrix is (Yan et al., 2005; 2006):

$$\begin{aligned} \sum_{j=1}^n K_{ij} &\stackrel{(89)}{=} \sum_{j=1}^n (-\frac{1}{2} H D H)_{ij} \\ &= \sum_{j=1}^n (-\frac{1}{2} (I - \frac{1}{n} \mathbf{1} \mathbf{1}^\top) D (I - \frac{1}{n} \mathbf{1} \mathbf{1}^\top))_{ij} \\ &= \frac{1}{2} \sum_{j=1}^n (-D_{ij} + \frac{1}{n} \sum_{i'=1}^n D_{i'j} \\ &\quad + \frac{1}{n} \sum_{j'=1}^n (D_{ij'} - \frac{1}{n} \sum_{k'=1}^n D_{k'j'})) \\ &= (-\frac{1}{2} \sum_{j=1}^n D_{ij} + \frac{1}{2n} \sum_{j=1}^n \sum_{i'=1}^n D_{i'j}) + (\frac{1}{2n} \sum_{j=1}^n \sum_{j'=1}^n D_{ij'} \\ &\quad - \frac{1}{2n^2} \sum_{j=1}^n \sum_{j'=1}^n \sum_{k'=1}^n D_{k'j'}) = \mathbf{0} + \mathbf{0} = \mathbf{0}. \end{aligned}$$

According to Eq. (6), the kernel used in kernel classical MDS or the geodesic kernel used in Isomap can be interpreted as the Laplacian matrix of graph of data as it satisfies its row-sum property. The optimization of MDS or Isomap is (Ghojogh et al., 2020b):

$$\begin{aligned} &\underset{Y}{\text{minimize}} \quad Y^\top K Y \\ &\text{subject to} \quad Y^\top Y = I, \end{aligned} \quad (90)$$

with kernel of Eq. (89). Comparing Eq. (90) with Eq. (76) shows that kernel classical MDS and Isomap are special cases of direct GE with $L = K$ and $B = I$.

6.4.6. LLE

Let $W \in \mathbb{R}^{n \times n}$ be the reconstruction weight matrix in LLE (Roweis & Saul, 2000; Ghojogh et al., 2020a). Let $\mathbb{R}^{n \times n} \ni M := (I - W)(I - W)^\top$. The row summation of matrix M is (Yan et al., 2005; 2006):

$$\begin{aligned} \sum_{j=1}^n M_{ij} &= \sum_{j=1}^n ((I - W)(I - W)^\top)_{ij} \\ &= \sum_{j=1}^n (I_{ij} - W_{ij} - W_{ji} + (W W^\top)_{ij}) \\ &= 1 - \sum_{j=1}^n (W_{ij} + W_{ji}) + \sum_{j=1}^n \sum_{k=1}^n (W_{ij} W_{jk}) \\ &= 1 - \sum_{j=1}^n (W_{ij} + W_{ji}) + \sum_{j=1}^n (W_{ij} \sum_{k=1}^n W_{jk}) \\ &\stackrel{(a)}{=} 1 - \sum_{j=1}^n W_{ij} - \sum_{j=1}^n W_{ji} + \sum_{j=1}^n W_{ij} = \mathbf{0}, \end{aligned}$$

where (a) is because the row summation of reconstruction matrix is one, i.e. $\sum_{k=1}^n W_{jk} = 1$, according to the constraint in the linear reconstruction step of LLE (see (Ghojogh et al., 2020a) for more details). According to Eq. (6), the matrix M , used in LLE, can be interpreted as the Laplacian matrix of graph of data as it satisfies its row-sum property. The optimization of LLE is (Ghojogh et al., 2020a):

$$\begin{aligned} &\underset{Y}{\text{minimize}} \quad \text{tr}(Y^\top M Y), \\ &\text{subject to} \quad \frac{1}{n} Y^\top Y = I, \\ &\quad Y^\top \mathbf{1} = \mathbf{0}, \end{aligned} \quad (91)$$

whose second constraint can be ignored because it is automatically satisfied (see (Ghojogh et al., 2020a)). Comparing Eq. (91), without its second constraint, to Eq. (76) shows that LLE is a special case of direct GE with $L = M$ and $B = (1/n)I$.

6.5. Other Improvements over Graph embedding

Some of the improved variants of graph embedding are kernel eigenmap for graph embedding with side informa-

tion (Brand, 2003), fuzzy graph embedding (Wan et al., 2017), graph embedding with extreme learning machine (Yang et al., 2019), and deep dynamic graph embedding (Goyal et al., 2018; 2020). Graph embedding has also been used for domain adaptation (Hedegaard et al., 2020a;b). A Python package for graph embedding toolbox is available (Goyal & Ferrara, 2018a).

7. Diffusion Map

Diffusion map, proposed in (Lafon, 2004; Coifman et al., 2005; Coifman & Lafon, 2006), is a nonlinear dimensionality reduction method which makes use of Laplacian of data. Note that the literature of diffusion map usually refers to Laplacian as the normalized kernel (see Section 2.2 where we explain why Laplacian is sometimes called the normalized kernel). Diffusion map reveals the nonlinear underlying manifold of data by a diffusion process, where the structure of manifold gets more obvious by passing some time on running the algorithm. Its algorithm includes several concepts explained in the following.

7.1. Discrete Time Markov Chain

Consider the graph of data, denoted by $\mathcal{X} = \{x_1, \dots, x_n\}$ where $x_i \in \mathbb{R}^d$ is the i -th data point in the dataset. Consider a k NN graph, or an adjacency matrix, of data. Starting from any point in the graph, we can randomly traverse the data graph to move to another point. This process is known as a random walk on the graph of data (De la Porte et al., 2008). Let $\mathbb{P}(x_i, x_j)$ denote the probability of moving from point x_i to x_j . Obviously, if the two points are closer to one another in the graph or adjacency matrix, i.e. if they are more similar, this probability is larger. **Movements from a point to another point in this graph can be modeled as a discrete time Markov chain²** (Ross, 2014). Let M be the transition matrix of a Markov chain on \mathcal{X} where its elements are the probabilities of transition (Ghojogh et al., 2019c). We can repeat this transition for any number of times. The transition matrix at the t -th step is obtained as M^t (Ross, 2014).

Let $W \in \mathbb{R}^{n \times n}$ be the adjacency matrix. Diffusion map usually uses the RBF kernel, Eq. (2), for the adjacency matrix (Nadler et al., 2006a;b). According to Eq. (7), the Laplacian is calculated for a value of α :

$$\mathbb{R}^{n \times n} \ni L^{(\alpha)} := D^{-\alpha} W D^{-\alpha}. \quad (92)$$

Let $D^{(\alpha)} \in \mathbb{R}^{n \times n}$ denote the diagonal degree matrix for $L^{(\alpha)}$ which is calculated similarly as Eq. (5) where W is replaced by $L^{(\alpha)}$ in that equation. Applying the so-called *random-walk graph Laplacian normalization* to the Lapla-

cian matrix gives us (Chung, 1997):

$$\mathbb{R}^{n \times n} \ni M := (D^{(\alpha)})^{-1} L^{(\alpha)}, \quad (93)$$

which we take as the transition matrix of our Markov chain here. Note that Eq. (93) is a stochastic transition matrix with non-negative rows which sum to one (Brand, 2003). According to the definition of transition matrix, the (i, j) -th element of transition matrix at time step t , i.e. M^t , is:

$$\mathbb{P}(x_j, t | x_i) = M^t(i, j). \quad (94)$$

7.2. The Optimization Problem

At the time step t , diffusion map calculates the embeddings $Y \in \mathbb{R}^{n \times d}$ by maximizing $\text{tr}(Y^\top M^t Y)$ where it assumes the embeddings have scaled identity covariance, i.e. $Y^\top Y = I$. The term $\text{tr}(Y^\top M^t Y)$ shows the transition matrix of data, at time t , after projection onto the column space of Y (Ghojogh & Crowley, 2019). The optimization problem is:

$$\begin{aligned} & \underset{Y}{\text{maximize}} && \text{tr}(Y^\top M^t Y) \\ & \text{subject to} && Y^\top Y = I, \end{aligned} \quad (95)$$

whose solution is the eigenvalue problem of M^t :

$$M^t Y = Y \Lambda, \quad (96)$$

where columns of Y are the eigenvectors (Ghojogh et al., 2019a) and diagonal elements of Λ are the eigenvalues of M^t . As it is a maximization problem, the eigenvectors are sorted from the corresponding largest to smallest eigenvalues. Then, we truncate the solution by taking the p leading eigenvectors to have the p -dimensional embedding $Y \in \mathbb{R}^{n \times p}$ where $p \leq d$. Note that diffusion map usually scales the embedding using eigenvalues. This will be explained in more detail in Section 7.3.

7.3. Diffusion Distance

We can define a diffusion distance measuring how dissimilar two points are with regards to their probability of random walk from their neighbors to them. In other words, this distance, at time step t , is defined as (Nadler et al., 2006a;b):

$$d^{(t)}(x_i, x_j) := \sqrt{\sum_{\ell=1}^n \frac{(\mathbb{P}(x_\ell, t | x_i) - \mathbb{P}(x_\ell, t | x_j))^2}{\psi_1(\ell)}}, \quad (97)$$

where $\psi_1 = [\psi_1(1), \dots, \psi_1(n)] \in \mathbb{R}^n$ is the leading eigenvector with the largest eigenvalue, i.e. $Y = [\psi_1, \dots, \psi_p]$. The denominator is for the sake of normalization. It is shown in (Nadler et al., 2006b) that this dis-

²A discrete time Markov chain is a sequence of random variables where the value of the next random variable depends only on the value of the current random value and not the previous random variables in the sequence.

tance is equivalent to the original diffusion distance proposed in (Coifman & Lafon, 2006):

$$d^{(t)}(\mathbf{x}_i, \mathbf{x}_j) := \sqrt{\sum_{\ell=1}^n \lambda_{\ell}^{2t} (\psi_{\ell}(i) - \psi_{\ell}(j))^2}, \quad (98)$$

where λ_{ℓ} denotes the ℓ -th largest eigenvalue.

If, at time t , the diffusion distance of two points is small, it implies that the two points are similar and close to each other in the embedding of step t . In other words, a small diffusion map between two points means those points probably belong to the same cluster in the embedding. Note that, as was explained in Section 7.2, the diffusion map, i.e. the embedding, is computed as $\mathbf{Y} = [\mathbf{y}_1, \dots, \mathbf{y}_n]^T \in \mathbb{R}^{n \times p}$ where (Coifman & Lafon, 2006):

$$\mathbb{R}^p \ni \mathbf{y}_i^{(t)} := [\lambda_1^t \psi_1(i), \dots, \lambda_p^t \psi_p(i)]^T, \quad (99)$$

for time step t . Note that diffusion map scales the embedding using the corresponding eigenvalues to the power of time step. Hence, the diffusion map gets larger and larger by going forward in time. That is why the name of this method is diffusion map as it diffuses in time where t is the time step and plays as a parameter for the scale, too.

Moreover, it is shown in (Nadler et al., 2006b) that:

$$d^{(t)}(\mathbf{x}_i, \mathbf{x}_j) = \|\mathbf{y}_i^{(t)} - \mathbf{y}_j^{(t)}\|_2. \quad (100)$$

Therefore, the diffusion distance is equivalent to the distance of embedded points in the diffusion map.

7.4. Other Improvements over Diffusion maps

An improvement over diffusion map is the vector diffusion map (Singer & Wu, 2012), which is a generalization of diffusion map and several other nonlinear dimensionality reduction methods. Diffusion map has also been used in several applications such as data fusion (Lafon et al., 2006), nonlinear independent component analysis (Singer & Coifman, 2008), and changing data (Coifman & Hirn, 2014). The Fokker-Planck operator, which describes the time evolution of a probability density function, has also been used in some variants of diffusion maps (Nadler et al., 2005).

8. Conclusion

In this paper, we provided a tutorial and survey for the nonlinear dimensionality reduction methods which are based on the Laplacian of graph of data. We introduced the definition of Laplacian and then we covered the most important methods in this family, namely spectral clustering, Laplacian eigenmap, locality preserving projection, graph embedding, and diffusion map.

Acknowledgement

Some parts of this tutorial paper (particularly some parts of spectral clustering and Laplacian eigenmap) have been

covered by Prof. Ali Ghodsi's tutorial videos, from University of Waterloo, on YouTube.

References

- Ah-Pine, Julien. Normalized kernels as similarity indices. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pp. 362–373. Springer, 2010.
- Ahmadizadeh, Saeed, Shames, Iman, Martin, Samuel, and Nešić, Dragan. On eigenvalues of Laplacian matrix for a class of directed signed graphs. *Linear Algebra and its Applications*, 523:281–306, 2017.
- Alperin, Jonathan L. *Local representation theory: Modular representations as an introduction to the local representation theory of finite groups*, volume 11. Cambridge University Press, 1993.
- Anderson Jr, William N and Morley, Thomas D. Eigenvalues of the Laplacian of a graph. *Linear and multilinear algebra*, 18(2):141–145, 1985.
- Banijamali, Ershad and Ghodsi, Ali. Fast spectral clustering using autoencoders and landmarks. In *International Conference Image Analysis and Recognition*, pp. 380–388. Springer, 2017.
- Belhumeur, Peter N., Hespanha, Joao P, and Kriegman, David J. Eigenfaces vs. Fisherfaces: Recognition using class specific linear projection. *IEEE Transactions on pattern analysis and machine intelligence*, 19(7):711–720, 1997.
- Belkin, Mikhail and Niyogi, Partha. Laplacian eigenmaps and spectral techniques for embedding and clustering. *Advances in neural information processing systems*, 14: 585–591, 2001.
- Belkin, Mikhail and Niyogi, Partha. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural computation*, 15(6):1373–1396, 2003.
- Belkin, Mikhail and Niyogi, Partha. Towards a theoretical foundation for Laplacian-based manifold methods. In *International Conference on Computational Learning Theory*, pp. 486–500. Springer, 2005.
- Belkin, Mikhail and Niyogi, Partha. Convergence of Laplacian eigenmaps. *Advances in neural information processing systems*, 19:129–136, 2006.
- Bengio, Yoshua, Païement, Jean-françois, Vincent, Pascal, Delalleau, Olivier, Roux, Nicolas, and Ouimet, Marie. Out-of-sample extensions for LLE, Isomap, MDS, eigenmaps, and spectral clustering. *Advances in neural information processing systems*, 16:177–184, 2003a.

- Bengio, Yoshua, Vincent, Pascal, Paiement, Jean-François, Delalleau, O, Ouimet, M, and LeRoux, N. Learning eigenfunctions of similarity: linking spectral clustering and kernel PCA. Technical report, Departement d'Informatique et Recherche Operationnelle, 2003b.
- Bengio, Yoshua, Vincent, Pascal, Paiement, Jean-François, Delalleau, Olivier, Ouimet, Marie, and Le Roux, Nicolas. *Spectral clustering and kernel PCA are learning eigenfunctions*, volume 1239. Citeseer, 2003c.
- Bengio, Yoshua, Delalleau, Olivier, Roux, Nicolas Le, Paiement, Jean-François, Vincent, Pascal, and Ouimet, Marie. Learning eigenfunctions links spectral embedding and kernel PCA. *Neural computation*, 16(10): 2197–2219, 2004a.
- Bengio, Yoshua, Paiement, Jean-françois, Vincent, Pascal, Delalleau, Olivier, Roux, Nicolas L, and Ouimet, Marie. Out-of-sample extensions for LLE, Isomap, MDS, eigenmaps, and spectral clustering. In *Advances in neural information processing systems*, pp. 177–184, 2004b.
- Bengio, Yoshua, Delalleau, Olivier, Le Roux, Nicolas, Paiement, Jean-François, Vincent, Pascal, and Ouimet, Marie. Spectral dimensionality reduction. In *Feature Extraction*, pp. 519–550. Springer, 2006.
- Blaschko, Matthew B and Lampert, Christoph H. Correlational spectral clustering. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8. IEEE, 2008.
- Boyd, Stephen, Boyd, Stephen P, and Vandenberghe, Lieven. *Convex optimization*. Cambridge university press, 2004.
- Brand, Matthew. Continuous nonlinear dimensionality reduction by kernel eigenmaps. In *International Joint Conference on Artificial Intelligence*, pp. 547–554, 2003.
- Bunte, Kerstin, Biehl, Michael, and Hammer, Barbara. A general framework for dimensionality-reducing data visualization mapping. *Neural Computation*, 24(3):771–804, 2012.
- Burago, Dmitri, Ivanov, Sergei, and Kurylev, Yaroslav. A graph discretization of the Laplace–Beltrami operator. *Journal of Spectral Theory*, 4(4):675–714, 2015.
- Cai, Deng, He, Xiaofei, and Han, Jiawei. Document clustering using locality preserving indexing. *IEEE Transactions on Knowledge and Data Engineering*, 17(12): 1624–1637, 2005.
- Carreira-Perpinán, Miguel A and Lu, Zhengdong. The Laplacian eigenmaps latent variable model. In *Artificial Intelligence and Statistics*, pp. 59–66, 2007.
- Chang, Yale. Graph embedding and extensions: A general framework for dimensionality reduction. Technical report, Department of ECE, Northeastern University, 2014.
- Chen, Sibao, Zhao, Haifeng, Kong, Min, and Luo, Bin. 2D-LPP: A two-dimensional extension of locality preserving projections. *Neurocomputing*, 70(4-6):912–921, 2007.
- Chen, Wei-Jie, Li, Chun-Na, Shao, Yuan-Hai, Zhang, Ju, and Deng, Nai-Yang. 2DRLPP: Robust two-dimensional locality preserving projection with regularization. *Knowledge-Based Systems*, 169:53–66, 2019.
- Chen, Wen-Yen, Song, Yangqiu, Bai, Hongjie, Lin, Chih-Jen, and Chang, Edward Y. Parallel spectral clustering in distributed systems. *IEEE transactions on pattern analysis and machine intelligence*, 33(3):568–586, 2010.
- Cheng, Jian, Liu, Qingshan, Lu, Hanqing, and Chen, Yen-Wei. Supervised kernel locality preserving projections for face recognition. *Neurocomputing*, 67:443–449, 2005.
- Chung, Fan RK. *Spectral graph theory*. Number 92. American Mathematical Soc., 1997.
- Coifman, Ronald R and Hirn, Matthew J. Diffusion maps for changing data. *Applied and computational harmonic analysis*, 36(1):79–107, 2014.
- Coifman, Ronald R and Lafon, Stéphane. Diffusion maps. *Applied and computational harmonic analysis*, 21(1):5–30, 2006.
- Coifman, Ronald R, Lafon, Stephane, Lee, Ann B, Maggioni, Mauro, Nadler, Boaz, Warner, Frederick, and Zucker, Steven W. Geometric diffusions as a tool for harmonic analysis and structure definition of data: Diffusion maps. *Proceedings of the national academy of sciences*, 102(21):7426–7431, 2005.
- Costeira, Joao and Kanade, Takeo. A multi-body factorization method for motion analysis. In *Proceedings of IEEE International Conference on Computer Vision*, pp. 1071–1076. IEEE, 1995.
- De la Porte, J, Herbst, BM, Hereman, W, and Van Der Walt, SJ. An introduction to diffusion maps. In *Proceedings of the 19th Symposium of the Pattern Recognition Association of South Africa (PRASA 2008)*, Cape Town, South Africa, pp. 15–25, 2008.
- Dunson, David B, Wu, Hau-Tieng, and Wu, Nan. Spectral convergence of graph Laplacian and heat kernel reconstruction in L^∞ from random samples. *Applied and Computational Harmonic Analysis*, 2021.

- Fisher, Ronald A. The use of multiple measurements in taxonomic problems. *Annals of eugenics*, 7(2):179–188, 1936.
- Friedman, Jerome, Hastie, Trevor, and Tibshirani, Robert. *The elements of statistical learning*, volume 2. Springer series in statistics New York, NY, USA:, 2009.
- Ghojogh, Benyamin. *Data Reduction Algorithms in Machine Learning and Data Science*. PhD thesis, University of Waterloo, 2021.
- Ghojogh, Benyamin and Crowley, Mark. Unsupervised and supervised principal component analysis: Tutorial. *arXiv preprint arXiv:1906.03148*, 2019.
- Ghojogh, Benyamin, Karray, Fakhri, and Crowley, Mark. Eigenvalue and generalized eigenvalue problems: Tutorial. *arXiv preprint arXiv:1903.11240*, 2019a.
- Ghojogh, Benyamin, Karray, Fakhri, and Crowley, Mark. Fisher and kernel Fisher discriminant analysis: Tutorial. *arXiv preprint arXiv:1906.09436*, 2019b.
- Ghojogh, Benyamin, Karray, Fakhri, and Crowley, Mark. Hidden Markov model: Tutorial. *engrXiv*, 2019c.
- Ghojogh, Benyamin, Samad, Maria N, Mashhadi, Sayema Asif, Kapoor, Tania, Ali, Wahab, Karray, Fakhri, and Crowley, Mark. Feature selection and feature extraction in pattern analysis: A literature review. *arXiv preprint arXiv:1905.02845*, 2019d.
- Ghojogh, Benyamin, Ghodsi, Ali, Karray, Fakhri, and Crowley, Mark. Locally linear embedding and its variants: Tutorial and survey. *arXiv preprint arXiv:2011.10925*, 2020a.
- Ghojogh, Benyamin, Ghodsi, Ali, Karray, Fakhri, and Crowley, Mark. Multidimensional scaling, Sammon mapping, and Isomap: Tutorial and survey. *arXiv preprint arXiv:2009.08136*, 2020b.
- Ghojogh, Benyamin, Karray, Fakhri, and Crowley, Mark. Generalized subspace learning by Roweis discriminant analysis. In *International Conference on Image Analysis and Recognition*, pp. 328–342. Springer, 2020c.
- Gisbrecht, Andrej, Lueks, Wouter, Mokbel, Bassam, and Hammer, Barbara. Out-of-sample kernel extensions for nonparametric dimensionality reduction. In *European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, volume 2012, pp. 531–536, 2012.
- Gisbrecht, Andrej, Schulz, Alexander, and Hammer, Barbara. Parametric nonlinear dimensionality reduction using kernel t-sne. *Neurocomputing*, 147:71–82, 2015.
- Goyal, Palash and Ferrara, Emilio. GEM: a python package for graph embedding methods. *Journal of Open Source Software*, 3(29):876, 2018a.
- Goyal, Palash and Ferrara, Emilio. Graph embedding techniques, applications, and performance: A survey. *Knowledge-Based Systems*, 151:78–94, 2018b.
- Goyal, Palash, Kamra, Nitin, He, Xinran, and Liu, Yan. DynGEM: Deep embedding method for dynamic graphs. *arXiv preprint arXiv:1805.11273*, 2018.
- Goyal, Palash, Chhetri, Sujit Rokka, and Canedo, Arquimedes. dyngraph2vec: Capturing network dynamics using dynamic graph representation learning. *Knowledge-Based Systems*, 187:104816, 2020.
- Guan-zhong, CAI Xiao-yan DAI and Li-bin, YANG. Survey on spectral clustering algorithms [j]. *Computer Science*, 7(005), 2008.
- Guo, Cuimei, Zheng, Sheng, Xie, Yaocheng, and Hao, Wei. A survey on spectral clustering. In *World Automation Congress 2012*, pp. 53–56. IEEE, 2012.
- Guo, Yi, Gao, Junbin, and Kwan, Paul WH. Kernel Laplacian eigenmaps for visualization of non-vectorial data. In *Australasian Joint Conference on Artificial Intelligence*, pp. 1179–1183. Springer, 2006.
- He, Xiaofei and Niyogi, Partha. Locality preserving projections. In *Advances in neural information processing systems*, pp. 153–160, 2004.
- He, Xiaofei, Yan, Shuicheng, Hu, Yuxiao, and Zhang, Hong-Jiang. Learning a locality preserving subspace for visual recognition. In *Proceedings Ninth IEEE International Conference on Computer Vision*, pp. 385–392. IEEE, 2003.
- He, Xiaofei, Yan, Shuicheng, Hu, Yuxiao, Niyogi, Partha, and Zhang, Hong-Jiang. Face recognition using Laplacianfaces. *IEEE transactions on pattern analysis and machine intelligence*, 27(3):328–340, 2005.
- Hedegaard, Lukas, Sheikh-Omar, Omar Ali, and Iosifidis, Alexandros. Supervised domain adaptation using graph embedding. *arXiv preprint arXiv:2003.04063*, 2020a.
- Hedegaard, Lukas, Sheikh-Omar, Omar Ali, and Iosifidis, Alexandros. Supervised domain adaptation: A graph embedding perspective and a rectified experimental protocol. *arXiv e-prints*, pp. arXiv–2004, 2020b.
- Hein, Matthias, Audibert, Jean-Yves, and Von Luxburg, Ulrike. From graphs to manifolds—weak and strong pointwise consistency of graph Laplacians. In *International Conference on Computational Learning Theory*, pp. 470–485. Springer, 2005.

- Hein, Matthias, Audibert, Jean-Yves, and Luxburg, Ulrike von. Graph Laplacians and their convergence on random neighborhood graphs. *Journal of Machine Learning Research*, 8(6), 2007.
- Hofmann, Thomas, Schölkopf, Bernhard, and Smola, Alexander J. Kernel methods in machine learning. *The annals of statistics*, pp. 1171–1220, 2008.
- Hu, Dewen, Feng, Guiyu, and Zhou, Zongtan. Two-dimensional locality preserving projections (2DLPP) with its application to palmprint recognition. *Pattern recognition*, 40(1):339–342, 2007.
- Kelner, Jonathan. An algorithm’s toolkit: Properties of the Laplacian, positive semidefinite matrices, spectra of common graphs, and connection to the continuous Laplacian. Technical report, Department of Mathematics and CSAIL, MIT University, 2007.
- Kumar, Abhishek and Daumé, Hal. A co-training approach for multi-view spectral clustering. In *Proceedings of the 28th international conference on machine learning*, pp. 393–400, 2011.
- Kumar, Abhishek, Rai, Piyush, and Daume, Hal. Co-regularized multi-view spectral clustering. In *Advances in neural information processing systems*, pp. 1413–1421, 2011.
- Lafon, Stephane, Keller, Yosi, and Coifman, Ronald R. Data fusion and multicue data matching by diffusion maps. *IEEE Transactions on pattern analysis and machine intelligence*, 28(11):1784–1797, 2006.
- Lafon, Stéphane S. *Diffusion maps and geometric harmonics*. PhD thesis, Yale University, 2004.
- Li, Bo, Li, Yan-Rui, and Zhang, Xiao-Long. A survey on Laplacian eigenmaps based manifold learning methods. *Neurocomputing*, 335:336–351, 2019.
- Li, Jun-Bao, Pan, Jeng-Shyang, and Chu, Shu-Chuan. Kernel class-wise locality preserving projection. *Information Sciences*, 178(7):1825–1835, 2008.
- Lombaert, Herve, Zikic, Darko, Criminisi, Antonio, and Ayache, Nicholas. Laplacian forests: semantic image segmentation by guided bagging. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 496–504. Springer, 2014.
- Marsden, Anne. Eigenvalues of the Laplacian and their relationship to the connectedness of a graph. *University of Chicago, REU*, 2013.
- Merris, Russell. Laplacian matrices of graphs: a survey. *Linear algebra and its applications*, 197:143–176, 1994.
- Mika, Sebastian, Rätsch, Gunnar, Weston, Jason, Schölkopf, Bernhard, and Müller, Klaus-Robert. Fisher discriminant analysis with kernels. In *Proceedings of the 1999 IEEE signal processing society workshop on Neural networks for signal processing IX*, pp. 41–48. IEEE, 1999.
- Nadler, Boaz, Lafon, Stephane, Coifman, Ronald R, and Kevrekidis, Ioannis G. Diffusion maps, spectral clustering and eigenfunctions of Fokker-Planck operators. *Advances in neural information processing systems*, 18, 2005.
- Nadler, Boaz, Lafon, Stéphane, Coifman, Ronald R, and Kevrekidis, Ioannis G. Diffusion maps, spectral clustering and reaction coordinates of dynamical systems. *Applied and Computational Harmonic Analysis*, 21(1): 113–127, 2006a.
- Nadler, Boaz, Lafon, Stephane, Kevrekidis, Ioannis, and Coifman, Ronald R. Diffusion maps, spectral clustering and eigenfunctions of Fokker-Planck operators. In *Advances in neural information processing systems*, pp. 955–962, 2006b.
- Nascimento, Maria CV and De Carvalho, Andre CPLF. Spectral methods for graph clustering—a survey. *European Journal of Operational Research*, 211(2):221–231, 2011.
- Ng, Andrew, Jordan, Michael, and Weiss, Yair. On spectral clustering: Analysis and an algorithm. *Advances in neural information processing systems*, 14:849–856, 2001.
- Perona, Pietro and Freeman, William. A factorization approach to grouping. In *European Conference on Computer Vision*, pp. 655–670. Springer, 1998.
- Polito, Marzia and Perona, Pietro. Grouping and dimensionality reduction by locally linear embedding. In *Advances in neural information processing systems*, pp. 1255–1262, 2002.
- Ross, Sheldon M. *Introduction to probability models*. Academic press, 2014.
- Roweis, Sam T and Saul, Lawrence K. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000.
- Roychowdhury, Shounak. Robust Laplacian eigenmaps using global information. In *2009 AAAI Fall Symposium Series*. Citeseer, 2009.
- Saul, Lawrence K and Roweis, Sam T. Think globally, fit locally: unsupervised learning of low dimensional manifolds. *Journal of machine learning research*, 4(Jun): 119–155, 2003.

- Scott, Guy L and Longuet-Higgins, Hugh Christopher. Feature grouping by “relocalisation” of eigenvectors of the proximity matrix. In *BMVC*, pp. 1–6, 1990.
- Shi, Jianbo and Malik, Jitendra. Normalized cuts and image segmentation. In *Proceedings of IEEE computer society conference on computer vision and pattern recognition*, pp. 731–737. IEEE, 1997.
- Shi, Jianbo and Malik, Jitendra. Normalized cuts and image segmentation. *IEEE Transactions on pattern analysis and machine intelligence*, 22(8):888–905, 2000.
- Shikkenawis, Gitam and Mitra, Suman K. Improving the locality preserving projection for dimensionality reduction. In *2012 Third International Conference on Emerging Applications of Information Technology*, pp. 161–164. IEEE, 2012.
- Shikkenawis, Gitam and Mitra, Suman K. On some variants of locality preserving projection. *Neurocomputing*, 173:196–211, 2016.
- Singer, Amit. From graph to manifold Laplacian: The convergence rate. *Applied and Computational Harmonic Analysis*, 21(1):128–134, 2006.
- Singer, Amit and Coifman, Ronald R. Non-linear independent component analysis with diffusion maps. *Applied and Computational Harmonic Analysis*, 25(2):226–239, 2008.
- Singer, Amit and Wu, H-T. Vector diffusion maps and the connection laplacian. *Communications on pure and applied mathematics*, 65(8):1067–1144, 2012.
- Strang, Gilbert. *Differential equations and linear algebra*. Wellesley-Cambridge Press Wellesley, 2014.
- Tang, Yun and Rose, Richard. A study of using locality preserving projections for feature extraction in speech recognition. In *2008 IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 1569–1572. IEEE, 2008.
- Tenenbaum, Joshua B, De Silva, Vin, and Langford, John C. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.
- Trillos, Nicolás García, Gerlach, Moritz, Hein, Matthias, and Slepčev, Dejan. Error estimates for spectral convergence of the graph Laplacian on random geometric graphs toward the Laplace–Beltrami operator. *Foundations of Computational Mathematics*, 20(4):827–887, 2020.
- Turk, Matthew and Pentland, Alex. Eigenfaces for recognition. *Journal of cognitive neuroscience*, 3(1):71–86, 1991.
- Von Luxburg, Ulrike. A tutorial on spectral clustering. *Statistics and computing*, 17(4):395–416, 2007.
- Von Luxburg, Ulrike, Belkin, Mikhail, and Bousquet, Olivier. Consistency of spectral clustering. *The Annals of Statistics*, pp. 555–586, 2008.
- Wan, Minghua, Lai, Zhihui, Yang, Guowei, Yang, Zhangjing, Zhang, Fanlong, and Zheng, Hao. Local graph embedding based on maximum margin criterion via fuzzy set. *Fuzzy Sets and Systems*, 318:120–131, 2017.
- Wang, Boyue, Hu, Yongli, Gao, Junbin, Sun, Yanfeng, Chen, Haoran, and Yin, Baocai. Locality preserving projections for Grassmann manifold. *arXiv preprint arXiv:1704.08458*, 2017.
- Weiss, Yair. Segmentation using eigenvectors: a unifying view. In *Proceedings of the seventh IEEE international conference on computer vision*, volume 2, pp. 975–982. IEEE, 1999.
- Weiss, Yair, Torralba, Antonio, and Fergus, Rob. Spectral hashing. *Advances in neural information processing systems*, 21:1753–1760, 2008.
- Wiskott, Laurenz and Schönfeld, Fabian. Laplacian matrix for dimensionality reduction and clustering. In *European Big Data Management and Analytics Summer School*, pp. 93–119. Springer, 2019.
- Wong, Wai Keung and Zhao, HT. Supervised optimal locality preserving projection. *Pattern Recognition*, 45(1):186–197, 2012.
- Xu, Yong, Feng, Ge, and Zhao, Yingnan. One improvement to two-dimensional locality preserving projection method for use with face recognition. *Neurocomputing*, 73(1-3):245–249, 2009.
- Yan, Shuicheng, Xu, Dong, Zhang, Benyu, and Zhang, Hong-Jiang. Graph embedding: A general framework for dimensionality reduction. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’05)*, volume 2, pp. 830–837. IEEE, 2005.
- Yan, Shuicheng, Xu, Dong, Zhang, Benyu, Zhang, Hong-Jiang, Yang, Qiang, and Lin, Stephen. Graph embedding and extensions: A general framework for dimensionality reduction. *IEEE transactions on pattern analysis and machine intelligence*, 29(1):40–51, 2006.

- Yang, Le, Song, Shiji, Li, Shuang, Chen, Yiming, and Huang, Gao. Graph embedding-based dimension reduction with extreme learning machine. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 2019.
- Yang, Yifang and Wang, Yuping. Fuzzy partition based similarity measure for spectral clustering. *International Journal of Signal Processing, Image Processing and Pattern Recognition*, 9(10):417–428, 2016.
- Yang, Yifang, Wang, Yuping, and Xue, Xingsi. A novel spectral clustering method with superpixels for image segmentation. *Optik*, 127(1):161–167, 2016.
- Yang, Yifang, Wang, Yuping, and Xue, Xingsi. Discriminant sparse locality preserving projection for face recognition. *Multimedia Tools and Applications*, 76(2):2697–2712, 2017.
- Ye, Jieping. Least squares linear discriminant analysis. In *Proceedings of the 24th international conference on machine learning*, pp. 1087–1093. ACM, 2007.
- Yin, Hongwei, Li, Fanzhang, Zhang, Li, and Zhang, Zhao. Multi-view clustering via spectral embedding fusion. *Soft Computing*, 23(1):343–356, 2019.
- Yin, Jun and Sun, Shiliang. Multiview uncorrelated locality preserving projection. *IEEE transactions on neural networks and learning systems*, 2019.
- Yu, Weiwei, Teng, Xiaolong, and Liu, Chongqing. Face recognition using discriminant locality preserving projections. *Image and Vision computing*, 24(3):239–248, 2006.
- Zelnik-Manor, Lihi and Perona, Pietro. Self-tuning spectral clustering. *Advances in neural information processing systems*, 17:1601–1608, 2004.
- Zhang, Limei, Qiao, Lishan, and Chen, Songcan. Graph-optimized locality preserving projections. *Pattern Recognition*, 43(6):1993–2002, 2010.