Int. J. Advance Soft Compu. Appl, Vol. 9, No. 2, July 2017
ISSN 2074-8523

Applications of (SPARSE)-PCA and LAPLACIAN EIGENMAPS to Biological Network Inference Problem using Gene Expression Data

Loc Hoang Tran and Linh Hoang Tran

University of New Brunswick
Ho Chi Minh City University of Technology-VNU HCM
e-mail: tran0398@umn.edu, linht@pdx.edu

Abstract

Sparse PCA and non-linear dimensional reduction methods have been developed and studied in depth for almost two decades. Their applications are huge such as speech recognition and face recognition. However, the sparse PCA and the non-linear dimensional reduction methods have not been applied to biological network inference problem. Thus, in this paper, we propose two new non-linear dimensional reduction methods which are un-normalized Laplacian Eigenmaps algorithm and symmetric normalized Laplacian Eigenmaps algorithm and the sparse PCA algorithm and apply these three new methods to the biological network inference problem using gene expression data. Experimental results show that the combination of Laplacian Eigenmaps methods and the un-supervised learning method and the combination of the sparse PCA method and the un-supervised learning method outperform the un-supervised learning method alone in terms of accuracy performance measures.

Keywords: *un-supervised learning, direct method, PCA, sparse PCA, Laplacian Eigenmaps, biological network inference.*

1 Introduction

Many phenomena in the world can be represented by sets of objects and sets of relationships among the objects. Sets of such relationships create networks. The problem predicting links of the networks is called the link prediction problem, which is one important task in data mining research area. A classic setting of the link prediction problem is to infer the unknown parts of the network from the

known parts of the network. Link prediction problem has many applications in many fields such as social network analysis [1,2] or bio-informatics [3,4]. For example, link prediction is used to predict the friendships among participants in the social network or link prediction is used in recommender system [5,6] of www.amazon.com. In the bio-informatics research field, link prediction is typically used to predict the interactions among proteins. This will lead to the experimental designs discovering new biological facts.

The link prediction problem can be viewed as the problem of completing an adjacency matrix representing the structure of the network. One of the typical approaches to the link prediction problem is to consider it as the binary classification problem of the elements of the adjacency matrix. In this paper, the approach solving the link prediction problem is the node-information-based approach. In the other words, this approach exploits the node information such as feature vectors of nodes. In the biological network, [3,4] exploit the gene expression profiles of genes (i.e. nodes of the network). One of the state of the art approaches for this link prediction problem is the pairwise support vector machine (i.e. the supervised approach) combining the node-wise kernel to construct the pairwise kernel [7,8,9]. Recently, in 2009, the semi-supervised approach [10], which utilizes the success of graph based semi-supervised learning methods in data mining field, is also proposed to solve this link prediction problem.

In this paper, we will focus on the reliable inference of biological network structure from gene expression data. However, this remains a challenging problem because of the noisy and high dimensional gene expression data. This will lead to the bad performance of the un-supervised, semi-supervised, and supervised approaches solving the link prediction of biological network. One way to overcome this difficulty is to integrate the dimensional reduction methods with the un-supervised (or semi-supervised or supervised) approaches to solve the link prediction problem. In this paper, we will try to combine the dimensional reduction methods with the un-supervised approach to infer the biological network. To the best of my knowledge, this work has not been investigated up to now.

In our literature review, many dimensional reduction methods have been successfully developed and applied to various applications such as speech recognition and face recognition, to name a few. To the best of my knowledge, there are two classes of dimensional reduction methods which are the linear and the non-linear techniques [11]. Linear dimensional reduction methods assume that the data lies on or close to linear subspace of the high-dimensional ambient space. Linear dimensional reduction methods have been developed and used for a long time. For example, Principle Component Analysis (i.e. PCA) was invented in 1901 and is still the most widely used dimensional reduction methods nowadays. For example, the PCA technique is employed in and successfully applied to speech recognition research field [12] and face recognition research field [13].

In this paper, PCA is also employed to solve the biological network inference problem.

However, the PCA has two major disadvantages which are the lack of sparsity of the loading vectors and each principle component is the linear combination of all variables. From data analysis viewpoint, sparsity is necessary for reduced computational time and better generalization performance. From modeling viewpoint, although the interpretability of linear combinations is usually easy for low dimensional data, it could become much harder when the number of variables becomes large. To overcome this hardness and to introduce sparsity, many methods have been proposed such as [21,22,23,24].

In this paper, we will introduce new approach for sparse PCA using Alternating Direction Method of Multipliers (i.e. ADMM method) [25]. Then, we will try to employ the sparse PCA dimensional reduction method to solve biological network inference problem. This work, to the best of our knowledge, has not been investigated.

In the other hand, non-linear dimensional reduction methods make no assumption about the linearity and are designed to identify complex non-linear manifolds as well as linear ones. Recently, many researchers have focused on developing various non-linear dimensional reduction methods such as Kernel PCA [14], Isomap [15], Local Linear Embedding [16], Laplacian Eigenmaps [17]. In this paper, we will try to apply the Laplacian Eigenmaps to solve the link prediction problem. To the best of my knowledge, the random walk Laplacian Eigenmaps have been successfully developed and applied to multiple applications. However, the un-normalized and symmetric normalized Laplacian Eigenmaps have not yet been developed and applied to any practical applications. Hence we will try to developed these two new variants of the random walk Laplacian Eigenmaps method and apply these two new methods to the biological network inference problem.

The direct approach to the biological network inference problem is the closeness-based approach. In the other words, two genes are likely to share the same edge if their distance is small enough. In this paper, Euclidean distance is considered as the measure of closeness between two gene expression profiles. A direct approach thus predicts that there exists an edge between these two genes if their Euclidean distance is below a threshold. This direct approach is also call un-supervised learning approach since no labels are assigned to the dataset. By changing the threshold, we can get different amounts of true positives and true negatives.

In our work, we first try the direct approach to the gene expression data alone and measure its accuracy performance measure. Finally, we try to apply the dimensional reduction methods to the gene expression data and then apply the direct approach to the “recently transformed” gene expression data and measure their accuracy performance measures.

We will organize the paper as follows: Section II will present how to derive the PCA method and present the classical PCA algorithms in detail. Section III will present the Alternating Direction Method of Multipliers. Section IV will derive the sparse PCA method using the ADMM method in detail. Section V will present the sparse PCA algorithm.. Section VI will present the definitions of the un-normalized, random walk, and the symmetric normalized graph Laplacian. Section VII will introduce the three graph Laplacian Eigenmaps in detail. In section VIII, we will compare the accuracy performance measures of the three graph Laplacian Eigenmaps algorithms, the PCA algorithm, the sparse PCA algorithm, and the direct approach alone applied to the biological network inference problem. Section IX will conclude this paper and the future direction of researches will be discussed.

2 PCA Algorithms

Principle Component Analysis (i.e. PCA) is one of the most popular dimensionality reduction techniques [18]. It has several applications in many areas such as pattern recognition, computer vision, statistics, and data analysis. It employs the eigenvectors of the covariance matrix of the feature data to project on a lower dimensional subspace. This will lead to the reduction of noises and redundant features in the data and the low time complexity of the direct approach solving the biological network inference problem.

In detail, PCA method, used in this paper, convert the original set of features to a different and more compact representation keeping as much information as possible and to try to increase the performance of the direct approach, especially the accuracy of the direct approach. The dimensional reduction stage is achieved by retaining only the relevant dimensions according to one specific criteria which is maximizing the variance. This stage helps solve the problem called the curse of dimensionality. Therefore, reducing the dimensionality of the gene expression data is the most direct way solving the problems caused by high dimensionalities.

Next, we will show how to derive the PCA algorithm from maximum variance

approach. First, assume that $X \epsilon R^{p*n}$ be the gene expression data, where *p* is the dimension of the gene expression profile and *n* is the total number of genes in the

gene expression data. Hence *X* can be expressed as $[x_1|x_2|\dots|x_n]$.

Let $\mu = \frac{1}{n}\sum_{i=1}^{n} x_i$ be the mean vector of all column vectors (i.e. gene expression

profiles) $x_1, x_2, \dots, x_n$.

Let $\tilde{X} = [x_1 - \mu|x_2 - \mu|\dots|x_n - \mu]$. (1)

Next, let's project the gene expression data *X* onto the line along the unit vector $u \epsilon R^{p*1}$. The variance along this line is

$$f(u) = \frac{1}{n}\sum_{i=1}^{n}\left(u^T(x_i - \mu)\right)^2 = u^T\left(\frac{1}{n}\sum_{i=1}^{n}(x_i - \mu)(x_i - \mu)^T\right)u = u^T\tilde{X}\tilde{X}^T u \quad (2)$$

We want to maximize the function $f(u)$ under the constraint that *u* is the unit vector. In the other words, we want to solve the following maximization problem:

$$max\{u^T\tilde{X}\tilde{X}^T u - \lambda(u^T u - 1)\} \quad (3)$$

Take the derivative of $f(u) - \lambda(u^T u - 1)$ with respect to *u*, we have

$$\frac{d(f(u) - \lambda(u^T u - 1))}{du} = 2\tilde{X}\tilde{X}^T u - 2\lambda u \quad (4)$$

Set this amount to zero, we have the equation 5:

$$\tilde{X}\tilde{X}^T u = \lambda u \quad (5)$$

Thus, *u* is the principle eigenvector of the covariance matrix $\tilde{X}\tilde{X}^T$.

Finally, we will present the PCA algorithm

Algorithm 1: PCA algorithm

---

1. Input: The gene expression data $X \epsilon R^{p*n}$, where *p* is the dimension of the gene expression profile and *n* is the total number of genes in the gene expression data

2. Compute $\tilde{X} = [x_1 - \mu | x_2 - \mu | \dots | x_n - \mu]$, where $\mu = \frac{1}{n}\sum_{i=1}^{n} x_i$ be the mean vector of all column vectors (i.e. gene expression profiles) $x_1, x_2, \dots, x_n$ of *X*

3. Compute the covariance matrix $\tilde{X}\tilde{X}^T$

4. Compute $U_k = [u_1|u_2|...|u_k]$ be the matrix with orthonormal columns, which are the eigenvectors associated with *k* largest eigenvalues of the covariance matrix (please note that $k < m$)

5. Output: The matrix $U_k^T X$

From the above algorithm, we easily recognize that PCA algorithm has many advantages such as finding feature groups that are highly correlated and supporting the feature extraction, outlier detection, and clustering process (i.e. by reducing noise and redundant features of the datasets). However, PCA algorithm is the linear dimensional reduction method. This is the major disadvantage of PCA algorithm.

3 Alternating Direction Method of Multipliers

In this section, we will introduce the Alternating Direction Method of Multipliers. The detailed information about the Alternating Direction Method of Multipliers can be found in [25]. First, assume that we want to solve the following problem

$$minimize f(x) + g(z) \tag{6}$$

$$subject\ to\ Ax + Bz = c \tag{7}$$

with variables $x \in R^n$ and $z \in R^m$, where $A \in R^{p*n}, B \in R^{p*m}$.

Next, we will form the augmented Lagrangian

$$L_\rho(x, z, y) = f(x) + g(z) + y^T(Ax + Bz - c) + \frac{\rho}{2}||Ax + Bz - c||_2^2 \tag{8}$$

Finally, $x^{k+1}, z^{k+1},\ and\ y^{k+1}$ can be solved as the followings

$$x^{k+1} = argmin_x L_\rho(x, z^k, y^k) \tag{9}$$

$$z^{k+1} = argmin_z L_\rho(x^{k+1}, z, y^k) \tag{10}$$

$$y^{k+1} = y^k + \rho(Ax^{k+1} + Bz^{k+1} - c), \tag{11}$$

where $\rho > 0$.

4 Sparse Principle Component Analysis Derivation

Assume that we are given the data matrix $X \in R^{p*n}$. Next, we will formulate our sparse PCA problem. This problem is in fact the following optimization problem:

$$minimize_{v,z}||X - \sigma uv^T||_F^2 + \frac{\mu}{2}||v||_2^2 + \lambda||z||_1 \tag{12}$$

$$such\ that\ v = z,$$

where σ, u, v are the singular value, the left singular vector, and the right singular vector of the Singular Value Decomposition (i.e. SVD) of X respectively. Information about the SVD and its relationship to PCA can be found in [18]. In the above optimization problem, $\sigma\ and\ u$ are fixed. Our objective is to find the sparse loading vectors v.

First, the augmented Lagrangian of the above optimization problem can be derived as the following Equation 13-34:

$$L_\rho(x,z,y) = ||X - \sigma uv^T||_F^2 + \frac{\mu}{2}||v||_2^2 + \lambda||z||_1 + y^T(v-z) + \frac{\rho}{2}||v-z||_2^2 \quad (13)$$

Then $v^{k+1}, z^{k+1}, and\ y^{k+1}$ can be solved as the followings

$$v^{k+1} = argmin_v L_\rho(v, z^k, y^k) \quad (14)$$

Hence

$$\frac{dv^{k+1}}{dv} = \frac{d}{dv}(||X - \sigma uv^T||_F^2 + \frac{\mu}{2}||v||_2^2 + y^{k^T}(v-z^k) + \frac{\rho}{2}||v-z^k||_2^2) \quad (15)$$

$$= \frac{d}{dv}(\sum_{i,j}(X_{ij} - (\sigma uv^T)_{ij})^2 + \frac{\mu}{2}||v||_2^2 + y^{k^T}(v-z^k) + \frac{\rho}{2}||v-z^k||_2^2) \quad (16)$$

$$= \frac{d}{dv}(\sum_{i,j}(X_{ij}^2 - 2X_{ij}(\sigma uv^T)_{ij} + (\sigma uv^T)_{ij}^2) + \frac{\mu}{2}||v||_2^2 + y^{k^T}(v-z^k) + \frac{\rho}{2}||v-z^k||_2^2) \quad (17)$$

$$= \frac{d}{dv}\left(||X||_F^2 - 2\sum_{ij}X_{ij}(\sigma uv^T)_{ij} + ||\sigma uv^T||_F^2 + \frac{\mu}{2}||v||_2^2 + y^{k^T}(v-z^k) + \frac{\rho}{2}||v-z^k||_2^2\right) \quad (18)$$

$$= \frac{d}{dv}(-2\sigma\sum_j\sum_i X_{ij}u_iv_j + \sigma^2 trace(vv^T) + \frac{\mu}{2}||v||_2^2 + y^{k^T}(v-z^k) + \frac{\rho}{2}||v-z^k||_2^2) \quad (19)$$

$$= \frac{d}{dv}(-2\sigma\sum_j(X^Tu)_jv_j + \sigma^2 trace(vv^T) + \frac{\mu}{2}||v||_2^2 + y^{k^T}(v-z^k) + \frac{\rho}{2}||v-z^k||_2^2) \quad (20)$$

$$= \frac{d}{dv}\left(-2\sigma v^T X^T u + \sigma^2 trace(vv^T) + \frac{\mu}{2}||v||_2^2 + y^{kT}(v - z^k) + \frac{\rho}{2}||v - z^k||_2^2\right) \quad (19)$$

$$= -2\sigma X^T u + 2\sigma^2 v + \mu v + y^k + \rho(v - z^k) \quad (20)$$

Next, we solve $\frac{dv^{k+1}}{dv} = 0 \Leftrightarrow (2\sigma^2 + \mu + \rho)v = 2\sigma X^T u - y^k + \rho z^k$ (21)

Thus, $v^{k+1} = \frac{2\sigma X^T u - y^k + \rho z^k}{(2\sigma^2 + \mu + \rho)}$ (22)

Next, we have

$$z^{k+1} = argmin_v L_\rho(v^{k+1}, z, y^k) \quad (23)$$

Hence

$$\frac{dz^{k+1}}{dz} = \frac{d}{dv}\left(\lambda||z||_1 + y^T(v - z) + \frac{\rho}{2}||v - z||_2^2\right) \quad (24)$$

$$= \lambda\xi - y^k + \rho(v^{k+1} - z)(-1) \quad (25)$$

$$= \lambda\xi - y^k + \rho(z - v^{k+1}), \quad (26)$$

where

$$\xi_i = \begin{cases} 1 \; if \; z_i > 0 \\ [-1,1] \; if \; z_i = 0 \\ -1 \; if \; z_i < 0 \end{cases} \quad (27)$$

Solve $\frac{dz^{k+1}}{dz} = 0$, we have

$$z_i^{k+1} = v_i^{k+1} + \frac{1}{\rho}y_i^k - \frac{\lambda}{\rho}\xi_i \quad (28)$$

If $z_i^{k+1} > 0, \xi_i = 1$, then

$$v_i^{k+1} + \frac{1}{\rho}y_i^k - \frac{\lambda}{\rho} > 0 \Longrightarrow v_i^{k+1} + \frac{1}{\rho}y_i^k > \frac{\lambda}{\rho} \quad (29)$$

If $z_i^{k+1} < 0, \xi_i = -1$, then

$$v_i^{k+1} + \frac{1}{\rho}y_i^k + \frac{\lambda}{\rho} < 0 \Longrightarrow v_i^{k+1} + \frac{1}{\rho}y_i^k < -\frac{\lambda}{\rho} \quad (30)$$

If $z_i^{k+1} = 0$, then

$$-\frac{\lambda}{\rho} \leq v_i^{k+1} + \frac{1}{\rho}y_i^k \leq \frac{\lambda}{\rho} \quad (31)$$

Thus,

$$z_i = sign(v_i^{k+1} + \frac{1}{\rho} y_i^k)\max(\left|v_i^{k+1} + \frac{1}{\rho} y_i^k\right| - \frac{\lambda}{\rho}, 0) \quad (32)$$

Finally, we have

$$y^{k+1} = y^k + \rho(v^{k+1} - z^{k+1}) \quad (33)$$

5 Sparse Principle Component Analysis algorithm

In this section, we will present the sparse PCA algorithm

Algorithm 2: Sparse PCA algorithm

1. Input: The dataset $X \epsilon R^{p*n}$, where *p* is the dimension of the dataset and *n* is the total number of observations in the dataset
2. Compute $\tilde{X} = [x_1 - \mu | x_2 - \mu | \dots | x_n - \mu]$, where $\mu = \frac{1}{n}\sum_{i=1}^{n} x_i$ be the mean vector of all column vectors $x_1, x_2, \dots, x_n$ of *X*
3. Randomly select parameters ρ, μ, λ.
4. Set $V = zeros(n, dim)$
5. for $i = 1{:}\, dim$
    i. Compute the SVD of $\tilde{X}$
    ii. Initialize $v^0,\ z^0, y^0$
    iii. Set $k = 0$
    iv. do
        a. Compute $v^{k+1} = argmin_v L_\rho(v, z^k, y^k)$
        b. Compute $z^{k+1} = argmin_v L_\rho(v^{k+1}, z, y^k)$
        c. Compute $y^{k+1} = y^k + \rho(v^{k+1} - z^{k+1})$

d. $k = k + 1$

v. while $||v^{k+1} - v^k|| > 10^{-10}$

vi. $v = \frac{v^{k+1}}{norm(v^{k+1})}$

vii. $V(:, i) = v$

viii. $\tilde{X} = \tilde{X}(I - vv^T)$

6. End
7. Output: The matrix *V*.

---

6 Definitions of Graph Laplacians

Given a graph *G=(V,E)*, where *V* is the set of vertices and *E* is the set of edges. Let *w(i,j)* be the weight of the edge *(i,j)*. Then *W* will be the $R^{|V|*|V|}$ matrix containing the weights of all edges of graph *G*. *W* is also called the weighted adjacency matrix of the graph *G*. Please note that $|V| = n$ is the total number of genes in the gene expression data.

Next, we can define the degree of vertex $i \epsilon V$ as follows

$$d(i) = \sum_j w(i,j) \tag{35}$$

Let D be diagonal matrix containing the degrees of vertices in its diagonal entries. Please note that D is the $R^{|V|*|V|}$ matrix.

Definition 1: Un-normalized graph Laplacian
The un-normalized graph Laplacian is defined as follows $$L = D - W$$
Definition 2: Symmetric normalized graph Laplacian
The symmetric normalized graph Laplacian is defined as follows $$L_{sym} = I - D^{-\frac{1}{2}} W D^{-\frac{1}{2}}$$
Definition 3: Random walk graph Laplacian
The random walk graph Laplacian is defined as follows $$L_{rw} = I - D^{-1} W$$

7 Laplacian Eigenmaps Algorithms

Laplacian Eigenmaps algorithms will be our next discussion. We will spend time to discuss the key features of these algorithms. In [17], Belkin and the co-authors propose an approach building a graph incorporating neighborhood information of the dataset. Then by using the graph Laplacian, they compute the low dimensional representation of the dataset that optimally preserves local neighborhood information. This algorithm is very closely related to the spectral clustering techniques used in machine learning and computer vision research field [19].

In specific, Laplacian Eigenmaps algorithm, originally derived from [17], try to solve the generalized eigenvalue problem:

$$Lf = \lambda Df \ (*), \tag{36}$$

where L is the un-normalized graph Laplacian, D is the diagonal matrix containing the degrees of vertices in its diagonal entries, and (λ, f) is the eigenvalue-eigenvector pair of the generalized eigenvalue problem (*).

Then this generalized eigenvalue problem (*) will lead to two different eigenvalue problems which are

$$D^{-1}Lf = \lambda f \Leftrightarrow (I - D^{-1}W)f = \lambda f \Leftrightarrow L_{rw}f = \lambda f(**) \tag{37}$$

and

$$D^{-\frac{1}{2}}Lf = \lambda D^{\frac{1}{2}}f \Leftrightarrow D^{-\frac{1}{2}}LD^{-\frac{1}{2}}u = \lambda u \Leftrightarrow \left(I - D^{-\frac{1}{2}}WD^{-\frac{1}{2}}\right)u = \lambda u \Leftrightarrow L_{sym}u = \lambda u(***) \tag{38}$$

where $u = D^{\frac{1}{2}}f$ which implies that $f = D^{-\frac{1}{2}}u$.

Obviously, solving the generalized eigenvalue problem (*) will lead to two completely different Laplacian Eigenmaps algorithms. This fact has not been pointed out clearly in [17]. In detail, solving the eigenvalue problem (**) will lead to the random walk Laplacian Eigenmaps algorithm. Then, solving the eigenvalue problem (***) will lead to the symmetric normalized Laplacian Eigenmaps algorithm.

Finally, we will present the three Laplacian Eigenmaps algorithms. First, let's discuss about the random walk Laplacian Eigenmaps algorithm.

Algorithm 3: Random walk Laplacian Eigenmaps algorithm

1. Input: The gene expression data $X \in R^{p*n}$, where p is the dimension of the gene expression profile and n is the total number of genes in the gene expression data

2. Construct the similarity graph *W* from the gene expression data *X* as follows:
    a. The ε-neighborhood graph: Connect all genes whose pairwise distances are smaller than ε.
    b. k-nearest neighbor graph: Gene *i* is connected with gene *j* if gene *i* is among the k-nearest neighbor of gene *j* or gene *j* is among the k-nearest neighbor of gene *i*.
    c. The fully connected graph: All genes are connected.
3. Compute the Gaussian similarity function (i.e. the weight of the edge *(i,j)*) as follows:

$$w_{ij} = s(X(:,i), X(:,j)) = \exp\left(-\frac{d(X(:,i), X(:,j))}{t}\right)$$

4. Compute the degree matrix *D*
5. Compute the random walk graph Laplacian $L_{rw} = I - D^{-1}W$
6. Compute all eigenvalues and eigenvectors of L_{rw} and sort all eigenvalues and their corresponding eigenvector in ascending order. Pick the first k eigenvectors $v_2, v_3, \ldots, v_{k+1}$ of L_{rw} in the sorted list. *k* can be determined in the following two ways:
    a. *k* is the number of connected components of L_{rw} [19]
    b. *k* is the number such that $\frac{\lambda_{k+2}}{\lambda_{k+1}}$ or $\lambda_{k+2} - \lambda_{k+1}$ is largest for all $2 \leq k \leq n$
7. Output: $V \in R^{n*k}$ be the matrix containing the vectors $v_2, v_3, \ldots, v_{k+1}$ as columns

---

Next, we will discuss about the symmetric normalized Laplacian Eigenmaps algorithm.

Algorithm 4: Symmetric normalized Laplacian Eigenmaps algorithm

1. Input: The gene expression data $X \epsilon R^{p*n}$, where *p* is the dimension of the gene expression profile and *n* is the total number of genes in the gene expression data
2. Construct the similarity graph *W* from the gene expression data *X* as follows:
    a. The ε-neighborhood graph: Connect all genes whose pairwise distances are smaller than ε.
    b. k-nearest neighbor graph: Gene *i* is connected with gene *j* if gene *i* is among the k-nearest neighbor of gene *j* or gene *j* is among the k-nearest neighbor of gene *i*.
    c. The fully connected graph: All genes are connected.

3. Compute the Gaussian similarity function (i.e. the weight of the edge *(i,j)*) as follows:

$$w_{ij} = s(X(:,i), X(:,j)) = \exp\left(-\frac{d(X(:,i), X(:,j))}{t}\right)$$

4. Compute the degree matrix *D*
5. Compute the symmetric normalized graph Laplacian $L_{sym} = I - D^{-\frac{1}{2}} W D^{-\frac{1}{2}}$
6. Compute all eigenvalues and eigenvectors of L_{sym} and sort all eigenvalues and their corresponding eigenvector in ascending order. Pick the first k eigenvectors $v_2, v_3, \ldots, v_{k+1}$ of L_{sym} in the sorted list. *k* can be determined in the following two ways:
   a. *k* is the number of connected components of L_{sym} [19]
   b. *k* is the number such that $\frac{\lambda_{k+2}}{\lambda_{k+1}}$ or $\lambda_{k+2} - \lambda_{k+1}$ is largest for all $2 \leq k \leq n$
7. Output: $V \in R^{n*k}$ be the matrix containing the vectors $v_2, v_3, \ldots, v_{k+1}$ as columns

---

Finally, we will discuss about the un-normalized Laplacian Eigenmaps algorithm.

Algorithm 5: Un-normalized Laplacian Eigenmaps algorithm

1. Input: The gene expression data $X \epsilon R^{p*n}$, where *p* is the dimension of the gene expression profile and *n* is the total number of genes in the gene expression data
2. Construct the similarity graph *W* from the gene expression data *X* as follows:
   a. The ε-neighborhood graph: Connect all genes whose pairwise distances are smaller than ε.
   b. k-nearest neighbor graph: Gene *i* is connected with gene *j* if gene *i* is among the k-nearest neighbor of gene *j* or gene *j* is among the k-nearest neighbor of gene *i*.
   c. The fully connected graph: All genes are connected.
3. Compute the Gaussian similarity function (i.e. the weight of the edge *(i,j)*) as follows:

$$w_{ij} = s(X(:,i), X(:,j)) = \exp\left(-\frac{d(X(:,i), X(:,j))}{t}\right)$$

4. Compute the degree matrix *D*
5. Compute the un-normalized graph Laplacian $L = D - W$
6. Compute all eigenvalues and eigenvectors of L and sort all eigenvalues and their corresponding eigenvector in ascending order. Pick the first k

eigenvectors $v_2, v_3, \ldots, v_{k+1}$ of L in the sorted list. k can be determined in the following two ways:

a. k is the number of connected components of L [19]
b. k is the number such that $\frac{\lambda_{k+2}}{\lambda_{k+1}}$ or $\lambda_{k+2} - \lambda_{k+1}$ is largest for all $2 \leq k \leq n$

7. Output: $V \in R^{n*k}$ be the matrix containing the vectors $v_2, v_3, \ldots, v_{k+1}$ as columns

---

From the above algorithms, we easily recognize that Laplacian Eigenmaps algorithms have many advantages such as the primary algorithms are very simple to implement and the locality preserving property of Laplacian Eigenmaps makes it insensitive to noise and outliers. Finally, the Laplacian Eigenmaps algorithm is the pre-processing step of the spectral clustering technique (i.e. it supports the clustering process). In the other words, the Laplacian Eigenmaps algorithm and the k-mean clustering technique are the two major steps in the spectral clustering algorithm.

8 Experiments and Results

8.1 Datasets

In this paper, we use the **StatSeq** dataset available from [20] and the **metabolic network** dataset available from [4]. The **StatSeq** dataset contains expression levels of 100 genes over 300 samples. In the other words, we are given gene expression data ($R^{100*300}$) matrix. Moreover, we are also given the gold standard network which has 100 nodes and 284 edges. The **metabolic network** dataset also contains the gold standard network and the gene expression data. This gene expression data contains the expression levels for 668 genes over 157 samples. The gold standard network in the metabolic network dataset contains 668 nodes and 2782 edges.

8.2 Experiments

First, we apply the direct method (i.e. the un-supervised learning method) to the two datasets. Then, we will apply the PCA algorithm and the direct method to the two datasets. Next, we will apply the sparse PCA algorithm and the direct method to the two datasets. Finally, the three Laplacian Eigenmaps algorithms and the direct method will be applied to the two datasets.

We experiment the above proposed methods in terms of accuracy performance measures. The accuracy performance measure Q is given as in Equation 39:

$$Q = \frac{True\ Positive + True\ Negative}{True\ Positive + True\ Negative + False\ Positive + False\ Negative} \quad (39)$$

True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN) are defined in the following Table 1.

Table 1: Definitions of TP, TN, FP, and FN

		Predicted Label	
		Positive	Negative
Known Label	Positive	True Positive (TP)	False Negative (FN)
	Negative	False Positive (FP)	True Negative (TN)

In the experiments, the parameter *t* of the Gaussian similarity function is set to 10^{-1}. The 3-nearest neighbor graph is used to construct the similarity graph (for Laplacian Eigenmaps algorithms) from the two datasets.

For the **StatSeq** dataset, the table 2 shows the accuracy performance measures of the direct method, the PCA and the direct method, the sparse PCA and the direct method, the three Laplacian Eigenmaps and the direct method.

Table 2: Comparisons of the direct method, the PCA and the direct method, and the un-normalized Laplacian Eigenmaps and the direct method, the random walk Laplacian Eigenmaps and the direct method, the symmetric normalized Laplacian Eigenmaps and the direct method

Accuracy (%)	
Direct Method	89.24
PCA + Direct Method	89.28
Sparse PCA + Direct Method	89.60
Un-normalized Laplacian Eigenmaps + Direct Method	90.64
Random walk Laplacian Eigenmaps + Direct Method	89.72
Symmetric normalized Laplacian Eigenmaps + Direct Method	90.20

For the **metabolic network** dataset, the table 3 shows the accuracy performance measures of the direct method, the PCA and the direct method, the three Laplacian Eigenmaps and the direct method.

Table 3: Comparisons of the direct method, the PCA and the direct method, and the un-normalized Laplacian Eigenmaps and the direct method, the random walk Laplacian Eigenmaps and the direct method, the symmetric normalized Laplacian Eigenmaps and the direct method

Accuracy (%)	
Direct Method	79.62
PCA + Direct Method	79.64
Sparse PCA + Direct Method	76.96
Un-normalized Laplacian Eigenmaps + Direct Method	76.99
Random walk Laplacian Eigenmaps + Direct Method	77.06
Symmetric normalized Laplacian Eigenmaps + Direct Method	76.98

From the above Table 2 and Table 3, we easily recognize that the PCA and direct method outperform the direct method alone since the PCA method reduce the noise and the redundant features in the gene expression data. Moreover, the three Laplacian Eigenmaps methods and the direct method outperform the PCA method and the direct method since the Laplacian Eigenmaps methods are the non-linear dimensional reduction methods. Finally, we recognize that the sparse PCA and direct method outperform the PCA and direct method.

9 Conclusions

We have propose the un-supervised learning method, the combination of PCA method and the un-supervised learning method, the combination of sparse PCA method and the un-supervised learning method, the combination of three Laplacian Eigenmaps methods and the un-supervised learning method to solve the biological network inference using gene expression data in detail. In specific, we show how to derive the PCA method by using maximum variance approach and present the PCA algorithm in detail. Moreover, we also present the sparse PCA in detail. Finally, we propose two new Laplacian Eigenmaps algorithms which are the un-normalized Laplacian Eigenmaps and the symmetric normalized Laplacian Eigenmaps. Then we apply all these proposed methods to the biological network inference problems. Experimental results show that the three Laplacian Eigenmaps algorithms and the direct method outperform other methods since they reduce the noise and the redundant features of the gene expression data. Moreover, they are the non-linear dimensional reduction methods. Please note that applying these dimensional reduction methods to the biological network inference problem will also lead to the low time complexity of the direct method. Interestingly, we also recognize that the combination of sparse PCA method and the direct method outperform the combination of PCA method and the direct method.

In the future, we will try to propose the new semi-supervised learning method utilizing the normalized and random walk Kronecker product Laplacian matrices and apply it to the biological network inference problem.

Finally, in the industrial speech recognition research area, we will try to apply the sparse PCA method to the MFCC feature matrices. Then we will apply some machine learning methods such as kernel ridge regression method or graph based semi-supervised learning method to the transformed MFCC matrices to classify the speech samples. This work, to the best of our knowledge, has not been investigated up to now.

References

[1] Liben-Nowell, David, and Jon Kleinberg. "The link-prediction problem for social networks." *Journal of the American society for information science and technology* 58.7 (2007): 1019-1031.

[2] Al Hasan, Mohammad, et al. "Link prediction using supervised learning." *SDM'06: Workshop on Link Analysis, Counter-terrorism and Security*. 2006.

[3] Yamanishi, Yoshihiro, J-P. Vert, and Minoru Kanehisa. "Protein network inference from multiple genomic data: a supervised approach." *Bioinformatics*20.suppl 1 (2004): i363-i370.

[4] Yamanishi, Yoshihiro, Jean-Philippe Vert, and Minoru Kanehisa. "Supervised enzyme network inference from the integration of genomic data and chemical information." *Bioinformatics* 21.suppl 1 (2005): i468-i477.

[5] Bliss, Catherine A., et al. "An evolutionary algorithm approach to link prediction in dynamic social networks." *Journal of Computational Science* 5.5 (2014): 750-764.

[6] Huang, Zan, Xin Li, and Hsinchun Chen. "Link prediction approach to collaborative filtering." *Proceedings of the 5th ACM/IEEE-CS joint conference on Digital libraries*. ACM, 2005.

[7] Basilico, Justin, and Thomas Hofmann. "Unifying collaborative and content-based filtering." *Proceedings of the twenty-first international conference on Machine learning*. ACM, 2004.

[8] Ben-Hur, Asa, and William Stafford Noble. "Kernel methods for predicting protein–protein interactions." *Bioinformatics* 21.suppl 1 (2005): i38-i46.

[9] Oyama, Satoshi, and Christopher D. Manning. "Using feature conjunctions across examples for learning pairwise classifiers." *Machine Learning: ECML 2004*. Springer Berlin Heidelberg, 2004. 322-333.

[10Zhou, Dengyong, et al. "Learning with local and global consistency." *Advances in neural information processing systems* 16.16 (2004): 321-328.

[11]Halevy, Avner. "Extensions of laplacian eigenmaps for manifold learning." (2011).

[12]Trang, Hoang, Tran Hoang Loc, and Huynh Bui Hoang Nam. "Proposed combination of PCA and MFCC feature extraction in speech recognition

system." *Advanced Technologies for Communications (ATC), 2014 International Conference on*. IEEE, 2014.
[13]Kim, Kwang In, Keechul Jung, and Hang Joon Kim. "Face recognition using kernel principal component analysis." *Signal Processing Letters, IEEE* 9.2 (2002): 40-42.
[14]Schölkopf, Bernhard, Alexander Smola, and Klaus-Robert Müller. "Kernel principal component analysis." *Artificial Neural Networks—ICANN'97*. Springer Berlin Heidelberg, 1997. 583-588.
[15]Balasubramanian, Mukund, and Eric L. Schwartz. "The isomap algorithm and topological stability." *Science* 295.5552 (2002): 7-7.
[16]Roweis, Sam T., and Lawrence K. Saul. "Nonlinear dimensionality reduction by locally linear embedding." *Science* 290.5500 (2000): 2323-2326.
[17]Belkin, Mikhail, and Partha Niyogi. "Laplacian eigenmaps and spectral techniques for embedding and clustering." *NIPS*. Vol. 14. 2001.
[18]Kokiopoulou, E., and Y. Saad. *PCA and kernel PCA using polynomial filtering: a case study on face recognition*. Technical Report umsi-2004-213, Minnesota Supercomputer Institute, University of Minnesota, Minneapolis, MN, 2004. submitted, 2004.
[19]Von Luxburg, Ulrike. "A tutorial on spectral clustering." *Statistics and computing* 17.4 (2007): 395-416.
[20]Pinna, Andrea, et al. "StatSeq Systems Genetics Benchmark." (2012).
[21]R.E. Hausman. "Constrained multivariate analysis." Studies in the Management Sciences, 19 (1982), pp. 137–151.
[22]Vines, S. K. "Simple principal components." Journal of the Royal Statistical Society: Series C (Applied Statistics) 49.4 (2000): 441-451.
[23]Jolliffe, Ian T., Nickolay T. Trendafilov, and Mudassir Uddin. "A modified principal component technique based on the LASSO." Journal of computational and Graphical Statistics 12.3 (2003): 531-547.
[24]Zou, Hui, Trevor Hastie, and Robert Tibshirani. "Sparse principal component analysis." Journal of computational and graphical statistics 15.2 (2006): 265-286.
[25]Boyd, Stephen, et al. "Distributed optimization and statistical learning via the alternating direction method of multipliers." Foundations and Trends® in Machine Learning 3.1 (2011): 1-122.