A New Simplex Sparse Learning Model to Measure Data Similarity for Clustering

Jin Huang, Feiping Nie, Heng Huang[*]
University of Texas at Arlington
Arlington, Texas 76019, USA
huangjinsuzhou@gmail.com, feipingnie@gmail.com, heng@uta.edu

Abstract

The Laplacian matrix of a graph can be used in many areas of mathematical research and has a physical interpretation in various theories. However, there are a few open issues in the Laplacian graph construction: (i) Selecting the appropriate scale of analysis, (ii) Selecting the appropriate number of neighbors, (iii) Handling multiscale data, and, (iv) Dealing with noise and outliers. In this paper, we propose that the affinity between pairs of samples could be computed using sparse representation with proper constraints. This parameter free setting automatically produces the Laplacian graph, leads to significant reduction in computation cost and robustness to the outliers and noise. We further provide an efficient algorithm to solve the difficult optimization problem based on improvement of existing algorithms. To demonstrate our motivation, we conduct spectral clustering experiments with benchmark methods. Empirical experiments on 9 data sets demonstrate the effectiveness of our method.

1 Background and Motivation

Graph theory is an important branch in mathematics and has many applications. There are many algorithms built on graphs: (1) Clustering algorithms [Ng *et al.*, 2002; Hagen and Kahng, 1992; Shi and Malik, 2000; Nie *et al.*, 2014], (2) Dimension reduction algorithms based on manifold, such as LLE [Roweis and Saul, 2000] and Isomap [Tenenbaum *et al.*, 2000], (3) Semi-supervised learning algorithms, such as label propagation [Zhu *et al.*, 2003; Wang *et al.*, 2014], (4) Ranking algorithms, such as Page-Rank [Page *et al.*, 1999] and Hyperlink-Induced Topic Search (HITS) [Kleinberg, 1999]. Many data sets also have a natural graph structure, web-pages and hyper-link structure, protein interaction networks and social networks, just to name a few. In this paper, we limit our discussion to the data sets that can be transformed to similarity graph by simple means.

The key to many applications mentioned lies in the appropriate construction of the similarity graphs. There are various ways to set up such graphs, which depend on both the application and data sets. However, there are still a few open issues: (i) Selecting the appropriate scale of analysis, (ii) Selecting the appropriate number of neighbors, (iii) Handling multi-scale data, and, (iv) Dealing with noise and outliers. There are already a few papers [Long *et al.*, 2006; Li *et al.*, 2007] solving one of these issues. In particular, the classic paper self-tuning spectral clustering [Zelnik-Manor and Perona, 2004] by L. Zelnik-Manor et al. solves issues (i) and (iii). However, there is no single method that solves all of these issues to the best of our knowledge.

In the remainder part of this section, we first provide a brief review about commonly used similarity graph construction methods. Next, we present a concise introduction to sparse representation so as to lay out the background of our method. In Section 2, we present our motivation and formulate our Simplex Sparse Representation (SSR) objective function Eq. (10). We propose a novel accelerated projected gradient algorithm to optimize the objective function in an efficient way. Our experiment part consists of two sections, one with a synthetic data set to highlight the difference between our method and previous method, the other with real data sets to demonstrate the impressive performance of SSR. We conclude this paper with the conclusion and future work.

1.1 Different Similarity Graphs

There are many ways to transform a given set $\mathbf{x}_1,\ldots,\mathbf{x}_n$ of data points with pairwise similarities S_{ij} or pairwise distance D_{ij} into a graph. The following are a few popular ones:

1. **The ε-neighbor graph**: We connect all points whose pairwise distances are smaller than ε.
2. **k-nearest neighbor graph**: The vertex $\mathbf{x}_i$ is connected with $\mathbf{x}_j$ if $\mathbf{x}_j$ is among the k-nearest neighbors of $\mathbf{x}_i$.
3. **The fully connected graph**: All points are connected with each other as long as they have positive similarities. This construction is useful only if the similarity function models local neighborhoods. An example of such function is the Gaussian similarity function $S(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(\frac{-\|\mathbf{x}_i-\mathbf{x}_j\|^2}{2\sigma^2}\right)$, where σ controls the width of neighborhoods.

[*] Corresponding Author. This work was partially supported by NSF IIS-1117965, IIS-1302675, IIS-1344152, DBI-1356628.

Note that all the above three methods suffer from some of the open issues mentioned. The ε-neighbor graph method needs to find the appropriate ε to establish the edge connections, the k-nearest neighbor graph uses the specified k to connect neighbors for each node, the σ in Gaussian similarity method clearly determines the overall shape of the similarity graph. In short, they are not parameter free. Also, ε is clearly prone to the influence of data scale, k and σ would also be dominated by any feature scale inconsistence among these data vectors. Therefore, a parameter free and robust similarity graph construction method is desired and this is our motivation of this paper.

1.2 Introduction to Sparse Representation

Suppose we have n data vectors with size d and arrange them into columns of a training sample matrix

$$\mathbf{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_n) \in \mathbb{R}^{d\times n}.$$

Given a new observation $\mathbf{y}$, sparse representation method computes a representation vector

$$\boldsymbol{\beta} = (\beta_1, \cdots, \beta_n)^T \ \in \mathbb{R}^n \tag{1}$$

to satisfy

$$\mathbf{y} \approx \sum_{i=1}^{n} \mathbf{x}_i \beta_i = \mathbf{X}\boldsymbol{\beta} \tag{2}$$

To seek a sparse solution, it is natural to solve

$$\min_{\boldsymbol{\beta}} \|\mathbf{X}\boldsymbol{\beta} - \mathbf{y}\|_2^2 + \lambda_0 \|\boldsymbol{\beta}\|_0 \tag{3}$$

,where pseudo norm ℓ_0 counts the number of non-zero elements in $\boldsymbol{\beta}$ and $\lambda_0 > 0$ is the controlling parameter.

Recent discovery in [Donoho, 2004; Candès and Tao, 2006] found that the sparse solution in Eq. (3) could be approximated by solving the ℓ_1 minimization problem:

$$\min_{\boldsymbol{\beta}} \ \|\boldsymbol{\beta}\|_1, \ s.t.\ \mathbf{X}\boldsymbol{\beta} = \mathbf{y} \tag{4}$$

or the penalty version:

$$\min_{\boldsymbol{\beta}} \ \|\mathbf{X}\boldsymbol{\beta} - \mathbf{y}\|_2^2 + \lambda_1 \|\boldsymbol{\beta}\|_1 \tag{5}$$

This ℓ_1 problem can be solved in polynomial time by standard linear programming methods [Roweis and Saul, 2000].

In the literature, sparse representation framework is well known for its robustness to noise and outliers in image classification [Wright *et al.*, 2008]. It is noting that in such scheme, there is no restriction on the scale consistence for the data vectors. In other words, sparse representation has the potential to address the scale inconsistence and outlier issue presented in the introduction.

2 Similarity Matrix Construction with Simplex Representation

Assuming we have the data matrix $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n] \in \mathbb{R}^{d\times n}$, where each sample is (converted into) a vector of dimension d. The most popular method to compute the similarity matrix $\mathbf{S}$ is using Gaussian kernel function

$$\mathbf{S}_{ij} = \begin{cases} \exp\{\dfrac{-\|\mathbf{x}_i - \mathbf{x}_j\|_2^2}{2\sigma^2}\}, & \mathbf{x}_i, \mathbf{x}_j \text{ are neighbors} \\ 0, & \text{otherwise} \end{cases}$$

One major disadvantage of this method is that the hyper-parameter σ in the Gaussian kernel function is very sensitive and is difficult to tune in practice.

The sparse representation proposed in [Roweis and Saul, 2000] can be applied here to compute the similarity matrix $\mathbf{S}$. Specifically, the similarities $\boldsymbol{\alpha}_i \in \mathbb{R}^{n-1}$ between the i-th feature and other features are calculated by a sparse representation as follows:

$$\min_{\boldsymbol{\alpha}_i} \|\mathbf{X}_{-i}\boldsymbol{\alpha}_i - \mathbf{x}_i\|_2^2 + \lambda \|\boldsymbol{\alpha}_i\|_1 \tag{6}$$

where $\mathbf{X}_{-i} = [\mathbf{x}_1, \ldots, \mathbf{x}_{i-1}, \mathbf{x}_{i+1}, \ldots, \mathbf{x}_n] \in \mathbb{R}^{d\times(n-1)}$, i.e, the data matrix without column i.

Since the similarity matrix is usually non-negative, we can impose the non-negative constraint on problem (6) to minimize the sum of sparse representation residue error:

$$\sum_{i=1}^{n} \min_{\boldsymbol{\alpha}_i \geq 0} \|\mathbf{X}_{-i}\boldsymbol{\alpha}_i - \mathbf{x}_i\|_2^2 + \lambda \|\boldsymbol{\alpha}_i\|_1 \tag{7}$$

The parameter λ is still needed to be well tuned here. It is easy to note that Eq. (7) is the sum of n independent variables, as a result, we will limit the discussion to a single term in the following context. More importantly, when the data are shifted by constants $\mathbf{t} = [t_1, \cdots, t_d]^T$, *i.e.*, $\mathbf{x}_k = \mathbf{x}_k + \mathbf{t}$ for any k, the computed similarities will be changed. To get the shift invariant similarities, we need the following equation:

$$\left\|(\mathbf{X}_{-i} + \mathbf{t}\mathbf{1}^T)\boldsymbol{\alpha}_i - (\mathbf{x}_i + \mathbf{t})\right\|_2^2 = \|\mathbf{X}_{-i}\boldsymbol{\alpha}_i - \mathbf{x}_i\|_2^2, \tag{8}$$

which indicates $\boldsymbol{\alpha}_i^T\mathbf{1} = 1$. Imposing the constraint $\boldsymbol{\alpha}_i^T\mathbf{1} = 1$ on the problem (7), we have

$$\begin{aligned} &\min_{\boldsymbol{\alpha}_i} \|\mathbf{X}_{-i}\boldsymbol{\alpha}_i - \mathbf{x}_i\|_2^2 + \lambda \|\boldsymbol{\alpha}_i\|_1 \\ &s.t.\ \ \boldsymbol{\alpha}_i \geq 0, \boldsymbol{\alpha}_i^T 1 = 1 \end{aligned} \tag{9}$$

Interestingly, the constraints in problem (9) makes the second term constant. So problem (9) becomes

$$\begin{aligned} &\min_{\boldsymbol{\alpha}_i} \|\mathbf{X}_{-i}\boldsymbol{\alpha}_i - \mathbf{x}_i\|_2^2 \\ &s.t.\ \ \boldsymbol{\alpha}_i \geq 0, \boldsymbol{\alpha}_i^T 1 = 1 \end{aligned} \tag{10}$$

The constraints in problem (10) is simplex, so we call this kind of sparse representation as simplex representation.

Note that the above constraints (ℓ_1 ball constraints) indeed introduce sparse solution $\boldsymbol{\alpha}_i$ and have empirical success in various applications. In order to solve the Eq. (10) for high dimensional data, it is more appropriate to apply first-order methods, i.e, use function values and their (sub)gradient at each iteration. There are quite a few classic first-order methods, including gradient descent, subgradient descent, and Nesterovs optimal method [Nesterov, 2003]. In this paper, we use the accelerated projected gradient method to optimize Eq. (10). We will present the details of the optimization and the algorithm in next section.

3 Optimization Details and Algorithm

Our accelerate projected gradient method introduces an auxiliary variable to convert the objective equation into an easier one, meanwhile the auxiliary variable approximates and converges to the solution $\boldsymbol{\alpha}_i$ in an iterative manner. The following part of this section provides the details. For the convenience of notations, we use $f(\boldsymbol{\alpha}_i)$ to represent the objective function in Eq. (10) and C to represent the associated constraints. In other words, we need to solve $\min_{\boldsymbol{\alpha}_i \in C} f(\boldsymbol{\alpha}_i)$.

Let us assume that the auxiliary variable is z_i and we will implement the alternative optimization method. We first solve $\boldsymbol{\alpha}_i^0$ via solving Eq. (9) with $\lambda = 1$ without considering the constraints, then initialize $\mathbf{z}_i^0 = \boldsymbol{\alpha}_i^0$ to start the iterative process, here $\mathbf{z}_i^{t-1}$ denotes $\mathbf{z}_i$ at iteration $t-1$. $c_1 = 1$ is the initial value of Newton acceleration coefficient, which will be updated at each iteration.

Next, we start the iterative approximation process. At iteration t, we approximate $\boldsymbol{\alpha}_i$ via Taylor expansion up to second order:

$$\boldsymbol{\alpha}_i^t = \underset{\boldsymbol{\alpha}_i}{argmin}\, f(\mathbf{z}_i^{t-1}) + (\boldsymbol{\alpha}_i - \mathbf{z}_i^{t-1})^T f^{'}(\boldsymbol{\alpha}_i^{t-1}) + \frac{L}{2} \left\| \boldsymbol{\alpha}_i - \mathbf{z}^{t-1} \right\|_F^2, \tag{11}$$

With extra terms independent of $\boldsymbol{\alpha}_i$, we could write the previous objective function into a more compact form as follows:

$$\boldsymbol{\alpha}_i^t = \arg\min_{\boldsymbol{\alpha}_i \in C} \frac{L}{2} \left\| \boldsymbol{\alpha}_i - (\mathbf{z}_i^{t-1} - \frac{1}{L} f^{'}(\mathbf{z}_i^{t-1})) \right\|_2^2 \tag{12}$$

The critical step of the our algorithm is to solve the following proximal problem:

$$\begin{array}{l} \min_{\boldsymbol{\alpha}_\mathbf{i}} \frac{1}{2} \left\| \boldsymbol{\alpha}_i - \mathbf{v} \right\|_2^2, \\ s.t \;\; \boldsymbol{\alpha}_\mathbf{i} \geq 0, \boldsymbol{\alpha}^T \mathbf{1} = 1. \end{array} \tag{13}$$

Here we use $\mathbf{v}$ to represent $\mathbf{z}_i^{t-1} - \frac{1}{L} f^{'}(\mathbf{z}_i^{t-1})$ for notation convenience. We will introduce an efficient approach to solve Eq. (13) in a separate subsection below.

Then, we update the acceleration coefficient as follows:

$$c_{t+1} = \frac{1 + \sqrt{1 + 4c_t^2}}{2} \tag{14}$$

Last the auxiliary variable is updated with the formula

$$\mathbf{z}_i^t = \boldsymbol{\alpha}_i^t + \frac{c_t - 1}{c_{t+1}} (\boldsymbol{\alpha}_i^t - \boldsymbol{\alpha}_i^{t-1}) \tag{15}$$

Note that Eq. (14) and Eq. (15) together accelerate our algorithm.

The number of iteration is updated with $t = t + 1$.

We repeat the Eqs. (11)-(15) until the algorithm converges according to our criteria. Note that our algorithm converges at the order of $O(t^2)$, here t is number of iterations. We have to omit the proof due to space constrain, interested readers may refer to [Nesterov, 1983; 2003; 2005; 2007].

Our whole algorithm is summarized as follows:

```
Input: Data Matrix $X$
Output: $\boldsymbol{\alpha}_1, \ldots, \boldsymbol{\alpha}_n$
1. Initialize $\boldsymbol{\alpha}_i$ and $\mathbf{z_i}$ for $i$ from 1 to n
while Convergence Criteria Not Satisfied do
    2. Optimize $\boldsymbol{\alpha}_i$ with the induced equation 13
    using the method introduced in the corresponding
    subsection.
    3. Update coefficient $c_{t+1} = \frac{1+\sqrt{1+4c_t^2}}{2}$
    4. Update $\mathbf{z}_i^t = \boldsymbol{\alpha}_i^t + \frac{c_t - 1}{c_{t+1}}(\boldsymbol{\alpha}_i^t - \boldsymbol{\alpha}_i^{t-1})$
    5. t=t+1
end
```

Algorithm 1: Accelerated Projected Gradient Method

Here the convergence criteria is that the relative change of $\|\boldsymbol{\alpha}_i\|$, the frobenius norm of $\boldsymbol{\alpha}_i$, is less than 10^{-4}. The empirical experiments conducted in this paper show that our algorithm converges fast and always ends within 30 iteration.

One subtle but important point we need to point out here that the similarity matrix $S = [\hat{\boldsymbol{\alpha}}_1, \ldots, \hat{\boldsymbol{\alpha}}_n]$ is not necessarily symmetric. Here, $\hat{\boldsymbol{\alpha}}_i$ is the result of inserting coefficient 0 for its i-th of $\boldsymbol{\alpha}_i$, which is of dimension n. The coefficient $\boldsymbol{\alpha}_{ij}$, the similarity between $\mathbf{x}_i$ and $\mathbf{x}_j$ for X_{-i}, does not have to be equal to $\boldsymbol{\alpha}_{ji}$. Our approach to deal with this is to get the final similarity matrix

$$\mathbf{W} = \frac{\mathbf{S} + \mathbf{S}^T}{2}, \tag{16}$$

i.e, get the average of S and its transpose. This is a conventional practice and the correctness is easy to see, so we omit the proof here. After we get the similarity matrix, we could then follow the standard spectral clustering procedure, i.e, calculate the Laplacian matrix, get the corresponding spectral vectors, then calculate the clustering based on K-means clustering. The key difference between our proposed method and standard spectral clustering method is how the similarity graph is constructed. Conventional similarity graph generally is constructed with one of the methods specified in the introduction part and clearly is not parameter-free. Also, these methods clearly are prone to outlier and data scale inconsistence influence.

We would like to summarize the main advantages of our method Simplex Sparse Representation (SSR).

First It is parameter free. There is no parameter tuning for either data vector scale or the number of neighbors.

Second The simplex representation adopts the merits of sparse representation. Our method is therefore robust to scale inconsistence and outlier noise issues.

Third Our optimization algorithm is easy to implement and converges fast.

3.1 Optimization Algorithm to Eq. (13)

We write the Lagrangian function of problem (13) as

$$\frac{1}{2}\|\boldsymbol{\alpha}_i - \mathbf{v}\|_2^2 - \gamma(\boldsymbol{\alpha}_i^T \mathbf{1} - 1) - \boldsymbol{\lambda}^T \boldsymbol{\alpha}_i, \tag{17}$$

where γ is a scalar and $\boldsymbol{\lambda}$ is a Lagrangian coefficient vector, both of which are to be determined. Suppose the optimal solution to the proximal problem (13) is $\boldsymbol{\alpha}^*$, the associate Lagrangian coefficients are γ^* and λ^*. Then according to the KKT condition [Boyd and Vandenberghe, 2004], we have the following equations:

$$\begin{cases} \forall j, \quad \alpha_{ij}^* - v_j - \gamma^* - \lambda_j^* = 0 & (18) \\ \forall j, \quad \alpha_{ij}^* \geq 0 & (19) \\ \forall j, \quad \lambda_j^* \geq 0 & (20) \\ \forall j, \quad \alpha_{ij}^* \lambda_j^* = 0 & (21) \end{cases}$$

here α_{ij}^* is the j-th scalar element of vector $\boldsymbol{\alpha}_i^*$. Eq. (18) can be written as $\alpha_{ij}^* - v_j - \gamma^* 1 - \lambda_j^* = 0$. According to the constraint $\mathbf{1}^T\boldsymbol{\alpha}_i^* = 1$, we have $\gamma^* = \frac{1-\mathbf{1}^T\mathbf{v}-\mathbf{1}^T\boldsymbol{\lambda}^*}{n}$. So $\boldsymbol{\alpha}^* = (\mathbf{v} - \frac{\mathbf{1}\mathbf{1}^T}{n}\mathbf{v} + \frac{1}{n}\mathbf{1} - \frac{\mathbf{1}^T\boldsymbol{\lambda}^*}{n}\mathbf{1}) + \lambda^*$.

Denote $\bar{\lambda}^* = \frac{\mathbf{1}^T\boldsymbol{\lambda}^*}{n}$ and $\mathbf{u} = \mathbf{v} - \frac{\mathbf{1}\mathbf{1}^T}{n}\mathbf{v} + \frac{1}{n}\mathbf{1}$, then we can write $\boldsymbol{\alpha}^* = \mathbf{u} + \boldsymbol{\lambda}^* - \bar{\lambda}^*\mathbf{1}$. So $\forall j$ we have

$$\alpha_{ij}^* = u_j + \lambda_j^* - \bar{\lambda}^*. \tag{22}$$

According to Eq. (19)-(22) we know $u_j + \lambda_j^* - \bar{\lambda}^* = (u_j - \bar{\lambda}^*)_+$, here $x_+ = max(x, 0)$. Then we have

$$\alpha_j^* = (u_j - \bar{\lambda}^*)_+. \tag{23}$$

So we can obtain the optimal solution $\boldsymbol{\alpha}^*$ if we know $\bar{\boldsymbol{\lambda}}^*$.

We write Eq.(22) as $\lambda_j^* = \alpha_{ij}^* + \bar{\lambda}^* - u_j$. Similarly, according to Eqs. (19)-(21) we know $\lambda_j^* = (\bar{\lambda}^* - u_j)_+$. As $\mathbf{v}$ is a $n-$1-dimensional vector, we have $\bar{\boldsymbol{\lambda}}^* = \frac{1}{n-1}\sum_{j=1}^{n-1}(\bar{\lambda}^* - u_j)_+$. Defining a function as

$$f(\bar{\lambda}) = \frac{1}{n-1}\sum_{i=1}^{n-1}(\bar{\lambda} - u_j)_+ - \bar{\lambda}, \tag{24}$$

so $f(\bar{\lambda}^*) = 0$ and we can solve the root finding problem to obtain $\bar{\lambda}^*$.

Note that $\lambda^* \geq 0$, $f'(\bar{\lambda}) \leq 0$ and $f'(\bar{\lambda})$ is a piecewise linear and convex function, we can use Newton method to find the root of $f(\bar{\lambda}) = 0$ efficiently, i.e,

$$\bar{\lambda}_{t+1} = \bar{\lambda}_t - \frac{f(\bar{\lambda}_t)}{f'(\bar{\lambda}_t)}$$

4 Experiments on Synthetic Data Sets

In this part, we would like to compare our proposed method with Self-Tuning Spectral Clustering (STSC) proposed in [Zelnik-Manor and Perona, 2004]. STSC is also a parameter free spectral clustering method and relative new. We will first give a brief overview about STSC and then design a synthetic data experiment to highlight its difference between our method. STSC uses local scale parameters carefully and avoids using an uniform scale parameter, therefore the outlier influence is reduced. On the other hand, SSR focuses on representing every data vector using other vectors directly, therefore we don't worry about how to measure the similarity between vectors using appropriate metrics.

Next, we conduct a synthetic data experiment to demonstrate the difference in similarity graph construction between STSC and our algorithm SSR.

```
Input: Data Matrix $X \in \mathbb{R}^{d \times n}$
Output: $\boldsymbol{\alpha}_1, \ldots, \boldsymbol{\alpha}_n$
1. Compute the local scale $\sigma_i$ for each $\mathbf{x}_i$ using the formula
       $\sigma_i = d(\mathbf{x}_i, \mathbf{x}_K)$,
   where $\mathbf{x}_K$ is $\mathbf{x}_i$'s K-th neighbor
2. Construct the affinity matrix and normalized Laplacian matrix $\mathbf{L}$. 3. Find $v_1, \ldots, v_C$ the $C$ largest eigenvectors of $\mathbf{L}$ and form the matrix $\mathbf{V} = [\mathbf{v}_1, \ldots, \mathbf{v}_C] \in \mathbb{R}^{n \times C}$, where C is the largest possible group number
4. Recover the rotation $\mathbf{R}$ which best aligns $\mathbf{V}$'s columns with the canonical coordinate system using the incremental gradient descent scheme
5. Grade the cost of the alignment for each group number, up to $C$, according to $J = \sum_{i=1}^{n}\sum_{j=1}^{C}\frac{Z_{ij}^2}{M_i^2}$, where $\mathbf{Z} = \mathbf{VR}$
6. Set the final group number $C_{best}$ to be the largest group number with minimal alignment cost.
7. Take the alignment result $\mathbf{Z}$ of the top $C_{best}$ eigenvectors and assign the original point $\mathbf{x}_i$ to cluster c if and only if $\max_j \left(Z_{ij}^2\right) = Z_{ic}^2$
```

Algorithm 2: Self-Tuning Spectral Clustering

We construct 3 groups of normal distributed vectors of dimension 10. First group consists of 10 vectors and with mean value 1, second, third group both contain 10 vectors but with mean 2 and 3 respectively. We try to do the clustering for these 3 groups vectors. To make the problem non-trivial, we also add noise with mean 0 and standard deviation 0.5 to the total 30 vectors. Such experiments are repeated 100 times. We now would like to introduce clustering measure metrics to evaluate clustering performance, which would be used to evaluate synthetic data set mentioned and real data sets in the next part.

4.1 Evaluation Metrics

To evaluate the clustering results, we adopt the three widely used clustering performance measures which are defined below.

Clustering Accuracy discovers the one-to-one relationship between clusters and classes and measures the extent to which each cluster contained data points from the corresponding class. Clustering Accuracy is defined as follows:

$$Acc = \frac{\sum_{i=1}^{n}\delta(map(r_i), l_i)}{n}, \tag{25}$$

where r_i denotes the cluster label of $\mathbf{x}_i$ and l_i denotes the true class label, n is the total number of samples, $\delta(x, y)$ is the delta function that equals one if $x = y$ and equals zero otherwise, and $map(r_i)$ is the permutation mapping function that maps each cluster label r_i to the equivalent label from the data set.

Normalized Mutual Information(NMI) is used for determining the quality of clusters. Given a clustering result, the

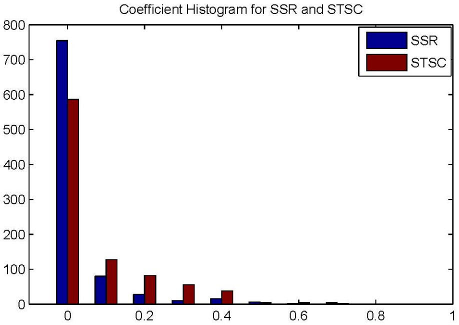

Figure 1: A demonstration for similarity graph coefficients between SSR and STSC. We do the histogram plot for the 900 pair-wise similarity coefficients for both methods. It can be observed that SSR yields a significant more sparse solution than STSC does. Such sparsity could also improve clustering.

NMI is estimated by

$$NMI = \frac{\sum_{i=1}^{c}\sum_{j=1}^{c} n_{i,j}\log\frac{n_{i,j}}{n_i\hat{n}_j}}{\sqrt{(\sum_{i=1}^{c} n_i \log\frac{n_i}{n})(\sum_{j=1}^{c}\hat{n}_j\log\frac{\hat{n}_j}{n})}} \tag{26}$$

where n_i denotes the number of data contained in the cluster $C_i (1 \le i \le c)$, $\hat{n}_j$ is the number of data belonging to the $L_j (1 \le j \le c)$, and $n_{i,j}$ denotes the number of data that are in the intersection between cluster C_i and the class L_j.

Purity measures the extent to which each cluster contained data points from primarily one class. The purity of a clustering is observed by the weighted sum of individual cluster purity values, given as follows:

$$Purity = \sum_{i=1}^{K}\frac{n_i}{n}P(S_i), \quad P(S_i) = \frac{1}{n_i}\max_j(n_i^j) \tag{27}$$

where S_i is a particular cluster size of n_i, n_i^j is the number of the i-th input class that was assigned to the j-th cluster. K is the number of the clusters and n is the total number of the data points.

4.2 Synthetic Data Experiment Results

We first plot the similarity graph coefficients for the two methods in Fig. (1). Since the random vectors vary a lot during different times of experiments, it makes no sense plotting the average coefficient. As a result, we have to demonstrate the plot for a typical experiment, however, the plot does not vary noticeable according to our empirical experiments. It can be observed that our method yields a much sparse similarity graph.

Next, we summarize the average clustering performance of these two methods on this synthetic data set in Table 1. It can be observed that both methods achieve very good clustering results due to the clear structure of data matrix. However, SSR slightly outperforms STSC in terms of all three measure metrics.

Measure	Acc
SSR	0.976 ± 0.013
STSC	0.953 ± 0.014
Measure	NMI
SSR	0.963 ± 0.011
STSC	0.943 ± 0.017
Measure	Purity
SSR	0.976 ± 0.013
STSC	0.953 ± 0.014

Table 1: Clustering Performance on Synthetic Data Set for Two Parameter-Free Methods

5 Experiments on Real Data Sets

In this section, we will evaluate the performance of the proposed method on benchmark data sets. We compare the our method with K-means, NMF [Lee and Seung, 2001], Normalized Cut (NCut) [Shi and Malik, 2000] and Self-Tuning Spectral Clustering (STSC) [Zelnik-Manor and Perona, 2004]. K-means and NMF are classic clustering methods and perfect for benchmark purpose. We also include NC and STSC here since their close connection with our proposed method.

Parameter setting is relatively easy for these methods. STSC and our method are both parameter free. For K-means and NMF, we set the number of clusters equal to the number of classes for each data set. For Ncut, we construct the similarity matrix via tuning the scale parameter σ from the list $\{10^{-2}, 10^{-1}, 1, 10^{1}, 10^{2}\}$. Under such setting of each method, we repeat clustering 30 times and report the average for each method.

There are in total 9 data sets used in our experiment section. Among those, 6 are image ones, AR [1][Martinez and Kak, 2001], AT&T [2][Samaria and Harter, 1994], JAFFE [Lyons *et al.*, 1998], subset of MNIST [LeCun *et al.*,], subset of PIE [3][Sim *et al.*, 2002], subset of UMIST. The other 3 non-image ones are from UCI machine learning repository [Frank and Asuncion, 2010]: Abalone, Ecoli and Scale.

Table 1 summarizes the characteristics of the data sets used in the experiments.

5.1 Experiment Results and Discussions

Table 3 summarizes the results for all methods on the benchmark data sets specified above. We bold the corresponding result if it is statistically significant better than results from other methods via T-test. It can be observed that our proposed method outperforms other methods on all these data sets with the specified metrics. In particular, our method significantly gets a better result than Ncut (standard spectral clustering) and STSC (parameter-free spectral clustering) on some image data sets. AR and PIE, which both contain a large number of

[1] http://www2.ece.ohio-state.edu/ aleix/ARdatabase.html, other downloads 1.

[2] http://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html

[3] http://www.zjucadcg.cn/dengcai/Data/data.html,we use first 10 images in each class

Data set	No. of Observations	Dimensions	Classes
AR	2600	792	100
AT&T	400	168	40
JAFFE	360	90	15
MNIST	150	196	10
PIE	680	256	68
UMIST	360	168	20
Abalone	4177	8	3
Ecoli	327	7	5
Scale	625	4	3

Table 2: Description of Data Set

(a) Accuracy

DataSets	k-means	NMF	NCut	STSC	SSR
AR	0.133	0.143	0.158	0.130	**0.324**
AT&T	0.664	0.678	0.698	0.685	**0.763**
JAFFE	0.789	0.774	0.795	0.813	**0.902**
MNIST	0.641	0.636	0.647	0.693	**0.796**
PIE	0.229	0.241	0.234	0.186	**0.325**
UMIST	0.475	0.457	0.443	0.394	**0.514**
Abalone	0.508	0.519	0.465	0.481	0.513
Ecoli	0.497	0.486	0.481	0.476	0.502
Scale	0.517	0.535	0.536	0.541	0.579

(b) NMI

DataSets	k-means	NMF	NCut	STSC	SSR
AR	0.321	0.317	0.376	0.353	**0.536**
AT&T	0.846	0.848	0.858	0.856	**0.892**
JAFFE	0.848	0.837	0.863	0.872	**0.913**
MNIST	0.665	0.654	0.676	0.681	**0.731**
PIE	0.537	0.528	0.531	0.524	0.553
UMIST	0.667	0.657	0.653	0.598	**0.713**
Abalone	0.115	0.133	0.123	0.118	**0.147**
Ecoli	0.678	0.684	0.687	0.668	0.695
Scale	0.129	0.089	0.107	0.118	0.147

(c) Purity

DataSets	k-means	NMF	NCut	STSC	SSR
AR	0.137	0.142	0.160	0.145	**0.344**
AT&T	0.712	0.724	0.737	0.725	0.750
JAFFE	0.817	0.812	0.811	0.803	**0.913**
MNIST	0.641	0.636	0.667	0.693	**0.796**
PIE	0.259	0.277	0.257	0.231	**0.369**
UMIST	0.545	0.527	0.517	0.487	0.564
Abalone	0.481	0.474	0.468	0.463	0.494
Ecoli	0.546	0.564	0.575	0.568	0.583
Scale	0.667	0.658	0.655	0.632	0.684

Table 3: Clustering Results on Benchmark Data Sets

classes and noisy samples, are widely used benchmark data sets for sparse representation related classification methods. Note that Ncut could get better result if we tune the scale parameter in a refined way. However, as we mentioned in the introduction, choosing an appropriate scale parameter is difficult when the ground truth is unknown. Choosing an uniform scale parameter is also prone to outlier influence. The results demonstrate that our method provides the potential solution to the open issue in spectral clustering.

5.2 Computational Complexity and Scalability Discussion

Our method provides a parameter-free way to construct the similarity graph for subsequent clustering task. Its computational complexity is comparable to conventional spectral clustering methods, indeed significantly faster. For each data vector, the sparse coefficient vector will converge in the quadratic order in each iteration due to integrated Newton acceleration idea. The algorithm usually converges after a limited number of iterations. Our method runs efficiently on individual machines.

It is noting that our method does not have scalability issue on distributed computing system either. In addition to its impressive performance on individual machine, our method can be easily extended to distributed computing platform. With Apache Spark, we can hash all data vectors in the memory, compute the individual coefficient vectors on work nodes, and transmit the $\boldsymbol{\alpha}$s back to name node. This is due to the fact the sparse vector $\boldsymbol{\alpha}$ for each individual vector can be easily paralleled.

6 Conclusion and Future Work

In this paper, we proposed a parameter free spectral clustering method that is robust to data noise and scale inconsistence. Our framework addresses the potential solution to many open issues of spectral clustering. Our simplex representation objective function is derived in a natural way and solved via a novel algorithm. The projected gradient method was accelerated via a combination of auxiliary variable and Newton root finding algorithm. Empirical experiments on both synthetic and real data sets demonstrate the effectiveness of our method.

In the future, we have two further research directions. First, we will try to encode label information in our graph construction, and therefore extend our framework to semi-supervised case. In the literature, there have been extensive work in semi-supervised graph learning [Kulis *et al.*, 2005; Huang *et al.*, 2013b]. In terms of our framework, we may consider using $\ell_{2,1}$ norm in Eq. (7) to take advantage of structural sparsity instead of flat sparsity. Second, we may further look into how to improve the clustering performance after we get the similarity graph, prior work include [Huang *et al.*, 2013a].

References

[Boyd and Vandenberghe, 2004] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[Candès and Tao, 2006] E. Candès and T. Tao. Near optimal signal recovery from random projections: Universal encoding strategies? *IEEE Transactions on Information Theory*, 52(12):5406–5425, 2006.

[Donoho, 2004] D.L. Donoho. For most large underdetermined systems of linear equations the minimal ℓ_1-norm solution is also the sparsest solution. *Communication on Pure and Applied Mathematics*, pages 797–829, 2004.

[Frank and Asuncion, 2010] A. Frank and A. Asuncion. Uci machine learning repository, 2010.

[Hagen and Kahng, 1992] L. Hagen and A. Kahng. New spectral methods for ratio cut partioning and clustering. *IEEE Transactions on Computer-Aided Design*, 11(6):1074–1085, 1992.

[Huang *et al.*, 2013a] J. Huang, F. Nie, and H. Huang. Spectral rotation versus k-means in spectral clustering. In *AAAI Conference on Artificial Intelligence*, pages 431–437, 2013.

[Huang *et al.*, 2013b] J. Huang, F. Nie, and H. Huang. Supervised and projected sparse coding for image classification. In *AAAI Conference on Artificial Intelligence*, pages 438–444, 2013.

[Kleinberg, 1999] J.M. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of ACM*, 46(5):604–632, 1999.

[Kulis *et al.*, 2005] B. Kulis, B. Basu, I. Dhillon, and R. Mooney. Semi-supervised graph clustering: a kernel approach. In *International Confernence on Machine Learning*, pages 457–464, 2005.

[LeCun *et al.*,] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.

[Lee and Seung, 2001] D.D. Lee and H.S. Seung. Algorithms for non-negative matrix factorization. In *Neural Information Processing Systems Conference*, pages 556–562, 2001.

[Li *et al.*, 2007] Z. Li, J. Liu, S. Chen, and X. Tang. Noise robust spectral clustering. In *Proceedings of the International Conference of Computer Vision*, pages 1–8, 2007.

[Long *et al.*, 2006] B. Long, Z. Zhang, X. Wu, and P. Yu. Spectral clustering for multi-type relational data. In *Proceedings of the International Conference on Machine Learning*, pages 585–592, 2006.

[Lyons *et al.*, 1998] M.J. Lyons, S. Akamatsu, M. Kamachi, and J. Gyoba. Coding facial expressions with gabor wavelets. In *3rd IEEE International Conference on Automatic Face and Gesture Recognition*, 1998.

[Martinez and Kak, 2001] A.M. Martinez and A.C. Kak. Pca versus lda. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2(23):228–233, 2001.

[Nesterov, 1983] Y. Nesterov. Method for solving a convex programming problem with convergence rate $o(1/k^2)$. *Soviet Math Dokl*, 1983(2):372–376, 1983.

[Nesterov, 2003] Y. Nesterov. *Introductary lecture notes on convex optimization: a basic course*. Kluwer Academic Publishers, 2003.

[Nesterov, 2005] Y. Nesterov. Smooth minimization of nonsmooth functions. *Mathematical Programming: Series A and B*, 103(1):127–152, 2005.

[Nesterov, 2007] Y. Nesterov. Gradient methods for minimizing composite objective function. 2007.

[Ng *et al.*, 2002] A. Ng, M. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In *Advances in Neural Information Processing Systems*, pages 849–856, 2002.

[Nie *et al.*, 2014] Feiping Nie, Xiaoqian Wang, and Heng Huang. Clustering and projected clustering with adaptive neighbor assignment. In *The 20th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 977–986, 2014.

[Page *et al.*, 1999] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web, 1999.

[Roweis and Saul, 2000] S. T Roweis and L. K Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000.

[Samaria and Harter, 1994] F. Samaria and A. Harter. Parameterisation of a stochastic model for human face identification. In *2nd IEEE Workshop on Applications of Computer Vision*, 1994.

[Shi and Malik, 2000] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

[Sim *et al.*, 2002] T. Sim, S. Baker, and M. Bsat. The cmu pose, illumination, and expression database. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(12):1615–1618, 2002.

[Tenenbaum *et al.*, 2000] J.B. Tenenbaum, V.de. Silva, and J.C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.

[Wang *et al.*, 2014] De Wang, Feiping Nie, and Heng Huang. Large-scale adaptive semi-supervised learning via unified inductive and transductive model. In *The 20th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 482–491, 2014.

[Wright *et al.*, 2008] J. Wright, A.Y. Yang, A. Ganesh, S.S. Sastry, and Y. Ma. Robust face recognition via sparse representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(2):210–217, 2008.

[Zelnik-Manor and Perona, 2004] L. Zelnik-Manor and P. Perona. Self-tuning spectral clustering. In *Advances in Neural Information Processing Systems*, pages 1601–1608, 2004.

[Zhu *et al.*, 2003] X. Zhu, Z. Ghahramani, and J.D. Lafferty. Semi-supervised learning using gaussian fields and harmonic functions. In *Proceedings of the 20th International Conference on Machine Learning*, pages 912–919, 2003.