Grassmannian Manifold Optimization Assisted Sparse Spectral Clustering

Qiong Wang[a] Junbin Gao[b] Hong Li[a]
[a]School of Mathematics and Statistics
Huazhong University of Science and Technology,Wuhan 430074, China
{wangqiong701,hongli}@hust.edu.cn
[b]Discipline of Business Analytics, The University of Sydney Business School
The University of Sydney, NSW 2006, Australia
junbin.gao@sydney.edu.au

Abstract

Spectral Clustering is one of pioneered clustering methods. It relies on the spectral decomposition criterion to learn a low-dimensional embedding of data for a basic clustering algorithm. The sparse spectral clustering (SSC) introduces the sparsity for the similarity in low-dimensional space by enforcing a sparsity-induced penalty, resulting a non-convex optimization, which is solved by a relaxed convex problem via the standard ADMM (Alternative Direction Method of Multipliers), rather than inferring latent representation from eigen-structure. This paper provides a direct solution as solving a new Grassmann optimization problem. By this way calculating latent embedding becomes part of optmization on manifolds and the recently developed manifold optimization methods can be applied. It turns out the learned new features are not only very informative for clustering, but also more intuitive and effective in visualization after dimensionality reduction. We conduct empirical studies on simulated datasets and several real-world benchmark datasets to validate the proposed methods. Experimental results exhibit the effectiveness of this new manifold-based clustering and dimensionality reduction method.

1. Introduction

Clustering is one of the main unsupervised learning tasks in exploratory data analysis, with applications ranging from statistics, computer science, biology to social sciences or psychology [13]. The focus of data clustering has moved from the classic centroid-oriented clustering [4] to the subspace clustering [13, 24] in which data are clustered according to their affinity towards subspace structures. With different motivation, researchers have developed many types of clustering algorithms. Spectral Clustering (SC) [10] is one of pioneered clustering methods in machine learning community. We have seen its applications for motion segmentation [13], image segmentation [22], speeach separation [3], scientific journal clustering and power load clustering [17]

Generally speaking, two major steps in the spectral clustering algorithm are, referring to Section 2 or [10, 21] etc., (1) forming/learning a similarity/affinity matrix for the given data sample set; and (2) performing general clustering methods to categorize data samples such as Normalized Cuts (NCut) [22]. These two major steps determine the performance of spectral clustering methods. A large body of work has been investigated to improve the performance of SC method via learning a better affinity matrix and inferring new better representation of data. This paper is related to the second paradigm aiming at learning latent representation for original data. As both paradigms are highly related to each other, it is worthwhile reviewing the recent research of the first paradigm.

Two classical representatives of learning similarity/affinity matrix for spectral clustering-based methods are Sparse Subspace Clustering (SSubC) [13] and Low-Rank Representation (LRR) [18]. Both SSubC and LRR rely on the self-expressive property in linear space [13]: *each data point in a union of subspace can be efficiently reconstructed by a linear combination of other points in the data set.* SSubC further induces sparsity by utilizing the l_1 Subspace Detection Property in an independent manner, while the LRR model considers the intrinsic relation among the data objects in a holistic way via the low-rank requirement. It has been proved that, when the data set is actually composed of a union of multiple subspaces, the LRR method can reveal this structure through subspace clustering [19, 8]. The self-expressive property builds on the linear relations among data. To exploit nonlinear information hidden in manifold structure of data particularly manifold-valued data, several authors have explored self-expressive in terms of manifold geometry and extended LRR for Stiefel manifolds [28], positive definite manifolds [14] and Grassmann manifolds [25, 26]. In the second paradigm, many researchers take it as an object to learn informative latent representation. In the recent Sparse Spectral Clustering (SSC), Lu *et al.* [20] specify a sparsity-induced penalty to learn more clusters favored latent representation. Due to the non-Frobenius norm constrain, the solution is no longer determined by eigenvectors and the latent representation is calculated through a post stage.

Indeed, the objective of SC is to characterize how close the eigen-structure of a similarity/affinity matrix is to a partition implied by the latent representation [3]. In other words, instead of explicitly inferring latent representation for eigen-structure, learning the subspace structure should be desired. This prompts that it is necessary and more reasonable to implement the SC optimization over subspaces, i.e., the points on Grassmann manifolds.

In recent years, Grassmann manifold [5, 9] has attracted great interest in the computer vision research community for numerous application tasks. Mathematically the Grassmann manifold $\mathcal{G}(d, N)$ is defined as the set of all d-dimensional subspaces in $\mathbb{R}^N$, where $0 \leq d \leq N$. Its Riemannian geometry has recently been well investigated in literature, e.g., [2, 1, 12]. This has paved the way for solving any optimization problems defined on a Grassman manifold. In fact, in recent years, the machine learning community has been paying attention to manifold optimization for machine learning tasks. Theis *et al.* [23] formulate independent component analysis with dimensionality reduction as optimization over the Stiefel manifold. Journee *et al.* [16] frame sparse principal component analysis over this manifold as well. And Cunningham and Ghahramani[11] propose a uniform framework for dimensionality reduction. Section 2 will make it clear how to embed the Grassmann manifold optimization in the SC framework.

Compared to the aforementioned LRR subspace clustering approaches, SSubC has its own advantages, e.g., its robustness and lower computation complexity. This paper is related to learning a better and efficient latent feature representation with Grassmann manifold optimization. It turns out the learned new features are not only very informative for clustering, but also more intuitive and effective in visualization after dimensionality reduction. The primary contributions of this paper are:

1. We take a straightforward way to optimize the sparse spectral clustering objective introduced in [20] by adopting Grassmann manifold optimization strategy;
2. We integrate the standard ADMM (Alternative Direction Method of Multipliers) [6] with Grassmann manifold optimization, inspired by recent works [7].
3. We explore the application of the new SC algorithm for dimensionality reduction based on the solution given by Grassmann manifold optimization algorithm.

The paper is organized as follows. In Section 2, we summarize the related works and the preliminaries for SC and SSC. Section 3 focuses on the framework of Grassmann manifold optimization and the necessary optimization related ingredients will be developed. Finally the clustering algorithm for solving the proposed model on the Grassmann manifold is proposed. In Section 4, the performance of the proposed method is evaluated via both the clustering and dimensionality reduction problems on both data and real-world datasets. Finally, conclusions and suggestions for future work are provided in Section 5.

2. The Problem Revisited

Before going on, we start with a brief introduction of spectral clustering. Suppose

$$\mathbf{X} = [\mathbf{x}_1, \cdots, \mathbf{x}_N] \in \mathbb{R}^{D \times N}$$

is a set of N data points to be clustered where D is the dimension of data. We denote the number of clusters by K which is a pre-specified integer, although K can be learned from the given dataset.

The purpose of clustering is to partition the dataset $\mathbf{X}$ into K clusters according to certain similarity criteria. The Spectral Clustering (SC) has been a widely used clustering technique [21]. To serve this paper, we repeat the generic SC algorithm as follows [20],

1. Construct a $N \times N$ matrix $\mathbf{W}$ of pairwise similarities (weights) among these N points.
2. Form the normalized graph Laplacian

   $$\mathbf{L} = \mathbf{I} - \mathbf{D}^{-\frac{1}{2}}\mathbf{W}\mathbf{D}^{-\frac{1}{2}}$$

   where $\mathbf{D}$ is the diagonal matrix with $d_{ii} = \sum_{j=1}^N w_{ij}$.
3. Solve the following constrained optimization for computing $\mathbf{U} \in \mathbb{R}^{N \times d}$,

   $$\min_{\mathbf{U} \in \mathbb{R}^{N \times d}} \langle \mathbf{U}\mathbf{U}^T, \mathbf{L} \rangle \qquad \text{s.t.} \quad \mathbf{U}^T\mathbf{U} = \mathbf{I}. \quad (1)$$

4. Normalize each row of $\mathbf{U} \in \mathbb{R}^{N \times d}$ to get a new matrix $\widehat{\mathbf{U}}$.
5. Conduct the k-means on the rows of $\widehat{\mathbf{U}}$ to cluster them into K groups.

The rows of $\mathbf{U}$ are actually regarded as the low-dimensional representation of the original D-dimension data $\mathbf{X}$. The elements of $\mathbf{U}\mathbf{U}^T$ represents the similarity or affinity between the latent representation (rows) of the original data, thus the objective function in problem (1) is to enforce the dissimilarity between data similarity and the graph Laplacian $\mathbf{L}$ from data. The columns of $\mathbf{U}$ indeed come from the first d eigenvectors of $\mathbf{L}$ corresponding to the smallest d eigenvalues, or equivalently the eigenvectors of the normalized affinity matrix $\mathbf{D}^{-\frac{1}{2}}\mathbf{W}\mathbf{D}^{-\frac{1}{2}}$ corresponding to the d largest eigenvalues. It has been observed by Ng *et al.* [21] that when some key eigenvalues are equal, the best new representation $\mathbf{U}$ is determined by the subspace rather than by the particular eigenvectors. Ng *et al.* also analyzed the reasons why an embedding in the low-dimensional is more effective in clustering.

As pointed out by Ng *et al.* [21], in the ideal scenarios, $\mathbf{U}\mathbf{U}^T$ can be permuted to block diagonal structure. A block diagonal structure is favored, as it improves clustering performance. The idea of inducing or enforcing sparsity in the Spectral Clustering has been exploited in a recent work [20] where the Sparse Spectral Clustering (SSC) has been proposed.

The SSC seeks for a better representation $\mathbf{U}$ by solving the following sparsity-induced problem

$$\min_{\mathbf{U}\in\mathbb{R}^{N\times d}} \langle \mathbf{U}\mathbf{U}^T, \mathbf{L}\rangle + \beta\|\mathbf{U}\mathbf{U}^T\|_0, \text{ s.t. } \mathbf{U}^T\mathbf{U} = \mathbf{I}, \quad (2)$$

where $\beta > 0$ trades off the objective of SC and the sparsity of $\mathbf{U}\mathbf{U}^T$. However, minimization of $\|\cdot\|_0$ results in a NP-hard combinatorial optimization problem. It is well-known that the ℓ_1-norm is the convex envelope of ℓ_0-norm within the ℓ_1-ball. It is a general practice in machine learning to replace ℓ_0-norm with ℓ_1-norm, thus problem (2) can be relaxed to the following problem for better numerical purpose,

$$\min_{\mathbf{U}\in\mathbb{R}^{N\times d}} \langle \mathbf{U}\mathbf{U}^T, \mathbf{L}\rangle + \beta\|\mathbf{U}\mathbf{U}^T\|_1, \text{ s.t. } \mathbf{U}^T\mathbf{U} = \mathbf{I}. \quad (3)$$

Ideally, the elements in $\mathbf{U}\mathbf{U}^T$ corresponding to the weak inter-cluster connections tends to be zeros, while the ones corresponding to the strong intra-cluster connections will be kept. However the relaxed problem (3) is still a constrained nonconvex optimization in terms of the latent variable $\mathbf{U}$. The authors of [20] go further to optimize the new variable $\mathbf{P} = \mathbf{U}\mathbf{U}^T$ instead, based on the fact that the set $S_2 = \{\mathbf{P} \in \mathbb{S}^{N\times N} \mid 0 \preceq \mathbf{P} \preceq \mathbf{I}, \text{tr}(\mathbf{P}) = d\}$ is the convex hull of $S_1 = \{\mathbf{U}\mathbf{U}^T \mid \mathbf{U} \in \mathbb{R}^{N\times d}, \mathbf{U}^T\mathbf{U} = \mathbf{I}\}$.

Finally the SSC aims at solving the following relaxed convex problem defined as follows:

$$\begin{aligned} &\min_{\mathbf{P}\in\mathbb{S}^{N\times N}} \langle \mathbf{P}, \mathbf{L}\rangle + \beta\|\mathbf{P}\|_1, \\ &\text{s.t. } 0 \preceq \mathbf{P} \preceq \mathbf{I}, \ \text{tr}(\mathbf{P}) = d. \end{aligned} \quad (4)$$

Now problem (4) can be solved by the standard ADMM procedure with the assistance of an efficient algorithm for the so-called capped simplex projection problem. For more details refer to [20, 27]. After solving (4) for $\mathbf{P}$, the final solution for the latent representation $\mathbf{U}$ to (3) can be approximated by using the first d eigenvectors corresponding to the largest d eigenvalues of $\mathbf{P}$ by its eigen decomposition. As we mentioned earlier, this $\mathbf{U}$ may not be the best solution for the following clustering process. In the next section, we propose to solve the relaxed problem (3) directly.

3. Grassmann Manifold Optimization

3.1. Reforming the Problem

Consider problem (3). Denote the objective function by

$$f(\mathbf{U}) = \langle \mathbf{U}\mathbf{U}^T, \mathbf{L}\rangle + \beta\|\mathbf{U}\mathbf{U}^T\|_1 \quad (5)$$

where $\mathbf{L}$ is the given Laplacian of data graph. The constraint condition in problem (3) defines the so-called Stiefel manifold , consisting of all the orthogonal column matrices,

$$\mathcal{S}(d, N) = \{\mathbf{U} \in \mathbb{R}^{N\times d} \mid \mathbf{U}^T\mathbf{U} = \mathbf{I}\}.$$

Hence problem (3) is an unconstrained manifold optimization problem on the Stiefel manifold $\mathcal{S}(d, N)$.

Consider the d-order group $\mathcal{O}(d) = \{\mathbf{Q} \in \mathbb{R}^{d\times d} \mid \mathbf{Q}^T\mathbf{Q} = \mathbf{I}\}$ of all the $d \times d$ orthogonal matrices. With $\mathcal{O}(d)$, we can define an equivalent relation $\sim$ on the Stiefel manifold $\mathcal{S}(d, N)$ in the sense that $\mathbf{U}_1 \sim \mathbf{U}_2$ means that there exists a $\mathbf{Q} \in \mathcal{O}(d)$ such that $\mathbf{U}_1 = \mathbf{U}_2\mathbf{Q}$. The quotient space of Stiefel manifold $\mathcal{S}(N, d)$ under this equivalent relation is actually the concrete representation of the abstract Grassmann manifold $\mathcal{G}(N, d)$, see [2, 12]. A point on Grassmann manifold is actually realized by the equivalent class $[\mathbf{U}] = \{\mathbf{U}\mathbf{Q} \mid \text{for all } \mathbf{Q} \in \mathcal{O}(d)\}$ where $\mathbf{U} \in \mathbb{R}^{N\times d}$ is an orthogonal column matrix, called a representative of Grassmann point $[\mathbf{U}]$.

It is easy to check that for any $\mathbf{Q} \in \mathcal{O}(d)$ we have

$$f(\mathbf{U}\mathbf{Q}) = f(\mathbf{U}),$$

which shows the objective function of problem (3) is equal valued on the equivalent class $[\mathbf{U}]$. In other words, if $\mathbf{U}$ is a solution to (3), so is $\mathbf{U}\mathbf{Q}$ for any $\mathbf{Q} \in \mathcal{O}(d)$.

Simply optimizing (3) on Stiefel manifold $\mathcal{S}(d, N)$ may result in identifiability issue due to the equal function value on the equivalent class. A better strategy is to re-form the problem on the Grassmann manifold as follows

$$\min_{[\mathbf{U}]\in\mathcal{G}(d,N)} f(\mathbf{U}) = \langle \mathbf{U}\mathbf{U}^T, \mathbf{L}\rangle + \beta\|\mathbf{U}\mathbf{U}^T\|_1, \quad (6)$$

where Equation 6 is an unconstrained Grassmann manifold optimization problem.

A number of methods have been developed to solve manifold optimization problems in the last two decades. The representative work [2] has developed a framework of optimizing functions defined on different types of matrix manifolds. The typical algorithms are the Riemannian gradient descent algorithm and Riemannian trust region algorithm, which are implemented in the Matlab toolbox ManOpt[1]. A successful algorithm for manifold optimization depends on the knowledge of rich Riemann geometry of a concerned manifold.

Edelman *et al.* [12] explored the Riemann geometry for both Stiefel and Grassmann manifolds and developed optimization algorithms on these manifolds. The most important ingredient for these manifold optimization algorithm is the Riemannian gradient induced by the Riemannian metric of a relevant manifold.

The rich geometry of Riemannian manifolds makes it possible to define Riemannian gradients and Hessians of objective functions f, as well as systematic procedures (called retractions) to move on the manifold starting at a point $\mathbf{U}$, along a specified tangent direction at $\mathbf{U}$. For most manifolds embedded in Euclidean space, like Grassmann manifolds, the Riemannian gradient is the projection of Euclidean gradient onto the relevant tangent space of the manifold. In the next subsection, we will focus on computing Euclidean gradient of the sparse spectral clustering objective function (5).

3.2. Computing the Gradient

The objective function (5) of the new optimization problem (6) is not differentiable at the location where elements of $\mathbf{U}\mathbf{U}^T$ are zero. However in this case we will adopt using the subdifferential.

To work out the formula for its Euclidean derivative, we introduce several notations. For a matrix $\mathbf{A}$ of size $m \times n$, vec$(\mathbf{A})$ is a mn-dimensional vector by stacking columns of $\mathbf{A}$ one by one, and ivec(vec$(\mathbf{A})) = \mathbf{A}$ the inverse operation of vec. $A \otimes \mathbf{B}$ is the Kronecker product of matrices $\mathbf{A}$ and $\mathbf{B}$. The transform $T_{m,n}$ is a matrix of size $mn \times mn$ such that vec$(\mathbf{A}) = T_{m,n}$vec$(\mathbf{A}^T)$.

For the first term in the objective function (5), we note that:

$$\langle \mathbf{U}\mathbf{U}^T, \mathbf{L} \rangle = \mathrm{tr}(\mathbf{U}\mathbf{U}^T\mathbf{L}) = \mathrm{tr}(\mathbf{U}^T\mathbf{L}\mathbf{U}).$$

Hence

$$\nabla \langle \mathbf{U}\mathbf{U}^T, \mathbf{L} \rangle = \mathbf{L}\mathbf{U} + \mathbf{L}^T\mathbf{U} = 2\mathbf{L}\mathbf{U}$$

because $\mathbf{L}$ is normally symmetric.

Consider the second term of the objective function. First, according to the chain rule, we have:

$$\mathrm{vec}\left(\frac{\|\mathbf{U}\mathbf{U}^T\|_1}{\partial \mathbf{U}}\right)^T = \mathrm{vec}(\mathrm{sgn}(\mathbf{U}\mathbf{U}^T))^T \frac{\partial \mathbf{U}\mathbf{U}^T}{\partial \mathbf{U}},$$

where

$$\frac{\partial \mathbf{U}\mathbf{U}^T}{\partial \mathbf{U}} = (\mathbf{I}_{N^2} + T_{N\times N, N\times N})(\mathbf{U} \otimes \mathbf{I}_N).$$

Define the column vector D as

$$D = \frac{\partial \mathbf{U}\mathbf{U}^T}{\partial \mathbf{U}} \mathrm{vec}(\mathrm{sgn}(\mathbf{U}\mathbf{U}^T))$$

Thus the Euclidean derivative of the objective function $f(\mathbf{U})$ is:

$$\nabla f(\mathbf{U}) = 2\mathbf{L}\mathbf{U} + \beta \mathrm{ivec}(D). \tag{7}$$

3.3. The Sparse Spectral Clustering Algorithm

At the representative $\mathbf{U}$ of a Grassmann point $[\mathbf{U}]$, the Riemann gradient can be simply calculated as

$$\mathrm{grad}_{[\mathbf{U}]} f = (\mathbf{I} - \mathbf{U}\mathbf{U}^T)\nabla f(\mathbf{U}).$$

With this calculation, we can employ the ManOpt toolbox to solve Grassmann manifold optimization problem (6) to get a solution $\mathbf{U}$. For the initial $N \times k$ latent representation matrix $\mathbf{U}^{(0)}$,it can be approximated by using the first k eigenvectors corresponding to the largest k eigenvalues of $\mathbf{W}$, where the $N \times N$ matrix $\mathbf{W}$ is the pairwise similarities (weights) among the N data points. We summarize the algorithm in Algorithm 1.

Algorithm 1 Grassmann Manifold Optimization Assisted Spectral Clustering (GSC) Algorithm

```
Input: The data matrix $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N]$, the number
    of latent dimension $d$ and the trade-off parameter $\beta$.
Output: The sparse latent representation $\mathbf{U}$.
  Form the affinity matrix $\mathbf{W}$, and compute the initial latent representation $\mathbf{U}^{(0)}$;
  Compute the normalized Laplacian matrix $\mathbf{L}$;
  With the initial $\mathbf{U}^{(0)}$, call the Riemannian trust-region (RTR) algorithm in ManOpt toolbox to optimize the objective, until a pre-defined termination criterion is satisfied.
```

There are two ways to use the solution $\mathbf{U}$ given by the Riemannian trust-region (RTR) algorithm. As the rows of $\mathbf{U}$ are regarded as latent representation of original data. When d is relevantly small, we can visualize them as the results from dimensionality reduction. Hence the optimization of problem (6) can be regarded as a dimensionality reduction while the criterion is equivalent to matching the similarity information of objects in both data space and latent space [21]. This has certain link with the idea of Twin Kernel Embedding [15] where a kernel relationship between data space and latent space was specified via kernel functions. Our experiments on both synthetic and real-world data show excellent performance in terms of dimensionality reduction and visualization. We call this the Grassmann Manifold Optimization Assisted Dimensionality Reduction (GDR).

The second way is to use $\mathbf{U}$ for final data clustering. For example, we can take as input $\mathbf{U}$ for the k-means. In this

[1] http://www.manopt.org

paper, we construct a new affinity matrix $\mathbf{W}^* = \mathbf{U}\mathbf{U}^T$ and take it as input for the Normalized Cut (NCut) method to separate the data into clusters. We call the algorithm Grassmann Manifold Optimization Assisted Spectral Clustering (GSC) algorithm.

4. Experiment Results

In this section, we first evaluate the performance of our GSC model on synthetic data sets for clustering. Secondly, we combine it with GDR model for dimensionality reduction and clustering. We also make a comparison between our algorithm and traditional Ncut algorithm and the convex Sparse Spectral Clustering (SSC) proposed in [20]. As the performance of SSC with Normalized Cut (NCut) is superior than with K-means. For a fair comparison, we changed the final SSC processing step in [20]. In our program,instead of approximating $\mathbf{U}$ by using the first k eigenvectors corresponding to k largest eigenvalues of $\mathbf{P}$, we directly take matrix $\mathbf{P}$ as input for the Ncut method to separate the data into clusters. The algorithm is coded in Matlab R2015a and conducted on a PC with a CPU 3.20GHz and 8G RAMs.

4.1. Experiments on Synthetic Data

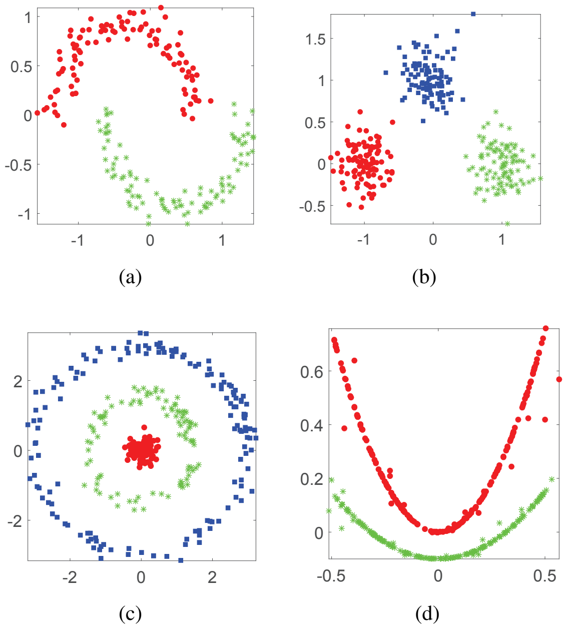
The synthetic data used in this experiment are (1) two moons data; (2) three Gaussian data; (3) three rings data; and (4) two disjoint quadratic para-curves. The data used in this experiment are shown in Figure 1 with their clusters colored.

Two-Moon data: There exist two clusters of data distributed in the half-moon shape with slightly crossing overlapping. This dataset is frequently used to test the performance of new clustering algorithms. The data was randomly generated from two sine-shape curves with the noise percentage set to 0.09. In this case, each cluster contains 100 samples.

Three-Gaussian data: The second toy data set is sampled from a mixture of three Gaussian components. Each cluster obeys a Gaussian distribution with a variance of 0.05. Each cluster has 100 samples.

Three-Ring data: The third synthetic data set is a randomly generated three-ring data in 2-dimensional plane, among which the data are distributed on circles, with the noise percentage set to 0.15. There are 100,100 and 150 samples in each cluster, respectively.

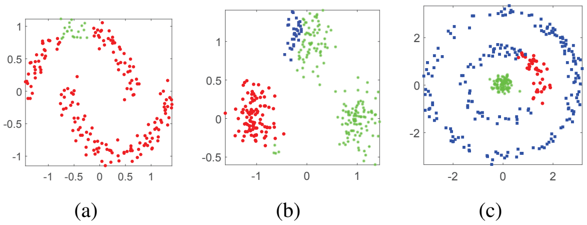
Two Disjoint Para-Curves data: The last data set consists of two clusters of data distributed on two disjoint parabolic-shape curves without overlapping, corrupted with Gaussian noise of 0 mean and variance 0.05. Each cluster contains 200 samples.



(a) (b)

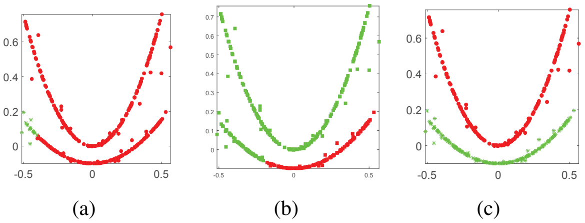
(c) (d)

Figure 1. Synthetic toy data sets: Two-moon data (a); Three-gaussian data (b); Three-ring data (c); and two disjoint para-curves (d).



(a) (b) (c)

Figure 2. Visualization of the Ncut data clustering results on two moon (a), three-gaussian (b) and three-ring data (c). Both GSC and SSC achieve 100% accuracy for these three cases.



(a) (b) (c)

Figure 3. Two disjoint para-curves clustering results, using Ncut (a), SSC (b) and GSC (c)

In our algorithm, we use the classical k-nearest-neighbor (k-nn for short) graph to construct affinity based on the neighborhood relationship between data points. Then the Gaussian kernel is computed as the affinity similarity, which can also be used as Step 1 of Algorithm 1. The standard deviation of the kernel is taken equal to the median of the pair-wise Euclidean distances between the data points. We

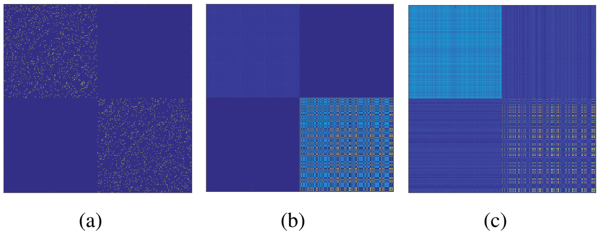


Figure 4. Plots of the affinity matrix of two disjoint para-curves data set: (a) k-nn with $k = 8$; (b) P by SSC in Eqution (4); and (c) $\mathbf{U}\mathbf{U}^T$ by GSC in Equation (6)

set $k = 6$ in two moon, three Gaussian data sets and two disjoint para-curves, while $k = 8$ for three-ring data set. In this work, we assume that the number of clusters K is known.

The clustering accuracy of the classical Ncut algorithm on two-moon, three-Gaussian, three-ring data sets is 56.50%, 60.10%, and 86.00%, respectively. The visualization comparison of the clustering results of the first three data sets based on Ncut are depicted in Figure 2. The results are not satisfactory.

β	Three-Gaussian		Three-Ring	
	SSC	GSC	SSC	GSC
β=0.00001	**100**	**100**	**100**	**100**
β=0.00005	**100**	**100**	81.71	**100**
β=0.0001	**100**	91.31	76.86	**100**
β=0.0005	53	**95.7**	49.43	**87.14**
β=0.001	51.34	**95.3**	41.71	**83.71**
β=0.005	45	**93**	38.00	**83.71**
β=0.01	44.33	**93**	34.57	**83.71**

Table 1. Clustering accuracy(%) of SSC and GSC against different β on three-Gaussian and three-ring data sets.

Instead of applying Ncuts directly on the affinity matrix of the data, we input the affinity matrix as the result of step 1 of Algorithm 1. After getting the solution $\mathbf{U}$ by Algorithm 1, we take as input to the NCut algorithm $\mathbf{W}^* = \mathbf{U}\mathbf{U}^T$. The best results of both GSC and SSC are achieved when $\beta = 0.00001$ for two-moon, three-Gaussian and three-ring data sets. Both achieve 100% accuracy. We compare the influence of β on the performance of both GSC and SSC. As β increases, GSC performs better than SSC, especially for three-ring and three-Gaussian data sets. Table 1 shows the clustering performance versus parameter β on the two data sets. Both cases indicate that GSC performance is much more robust than SSC against β.

For the two disjoint para-curves dataset, the clustering accuracy of Ncut method is poorly 54.50%. The best result for GSC method is 100% when β=0.000001, while the best SSC performance is only 83.50% accuracy. The visualization comparison results on clustering accuracy are shown in Figure 3.

In order to explore the underlying low-dimensional structure within data, we also provide a visual comparison of affinity matrices of two disjoint para-curves data set. Figure 4(a) plots the affinity matrix W by k-nn with $k = 8$; Figure 4(b) shows P by SSC in Equation (4); and Figure 4(c) demonstrates $\mathbf{U}\mathbf{U}^T$ by GSC in Equation (6). The affinity matrix obtained by GSC effectively reveals the cluster structure of data so as to benefit the subsequent clustering task.

From the listed results, three observations can be made:

1. For ordinary data sets such as two-moon, three-Gaussian adnd three-ring data sets, both GSC and SSC can achieve the same best accuracy. The clustering results from GSC and SSC are better than that from the Ncut algorithm.

2. The clustering results for GSC is much more robust than SSC versus the sparse regularization parameter β, especially for the complicated three-Gaussian, three-ring data, and the two disjoint para-curves data sets.

3. For the data such as the two disjoint para- curves, GSC outperforms SSC. The reason may be that GSC takes care of nonlinear manifold structures of data more seriously. This motivates us to conduct our GSC algorithm on many high-dimensional data, and pursuit their latent low-dimensional representation with our GDR method.

4.2. Experiments on Real-World Benchmark Datasets

In this subsection, we conduct several experiments on some public databases to assess the proposed GSC model. We evaluate the proposed clustering methods on extended Yale B data set, ORL data set and the MNIST handwritten digits database. We use the accuracy for performance assessment.

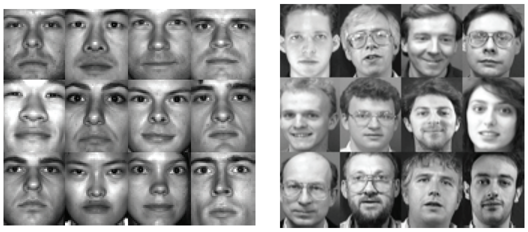
4.2.1 Experiments on Clustering

For the clustering problem, all of the experiments are conducted on the following three public available datasets:

1. The YaleB face database (`http://vision.ucsd.edu/content/yale-face-database`).

2. The ORL face database (`http://www.cam-orl.co.uk`).

3. A subset of handwritten digits images from the MNIST database (`http://yann.lecun.com/exdb/mnist`).



(a) (b)

Figure 5. Examples of the face datasets: (a) Extended Yale B and (b) ORL faces.

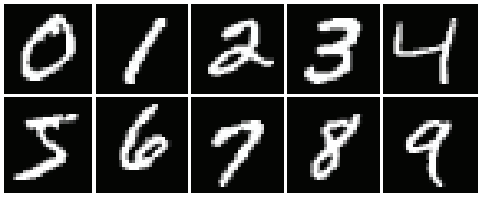


Figure 6. Examples of the MNIST handwritten digits

The Extended Yale B dataset consists of face images of 38 human subjects. Some sample face images are shown in Figure 5(a). For each subject, there are $N_i = 64$ frontal face images acquired under various lighting conditions. The images were captured under different illumination and expression conditions. To reduce the computational cost and the memory requirements of all algorithms, we downsample the images to 32×32 pixels and treat each 1032D vectorized image as a data point.

The ORL face database is composed of 400 images of size 112 x 92 and some samples are shown in Figure 5(b). There are 40 individuals, 10 images per each person. The images were taken at different times, lighting and facial expressions. The faces are in an upright position in frontal view, with a slight left-right rotation. All the data is collected in a matrix of shape 10304 (pixels) x 400 (faces). To avoid large values, the data matrix is divided by 100.

We follow the settings in [13] to construct the affinity matrix by l_1-graph and apply Ncut, SSC and GSC on the constructed affinity matrix for these two face data sets. Six subsets are constructed which consist of all the images of the randomly selected subjects with the number of clusters, i.e., K ranging from 5 to 18. We set $\{0.00001, 0.00005, 0.0001, 0.0005, 0.001, 0.05, 0.01\}$ as the candidate set of parameter β both in GSC (6) and SSC (3),and conducted grid search to tune the parameter β. For each dataset and each algorithm, exhaustive tests were done to find the quasi-optimal parameters setting.The final performance scores were computed by averaging the scores from 20 trials.

Method	Ncut	SSC	GSC
$K = 5$	61.56(9.34)	95.64(5.90)	**96.58(3.94)**
$K = 8$	56.77(8.84)	88.95(4.76)	**91.65(4.64)**
$K = 10$	48.39(7.61)	82.86(4.86)	**85.24(4.34)**
$K = 12$	46.94(4.82)	80.17(4.40)	**82.64(4.20)**
$K = 15$	45.51(4.06)	76.91(1.57)	**77.82(1.63)**
$K = 18$	45.33(3.80)	75.06(1.59)	**76.84(1.49)**

Table 2. Clustering results in terms of accuracy (%) and standard deviation on YaleB dataset.

We also evaluate the stability of the GSC model versus SSC with respect to the sparse regularization parameter β. This time, we set the same β in these two methods, and the number of clusters K is set 5, 10 and 15, respectively.

β	$K = 5$		$K = 10$		$K = 15$	
	SSC	GSC	SSC	GSC	SSC	GSC
β=0.00001	95.53	**96.14**	82.24	**83.30**	76.82	**77.14**
β=0.00005	95.35	**96.25**	81.93	**83.39**	76.67	**77.04**
β=0.0001	95.25	**96.34**	76.25	**81.96**	76.37	**76.59**
β=0.0005	80.38	**95.89**	69.27	**81.61**	67.44	**76.59**
β=0.001	57.98	**85.94**	54.37	**69.85**	68.75	**73.70**
β=0.005	36.98	**72.85**	32.75	**61.35**	52.48	**70.13**
β=0.01	30.59	**62.51**	22.37	**55.00**	38.14	**69.87**

Table 3. Clustering accuracy(%) of SSC and GSC against different β on Yale B face data sets.

The experiments are repeated 20 times and the mean and standard deviation of the clustering accuracy rates with quasi-optimal parameters setting are summarized in Tables 2 and 4 for two datasets. Tables 3 and 5 show the clustering performance against different β on the two face datasets by averaging the clustering accuracy rates of the above repeated 20 times experiments.These results show GSC method outperforms SSC method not only in accuracy but also in stability.

The subset of handwritten digits images in Figure 6 is selected from MNIST database, which contains 60000 digital images with 600 images of each digit. All images are in grayscale and have a uniform size of 28 × 28 pixels. We use a subset which has 400 samples with 5 clusters (each has 80 samples). In this test, we compare the affinity matrices from the original l_1-graph, SSC and GSC algorithms. Clearly our GSC algorithm captures the structure information of clusters.

Now we discuss the convergence of the algorithm.

Convergence and Optimality: The new optimization GSC in (6) is nonlinear and we use the Riemannian trust-region (RTR) algorithm to solve. There could be many local minima. The general convergence analysis has been done in e.g. [1]. For our case, the conditions for the convergence are satisfied, the special initialization from data similarity W may drive the iterative process towards a good solution.

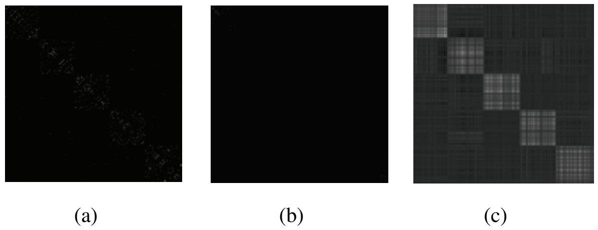


Figure 7. Affinity matrix of 5 classes MNIST digits: (a) l_1-graph; (b) P by SSC in Eq. (4); and (c) $\mathbf{U}\mathbf{U}^T$ by GSC in Eq (6).

Method	Ncut	SSC	GSC
$K = 5$	59.85(6.98)	97.25(6.62)	**97.80(5.41)**
$K = 8$	55.75(6.31)	91.25(5.89)	**93.50(5.41)**
$K = 10$	55.25(5.91)	80.95(10.67)	**82.77(6.32)**
$K = 12$	51.35(6.98)	79.55(11.32)	**82.50(10.82)**
$K = 15$	50.47(5.07)	78.85(7.06)	**79.67(4.79)**
$K = 18$	50.10(4.76)	77.95(7.41)	**78.96(5.21)**

Table 4. Clustering results in terms of accuracy (%) and standard deviation on ORL dataset.

All the experimental cases show good convergent speed.

β	$K = 5$		$K = 10$		$K = 15$	
	SSC	GSC	SSC	GSC	SSC	GSC
β=0.00001	94.60	**96.20**	79.90	**81.30**	77.26	**77.73**
β=0.00005	94.60	**96.20**	79.70	**80.80**	77.07	**77.47**
β=0.0001	94.25	**96.00**	79.10	**80.96**	76.77	**77.69**
β=0.0005	95.60	**96.00**	80.60	**81.21**	77.13	**77.73**
β=0.001	95.80	**96.67**	80.70	**80.80**	77.45	**78.00**
β=0.005	90.00	**96.00**	75.50	**81.30**	70.20	**77.13**
β=0.01	78.25	**87.33**	59.80	**77.10**	54.53	**73.87**

Table 5. Clustering accuracy(%) of SSC and GSC against different β on ORL face datasets.

4.2.2 Experiments on Dimensionality Reduction

Dimensionality reduction (DR) is a method to represent high-dimensional data by their low-dimensional embeddings so that the low-dimensional data can be effectively used either in a pre-processing system, or for better understanding by avoiding the curse of dimensionality. It has been proved that DR is an important tool and DR has been widely used in many fields of data mining, data visualization, and machine learning.

Besides the superiority in clustering accuracy, another advantage of our Grassmannian manifold optimization is that it can get the embedded space given by matrix $\mathbf{U}$, whose rows are regarded as latent representation of original data. In the low-dimensional space defined by the rows of $\mathbf{U}$, we would expect that data clusters can be revealed.

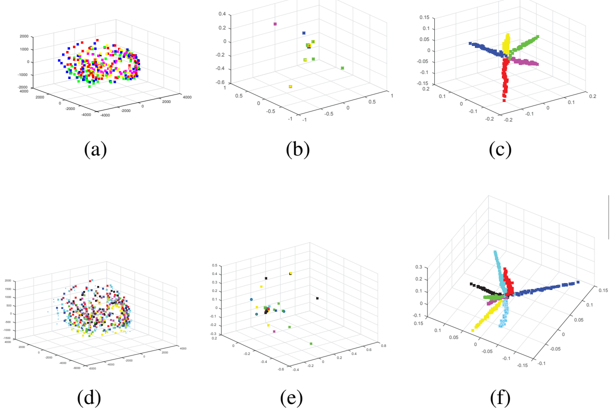


Figure 8. Visualization of the data dimension reduction of PCA of original data set (a) and (d); matrix $\mathbf{U}$ of SSC (b) and (e), and $\mathbf{U}$ of GDR (c) and (f): 5 classes case on the first row and 8 classes case on the second row for the YaleB faces data set.

To test our algorithm's DR capability we test GDR on the Yale B face dataset.

Figure 8(a) and (d) demonstrate the 3-dimensional scattering of 5 and 8 classes, respectively, from the Yale B faces data sets used in Section 4.2 after the Principal Component Analysis (PCA) linear dimensionality reduction method. Similarly, Figure 8(b) and (e) show the 3-dimensional scattering of the solution $\mathbf{U}$ of SSC by using the first 3 eigenvectors corresponding to the first 3 largest eigenvalues, and Figure 8(c) and (f) demonstrate the 3-dimensional scattering of the solution $\mathbf{U}$ given by our GDR. It clearly shows that GDR is the winner for meaningful visualization of datasets.

5. Conclusions

This paper proposes the GSC model which adopts Grassmann manifold optimization strategy to optimize the sparse spectral clustering objective introduced in [20] in a straightforward way, We also propose the GDR model which visualizes the latent representation of original data as the results from dimensionality reduction. Extensive experiments conducted on several real-world datasets demonstrate the effectiveness of our methods.

Acknowledgement

The research project is supported by the Australian Research Council (ARC) through the grant DP140102270 and partly supported by National Natural Science Foundation of China (Grant No. 61472155), and Huazhong University of Science and Technology Innovation Fund (Grant No. 2015QN130).

References

[1] P. A. Absil, R. Mahony, and R. Sepulchre. Riemannian geometry of Grassmann manifolds with a view on algorithmic computation. *Acta Applicadae Mathematicae*, 80(2):199–220, 2004.

[2] P.-A. Absil, R. Mahony, and R. Sepulchre. *Optimization algorithms on matrix manifolds*. Princeton University Press, 2008.

[3] F. R. Bach and M. I. Jordan. Learning spectral clustering, with application to speech separation. *Journal of Machine Learning Research*, 7:1963–2001, 2006.

[4] C. Bishop. *Pattern Recognition and Machine Learning*. Information Science and Statistics. Springer, 2006.

[5] N. Bounmal and P.-A. Absil. Low-rank matrix completion via preconditioned optimization on the Grassmann manifold. *Optimization-online*, pages 1–31, 2014.

[6] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, 2010.

[7] H. Chen, Y. Sun, J. Gao, and Y. Hu. Fast optimization algorithm on riemannian manifolds and its application in low-rank representation. *CoRR*, abs/1512.01927, 2015.

[8] J. Chen and J. Yang. Robust subspace segmentation via low-rank representation. *IEEE Transactions on Cybernetics*, 44(8):1432–1445, 2014.

[9] T. Connie, M. K. O. Goh, and A. B. J. Teoh. A grassmannian approach to address view change problem in gait recognition. *IEEE Transactions on Cybernetics*, 2016.

[10] N. Cristianini, J. Shawe-Taylor, and J. S. Kandola. Spectral kernel methods for clustering. In *Proceedings of Advances in Neural Information Processing Systems*, page 649655, 2001.

[11] J. P. Cunningham and Z. Ghahramani. Unifying linear dimensionality reduction. *arXiv:1406.0873*, 1, 2014.

[12] A. Edelman, T. A. Arias, and S. T. Smith. The geometry of algorithms with orthogonality constraints. *SIAM Journal of Matrix Analysis and Applications*, 20(2):303–353, 1998.

[13] E. Elhamifar and R. Vidal. Sparse subspace clustering: Algorithm, theory, and applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(11):2765–2781, 2013.

[14] Y. Fu, J. Gao, X. Hong, and D. Tien. Low rank representation on Riemannian manifold of symmetrical positive definite matrices. In *SIAM Conferences on Data Mining (SDM)*, pages 316–324, 2015.

[15] Y. Guo, J. Gao, and P. P. Kwan. Twin kernel embedding. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 30(8):1490–1495, 2008.

[16] M. Journée, F. Bach, P.-A. Absil, and R. Sepulchre. Low-rank optimization on the cone of positive semidefinite matrices. *SIAM Journal on Optimization*, 20(5):2327–2351, 2010.

[17] R. Langone, R. Mall, C. Alzate, and J. A. K. Suykens. *Unsupervised Learning Algorithms*, chapter Kernel Spectral Clustering and applications. Springer, 2016.

[18] G. Liu, Z. Lin, J. Sun, Y. Yu, and Y. Ma. Robust recovery of subspace structures by low-rank representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(1):171–184, 2013.

[19] G. Liu, Z. Lin, and Y. Yu. Robust subspace segmentation by low-rank representation. In *International Conference on Machine Learning*, pages 663–670, 2010.

[20] C. Lu, S. Yan, and Z. Lin. Convex sparse spectral clustering: Single-view to multi-view. *IEEE Transactions on Image Processing*, 25(6):2833–2843, 2016.

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

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

[23] F. J. Theis, T. P. Cason, and P.-A. Absil. Soft dimension reduction for ica by joint diagonalization on the stiefel manifold. In *International Conference on Independent Component Analysis and Signal Separation*, pages 354–361. Springer, 2009.

[24] R. Vidal. Subspace clustering. *IEEE Signal Processing Magazine*, 28(2):52–68, 2011.

[25] B. Wang, Y. Hu, J. Gao, Y. Sun, and B. Yin. Low rank representation on Grassmann manifolds. In *Proceedings of Asian Conference on Computer Vision (ACCV)*, 2014.

[26] B. Wang, Y. Hu, J. Gao, Y. Sun, and B. Yin. Product grassmann manifold representation and its lrr models. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI-16)*, Phoenix, United States, 2016.

[27] W. Wang and C. Lu. Projection onto the capped simplex. *CoRR*, abs/1503.01002, 2015.

[28] M. Yin, J. Gao, and Y. Guo. A nonlinear low-rank representation on Stiefel manifold. *Electronics Letters*, 51(10):749–751, 2015.