Principal Coordinate Clustering

Ali Sekmen*, Akram Aldroubi†, Ahmet Bugra Koku*‡, Keaton Hamm†
*Department of Computer Science, Tennessee State University
†Department of Mathematics, Vanderbilt University
‡Department of Mechanical Engineering, Middle East Technical University

***Abstract*—This paper introduces a clustering algorithm, called *principal coordinate clustering*. It takes in a similarity matrix S_W of a data matrix W and computes the singular value decomposition of S_W to determine the principal coordinates to convert the clustering problem to a simpler domain. It is a relative of spectral clustering, however, principal coordinate clustering is easier to interpret, and gives a clear understanding of why it performs well. In a fashion, this gives intuition behind why spectral clustering works from a more simple, linear algebra perspective, beyond the typical explanations via graph cuts, or other techniques. Moreover, it was demonstrated through experimentation on real and synthetic data that the proposed method performs equally well on average as spectral clustering, and that the method has the ability to scale quite easily to truly large data.**

***Index Terms*—Spectral clustering, principal component analysis, similarity matrix, clustering.**

I. Introduction

Currently, there are many methods which attempt to tackle the problem of *clustering* data lying in very high dimensional Euclidean space. From dimension reduction techniques to manifold learning to graph-based clustering algorithms, every method competes to take a set of data and reduce its complexity to make it more managable. Clustering methods strive to sort the data into meaningful subgroups, or clusters, which represent different natures represented therein.

Prior to implementing a clustering algorithm, often a *similarity matrix* related to the data is obtained which somehow captures the nature of how connected the data is. Spectral clustering in particular, takes a given similarity matrix to be the adjacency matrix of an undirected, weighted graph, which is then converted into a positive semi-definite matrix by taking its *graph Laplacian*. Subsequently, if there are assumed to be k clusters in the data, a matrix U is formed whose columns are the k eigenvectors of the graph Laplacian corresponding to its k largest eigenvalues. Finally, the rows of U are clustered using a traditional clustering technique such as k–means. Spectral clustering is also widely used in many subspace segmentation algorithms (as last step) that focus on clustering data that comes from a union of subspaces [1], [2], [3], [4], [5], [6], [7], [8].

In [9], it is stated that "*on the first glance spectral clustering appears slightly mysterious, and it is not obvious to see why it works at all and what it really does*," an assessment which is borne out by the many different viewpoints of what spectral clustering is. The goal of this paper is to present a simple algorithm which is quite similar in spirit to traditional spectral clustering, and moreover which performs essentially the same as spectral clustering on real and synthetic data, but whose success is more intuitively easy to understand. Moreover, the proposed method is exceedingly easy to implement, and in theory, requires no pre-processing of a similarity matrix, contrary to spectral clustering. Reflecting its similarity to principal component analysis, the algorithm is called *principal coordinate clustering*, and is derived simply from the Singular Value Decomposition of the similarity matrix.

Definition 1 (Similarity Matrix)**.** *Let* $\mathbf{W} \in \mathbb{R}^{m\times n}$ *be a data matrix. We say* $S_{\mathbf{W}}$ *is a* similarity matrix *for* $\mathbf{W}$ *if (i)* $S_{\mathbf{W}}$ *is symmetric, and (ii)* $S_{\mathbf{W}}(i,j) \neq 0$ *implies that* w_i *and* w_j *come from the same cluster, and* $S_{\mathbf{W}}(i,j) = 0$ *implies that* w_i *and* w_j *come from different clusters.*

II. Preliminaries

A. Graph Laplacian Matrix

Given a similarity matrix, S, the spectral clustering algorithms compute eigenvectors of a graph Laplacian matrix, which is typically in one of the following forms:

$$L := D - S \quad \text{unnormalized}$$
$$L_{rw} := I_n - D^{-1}S \quad \text{normalized}$$
$$L_{sym} := D^{-1/2}LD^{-1/2} \quad \text{symmetrically normalized,}$$

each of which can be shown to be positive semi-definite. Additionally, L and L_{sym} are symmetric.

Algorithm 1: Spectral Clustering Algorithm

Require: Assume columns of data matrix $\mathbf{W} = [w_1 \cdots w_n]$ are in $\mathbb{R}^m$.

- A similarity (affinity) matrix $S_{\mathbf{W}} = (s_{ij})$, where s_{ij} determines how close x_i and x_j in some sense. For example, $s_{ij} = exp(-||x_i - x_j||_2^2/2\sigma^2)$ if $i \neq j$ and $s_{ii} = 1$.
- The number of clusters, k.

1: Compute the diagonal degree matrix $D = Diag(d_1, \cdots, d_n)$ where $d_i = \sum_{j=1}^{n} s_{ij}$.
2: Compute a graph Laplacian matrix, L.
3: Compute the k eigenvectors $u_1, \cdots, u_k$ corresponding to k largest eigenvalues of the graph Laplacian.
4: Build the matrix $U_k = [u_1 \ \cdots \ u_k] \in \mathbb{R}^{n \times k}$.
5: Apply a traditional clustering technique (such as k-means) in $\mathbb{R}^k$ to the rows of U_k.

B. Singular Value Decomposition

Let $\mathbf{W} \in \mathbb{R}^{m \times n}$ with $m > n$. Then its Singular Value Decomposition (SVD) is given as follows:

$$\mathbf{W} = U\Sigma V^T$$

$$= \begin{bmatrix} u_1 & u_2 & \cdots & u_m \end{bmatrix} \begin{bmatrix} \sigma_1 & 0 & 0 & \cdots & 0 \\ 0 & \sigma_2 & 0 & \cdots & 0 \\ 0 & 0 & \ddots & 0 & 0 \\ 0 & 0 & \cdots & \sigma_{n-1} & 0 \\ 0 & 0 & \vdots & 0 & \sigma_n \\ 0 & 0 & \cdots & 0 & 0 \end{bmatrix} \begin{bmatrix} v_1^T \\ v_2^T \\ \vdots \\ v_n^T \end{bmatrix} \tag{II.1}$$

where $U \in \mathbb{R}^{m \times m}$, $\Sigma \in \mathbb{R}^{m \times n}$, and $V \in \mathbb{R}^{n \times n}$. If $\text{rank}(\mathbf{W}) = r$, then the skinny SVD of $\mathbf{W}$ is $U_r \Sigma_r V_r^T$, where U_r comprises the first r columns of U, $\Sigma_r = \text{Diag}(\sigma_1, \ldots, \sigma_r)$, and V_r^T comprises the first r rows of V^T. Note here that the singular values are in descending order $\sigma_1 \geq \ldots \geq \sigma_r$.

C. Matrix Operations

Given a matrix A, we define its *binary version* to be the matrix whose entries are 1 if the corresponding entry of A is nonzero and 0 otherwise. Similarly, the *absolute value version* of a matrix A is given by $abs(A)(i,j) = |A(i,j)|$.

III. Principal Coordinate Clustering

Algorithm 2 is the proposed clustering algorithm, which, while similar in spirit to spectral clustering, provides some natural advantages, which will be seen in the experimentation on both synthetic and real data below.

Algorithm 2: Principal Coordinate Clustering Algorithm

Require: Assume columns of data matrix $\mathbf{W} = [w_1 \cdots w_n]$ are in $\mathbb{R}^m$.

- A similarity (affinity) matrix $S_{\mathbf{W}} = (s_{ij})$,
- The number of clusters, k.

1: Compute rank-k skinny SVD of $S_{\mathbf{W}} = U_k \Sigma_k V_k^T$.
2: Build the matrix $\Sigma_k V_k^T \in \mathbb{R}^{k \times n}$.
3: Apply a traditional clustering technique (such as k-means) in $\mathbb{R}^k$ to the columns of $\Sigma_k V_k^T$.

Remark 1. *The graph Laplacian matrices L and L_{sym} are symmetric, and therefore their rank-k skinny SVD can be represented as $L = U_k \Sigma_k U_k^T$. The spectral clustering algorithm then clusters the columns of U_k^T, which correspond to the coordinates with respect to the basis vectors from the columns of $U_k \Sigma_k$. On the other hand, the principal coordinate (PC) clustering algorithm clusters the columns of $\Sigma_k U_k^T$, which are the principal coordinates with respect to the basis vectors $u_1, \ldots, u_k$.*

Remark 2. *One of the purposes of computing the graph Laplacian in the first step of spectral clustering is to obtain a positive semi-definite matrix. However, principal coordinate clustering requires no such positive semi-definite constraint.*

The method proposed here is based on a simple observation that the rank of $S_{\mathbf{W}}$ is equivalent to the number of clusters when the binary version of $S_{\mathbf{W}}$ is used. For the following illustrative example, we consider $\mathbf{W} = [w_1 \cdots w_n] \in \mathbb{R}^{15 \times 400}$ where the first 150 points come from one cluster, and remaining 250 come from another. Here, without loss of generality, we assume that $\mathbf{W}$ is sorted, so that data points belonging to the same cluster are next to each other. At the beginning, we also assume that there is no noise in $\mathbf{W}$.

A. An Illustrative Example

Let's first assume that we can find a perfect similarity matrix for $\mathbf{W}$ such that if w_i and w_j are in the same cluster, then $S_{\mathbf{W}}(i,j) = 1$, and $S_{\mathbf{W}}(i,j) = 0$ otherwise. Hence, if we plot this similarity matrix $S_{\mathbf{W}}$ as an image, we should get an image similar to the one illustrated in Fig. 1, where white regions correspond to 1 and indicate that these data points are in the same clusters, and black regions correspond to 0 and indicate otherwise.

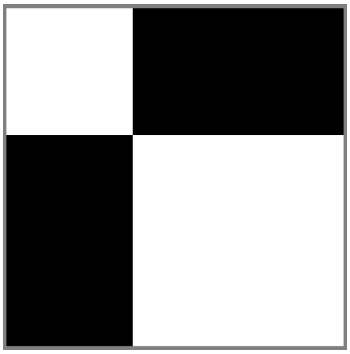

Fig. 1. An ideal similarity matrix.

If we consider this ideal similarity matrix $S_{\mathbf{W}}$ as a data matrix, then its rank should be 2 in this particular example, which is precisely the number of clusters in $\mathbf{W}$. Therefore, if $S_{\mathbf{W}} = U_2\Sigma_2V_2^T$, then the principal coordinates of $S_{\mathbf{W}}$ – i.e. the columns of $\Sigma_2V_2^T$ – will perfectly gather at exactly two points on the corresponding principal axes u_1 and u_2.

On the other hand, consider the slightly more general case where $\mathbf{W}$ is still noise free, but we do not have an ideal similarity matrix. A plot of the principal coordinates of $S_{\mathbf{W}}$ in this case is shown in Fig. 2.

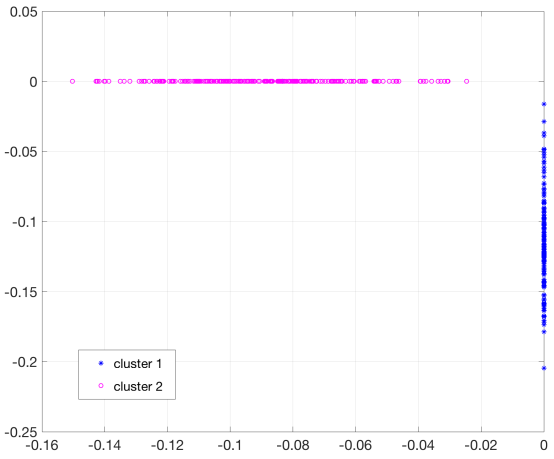

Fig. 2. Principal coordinates of a non-ideal similarity matrix with 2 clusters and no noise.

Here, rather than seeing only one point on each principal coordinate axis as in the ideal similarity matrix case, it can be seen that the columns spread out to some extent, but remain on the coordinate axes due to the absence of noise.

Next, consider a more general case where $\mathbf{W}$ contains noise, and hence the similarity matrix $S_{\mathbf{W}}$ (shown in Fig. 3) is no longer ideal. Further analysis of this similarity matrix reveals that even though the rank of $S_{\mathbf{W}}$ is no longer 2, the best estimate of the rank should still be 2. The principal coordinates of $S_{\mathbf{W}}$ according to Algorithm 2 are shown in Fig. 4 – i.e. these are the columns of $\Sigma_2V_2^T$.

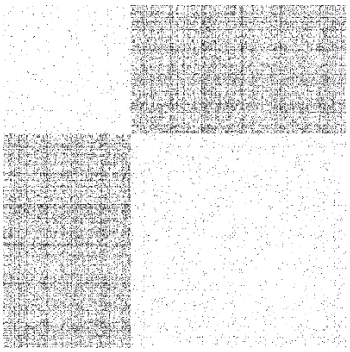

Fig. 3. Similarity matrix for $\mathbf{W}$ that contains low levels of noise.

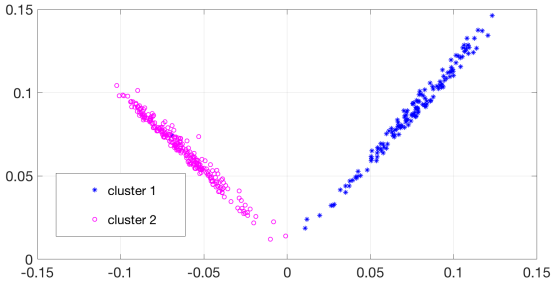

Fig. 4. Principal coordinates of a similarity matrix with 2 clusters and relatively low noise.

Finally, consider the general case where $\mathbf{W}$ contains a large amount of noise; here the similarity matrix $S_{\mathbf{W}}$ is far from ideal as shown in Fig. 5. Even though the boundaries between clusters in $S_{\mathbf{W}}$ are evidently fading, the best estimate of the rank is still 2. The principal coordinates of $S_{\mathbf{W}}$ are shown in Fig. 6.

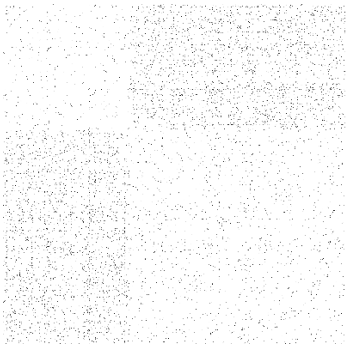

Fig. 5. Similarity matrix for $\mathbf{W}$ that contains high levels of noise.

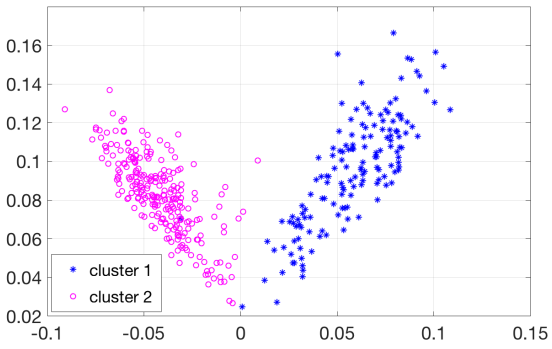

Fig. 6. Principal coordinates of a similarity matrix with 2 clusters and relatively high noise.

B. Remarks

To sum up, the principal coordinates of a similarity matrix $S_{\mathbf{W}}$ can be easily clustered for cases that are close to ideal. It is observed that as the data matrix $\mathbf{W}$ is contaminated with higher levels of noise, principal coordinates merge into each other which will naturally degrade the clustering performance. Particular examples so far are provided for 2 clusters in order for the results to be easier to plot and understand. Yet given that the principal axes (columns of U_k) are orthogonal, the proposed method nicely scales up for any cluster size k.

IV. Experimental Results

A set of experiments are conducted on real and synthetic data to understand how the aforementioned clustering method performs compared to the well-known spectral clustering algorithm. In all of the comparisons, the same similarity matrix is fed to both methods for clustering, and to make a consistent comparison, the final step in both algorithms is to use k means to cluster the columns (or rows) of the appropriate matrix.

A. Real Data – Motion Segmentation

The real data is related to a subspace clustering problem known as motion segmentation, and Hopkins155 [10] is a well-known benchmark dataset for this problem. For motion segmentation, the data comes from a union of low-dimensional subspaces which, in the ideal case, may be assumed to be independent. It has been shown [11] that in this case, taking the appropriate power of the absolute value version of the *shape interaction matrix* of Costeira and Kanade [12] yields a similarity matrix in the noiseless case. If $\mathbf{W}$ is the data matrix whose columns come from the union of low-dimensional independent subspaces, and whose skinny SVD is $\mathbf{W} = U_k \Sigma_k V_k^T$, then the corresponding shape interaction matrix is $V_k V_k^T$.

For motion data, each vector lies in a 4-dimensional subspace by nature [13]. Each set of data in the Hopkins155 dataset has a specified number of features. For instance, if one data matrix from the dataset has 3 features, each lying in a 4-dimensional space, then the rank of this matrix is well-estimated to be 12. To compare the performance of Algorithms 1 and 2 on the Hopkins155 dataset, for each of the 155 separate data matrices containing feature trajectories, the corresponding shape interaction matrix is used as $S_{\mathbf{W}}$ and fed into the algorithms with the rank of each matrix assumed to be 4 times the number of features as discussed above.

Results are shown in Fig. 7 where accuracy values corresponding to the same case are connected with a line. Given that all methods rely on k-means at the end, this comparison returns slightly different results on each run. Therefore, for the results to be statistically meaningful, this comparison is run 1000 times over the Hopkins155 dataset. Results given in Tables I-III show that, on the average of all 155 cases, PC clustering has similar performance to spectral clustering.

Remark 3. *In addition to the PC clustering algorithm, we also ran a normalized PC clustering algorithm, where the columns of $\Sigma_k U_k^T$ were normalized prior to running k-means. In all trials on the Hopkins155 dataset, the performance of PC and normalized PC clustering were the same. Thus, it is suspected that this normalization step is extraneous in general.*

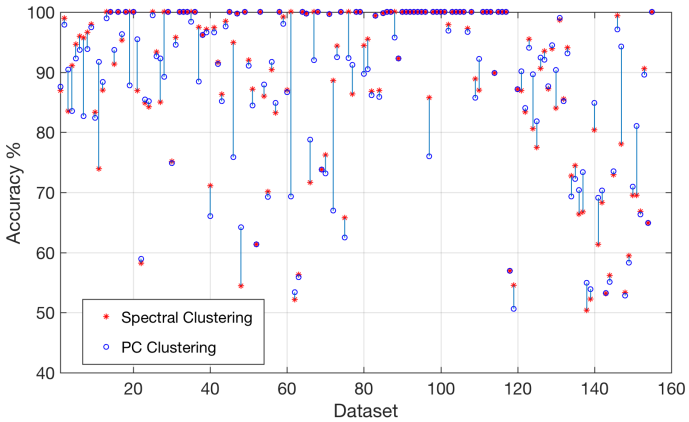

Fig. 7. Accuracy comparison of PC clustering and spectral clustering on Hopkins155 dataset.

TABLE I
% CLASSIFICATION ACCURACY FOR TWO MOTIONS.

Checker (78)	Spectral	PC
Average	90.29	88.97
Median	95.72	92.35
Traffic (31)	Spectral	PC
Average	97.58	96.61
Median	100.00	100.00
Articulated (11)	Spectral	PC
Average	88.69	88.81
Median	100.00	100.00
All (120 seq)	Spectral	PC
Average	92.03	90.93
Median	97.94	95.99

TABLE II
% CLASSIFICATION ACCURACY FOR THREE MOTIONS.

Checker (26)	Spectral	PC
Average	77.94	79.68
Median	80.52	84.45
Traffic (7)	Spectral	PC
Average	69.61	73.35
Median	69.53	70.99
Articulated (2)	Spectral	PC
Average	82.45	82.45
Median	82.45	82.45
All (35 seq)	Spectral	PC
Average	76.01	78.36
Median	77.52	81.83

B. Synthetic Data

Synthetic data experiments aim to compare the performance of the PC clustering algorithm to that of spectral clustering systematically. Unlike using real data, in this case we can control certain parameters and observe if there are any changes in performance as these parameters change. Level of noise in the data and the number of clusters are chosen as parameters to vary in the subsequent experiments.

To allow for comparison to the real data in Hopkins155, in what follows, data points in $\mathbf{W}$ are randomly drawn from a union of subspaces where each coordinate in a datapoint is uniformly randomly selected from interval $[-1,1]$. Then random Gaussian noise with zero mean and standard deviation σ is added independently to every element in $\mathbf{W}$.

Parameters used in the generation of the data matrix $\mathbf{W}$ are described below:

- D : The ambient space dimension
- n : The set of dimensions of subspaces from which the data is drawn
- N : The set of the number of data coming from each subspace
- σ : Standard deviation of noise added to the noise-free data matrix
- n_exp : Number of different experiments conducted

TABLE III
% CLASSIFICATION ACCURACY FOR ALL SEQUENCES.

All (155 seq)	Spectral	PC
Average	88.53	88.14
Median	94.63	92.31

In the remainder of the paper, these parameters as a set will be referred to as *process parameters*.

A derived term, the so called *occupancy factor* based on the process parameters is defined as follows:

$$\kappa = min\left(\frac{\sum_{i=1}^{|n|} n_i}{D}, 1\right) \tag{IV.1}$$

Observe that as the sum of subspace dimensions equals or surpasses the ambient space , D, the occupancy parameter κ saturates at 1. The effect of κ is also investigated along with the level of noise.

1) Effect of Noise Level: In the first set of experiments, the response of PC clustering and spectral clustering to the change in noise level, σ, and occupancy factor, κ, are investigated. Five different cases are investigated for 5 different values of D. In each case, the noise level σ is gradually increased. For D, σ and all other process parameters being fixed, n_exp experiments are run. In this context, an experiment refers to the clustering of one particular instance of $\mathbf{W}$ generated from the given process parameters. At the end of each run, the mean and standard deviation of the clustering accuracy of both methods are recorded.

The 5 test cases for different values of D have the following process parameters:

- $D = \{200, 150, 75, 30, 15\}$
- $n = \{5,5,5\}$
- $N = \{25,25,25\}$
- $\sigma = \{0.01, 0.05, 0.1, 0.15, 0.17, 0.19, 0.21\}$
- $n\_exp = 100$

Occupancy values that correspond to D for these 5 cases are κ=$\{0.075, 0.100, 0.200, 0.500, 1.000\}$. Results for Cases 1 to 5 are given in Tables IV-VIII respectively.

The results indicate that spectral clustering and PC clustering perform similarly given data with varying level of noise and occupation factors. For all cases, it is observed that both methods degrade in performance as the level of noise increases. It is also observed that both methods respond similarly to changes in occupancy

factor. In fact, as the occupancy factor increases, both methods better cope with higher levels of noise.

TABLE IV
% CLASSIFICATION ACCURACY FOR CASE 1: $\kappa = 0.075$.

Noise Level	Spectral Mean / STD	PC Mean / STD
0.01	100.00 / 0.00	100.00 / 0.00
0.05	99.91 / 0.34	99.89 / 0.36
0.10	99.21 / 0.93	99.16 / 0.96
0.15	91.71 / 4.24	90.95 / 4.78
0.17	80.96 / 7.96	79.71 / 8.53
0.19	68.05 / 10.53	65.61 / 10.91
0.21	54.55 / 7.62	53.68 / 7.30

TABLE V
% CLASSIFICATION ACCURACY FOR CASE 2: $\kappa = 0.100$.

Noise Level	Spectral Mean / STD	PC Mean / STD
0.01	100.00 / 0.00	100.00 / 0.00
0.05	99.92 / 0.32	99.92 / 0.32
0.10	99.21 / 1.04	99.07 / 1.22
0.15	92.89 / 4.60	92.00 / 4.79
0.17	84.33 / 7.41	82.99 / 7.62
0.19	73.35 / 9.72	72.75 / 10.40
0.21	59.88 / 9.55	58.49 / 9.51

TABLE VI
% CLASSIFICATION ACCURACY FOR CASE 3: $\kappa = 0.200$.

Noise Level	Spectral Mean / STD	PC Mean / STD
0.01	100.00 / 0.00	100.00 / 0.00
0.05	99.93 / 0.29	99.92 / 0.37
0.10	99.28 / 0.96	99.31 / 0.92
0.15	95.79 / 3.11	95.47 / 3.17
0.17	90.19 / 5.18	89.11 / 5.80
0.19	80.49 / 9.27	78.40 / 10.54
0.21	69.77 / 11.21	68.16 / 10.67

TABLE VII
% CLASSIFICATION ACCURACY FOR CASE 4: $\kappa = 0.500$.

Noise Level	Spectral Mean / STD	PC Mean / STD
0.01	100.00 / 0.00	100.00 / 0.00
0.05	99.92 / 0.32	99.88 / 0.38
0.10	99.01 / 1.13	98.96 / 1.27
0.15	94.83 / 3.39	94.51 / 3.53
0.17	91.09 / 4.23	90.43 / 4.83
0.19	83.05 / 8.93	82.43 / 8.90
0.21	74.01 / 10.37	72.76 / 10.23

2) Effect of Number of Clusters: In studying the performance of both clustering methods when the number of clusters in the data changes, 10 more cases are tested. The first 5 cases have common process parameters as follows:

TABLE VIII
% CLASSIFICATION ACCURACY FOR CASE 5: $\kappa = 1.000$.

Noise Level	Spectral Mean / STD	PC Mean / STD
0.01	99.97 / 0.19	99.96 / 0.23
0.05	99.57 / 0.82	99.49 / 0.86
0.10	96.89 / 2.41	96.53 / 2.62
0.15	86.47 / 8.01	84.95 / 8.61
0.17	81.72 / 9.48	80.80 / 9.37
0.19	74.99 / 10.97	74.29 / 10.18
0.21	66.60 / 10.33	65.28 / 9.99

- $D = 50$
- $\sigma = 0.05$
- $n\_exp = 100$

where the number of clusters and number of data in each cluster is as follows for each case:

Case 6: Two clusters of dimension 5 with 50 data points in each. Hence, $n = \{5,5\}$, $N = \{50,50\}$.

Case 7: Three clusters of dimension 5 with 50 data points in each. Hence, $n = \{5,5,5\}$, $N = \{50,50,50\}$.

Case 8: Five clusters of dimension 5 with 50 data points in each, n and N scale similarly.

Case 9: Seven clusters of dimension 5 with 50 data points in each.

Case 10: Ten clusters of dimension 5 with 50 data points in each.

In Cases 6–10, given that the sum of cluster dimensions is less than or equal to the ambient space dimension, these clusters correspond to independent subspaces. Results for these cases are given Table IX, and indicate that both spectral clustering and PC clustering perform similarly. It is also observed in this case PC clustering does not favor higher occupancy levels.

TABLE IX
% CLASSIFICATION ACCURACY FOR CASES 6-10.

Noise Level	Spectral Mean / STD	PC Mean / STD
Case 6	99.98 / 0.20	99.98 / 0.20
Case 7	99.95 / 0.26	99.93 / 0.29
Case 8	99.94 / 0.22	99.92 / 0.24
Case 9	99.90 / 0.22	99.70 / 1.94
Case 10	99.82 / 0.28	96.13 / 6.22

It is known that the Shape Interaction Matrix does not work well as a similarity matrix for data coming from non-independent subspaces. Hence, a final comparison for 5 more cases is considered where the data comes

from subspaces that are not necessarily independent. For these cases, common process parameters are as follows:

- $D = 10$
- $\sigma = 0.05$
- $n\_exp = 100$.

In Cases 11–15, the values of n and N are the same as Cases 6–10 respectively. Observe that all of these cases share a common occupancy factor of $\kappa = 1.0$. For Case 11, the subspaces are still independent, but for all remaining cases, the data comes from subspaces that are not independent.

Results for Cases 11–15 are given in Table X. These results also indicate that both algorithm degrade in performance similarly in clustering data from non-independent subspaces.

TABLE X
% CLASSIFICATION ACCURACY FOR CASES 11-15.

	Spectral	PC
Noise Level	Mean / STD	Mean / STD
Case 11	99.36 / 1.10	99.40 / 1.04
Case 12	95.92 / 2.80	95.63 / 3.02
Case 13	62.57 / 8.61	62.15 / 8.59
Case 14	45.01 / 6.22	45.17 / 5.22
Case 15	34.56 / 2.58	35.46 / 2.73

V. CONCLUSIONS

A relative of spectral clustering, which we call *principal coordinate (PC) clustering*, was introduced. It was shown that performing k-means on the k principal coordinates (which come from the singular value decomposition) of a similarity matrix yields similar results to doing the same for the k eigenvectors of the graph Laplacian corresponding to its k largest eigenvalues. However, PC clustering is remarkably easier to interpret, and gives a clear understanding of why it performs well. In a fashion, this gives intuition behind why spectral clustering works from a more simple, linear algebra perspective, beyond the typical explanations via graph cuts, or other techniques. Moreover, it was demonstrated through experimentation on real and synthetic data that the proposed method performs equally well on average as spectral clustering, and that the method has the ability to scale quite easily to truly large data. In this study, the objective was not to get the best performance, but rather to show that PC clustering is similar to spectral clustering. There are numerous improvements of basic spectral clustering methods in the literature, which would thus likely enhance the performance of PC clustering as well.

ACKNOWLEDGEMENT

The research of A. Sekmen, and A. Koku is supported by DoD Grant W911NF-15-1-0495. The research of A. Aldroubi is supported by NSF Grant NSF/DMS 132099. The research of A. Koku is also supported by TUBITAK-2219-1059B191600150.

REFERENCES

[1] E. Elhamifar, R. Vidal, Sparse subspace clustering: Algorithm, theory, and applications, CoRR abs/1203.1005.
URL http://arxiv.org/abs/1203.1005

[2] E. Elhamifar, R. Vidal, Sparse subspace clustering: Algorithm, theory, and applications, IEEE Transactions on Pattern Analysis and Machine Intelligence.

[3] R. Vidal, R. Tron, R. Hartley, Multiframe motion segmentation with missing data using powerfactorization and GPCA, International Journal on Computer Vision 79 (1) (2008) 85–105.

[4] G. Liu, Z. Lin, S. Yan, J. Sun, Y. Yu, Y. Ma, Robust recovery of subspace structures by low-rank representation, arXiv.org.
URL http://arxiv.org/abs/1010.2955

[5] G. Liu, Z. Lin, Y. Yu, Robust subspace segmentation by low-rank representation, in: International Conference on Machine Learning, 2010, pp. 663–670.

[6] R. Vidal, A tutorial on subspace clustering, IEEE Signal Processing Magazine 28 (2010) 52–68.

[7] G. Chen, S. Atev, G. Lerman, Kernel spectral curvature clustering (kscc), in: 4th international workshop on Dynamical Vision, 2009.

[8] P. Ji, M. Salzmann, H. Li, Shape interaction matrix revisited and robustified: Efficient subspace clustering with corrupted and incomplete data, CoRR abs/1509.02649.
URL http://arxiv.org/abs/1509.02649

[9] U. V. Luxburg, A tutorial on spectral clustering, Statistics and Computing 17 (2007) 395–416.

[10] R. Tron, R. Vidal, A benchmark for the comparison of 3-d motion segmentation algorithms, in: IEEE Conference on Computer Vision and Pattern Recognition, 2007, pp. 1–8.

[11] A. Aldroubi, A. Sekmen, A. B. Koku, A. F. Cakmak, Similarity matrix framework for data from union of subspaces, Submitted to Applied and Computational Harmonic Analysis - Under Review.

[12] J. Costeira, T. Kanade, A multibody factorization method for independently moving objects, International Journal of Computer Vision 29 (3) (1998) 159–179.

[13] K. Kanatani, C. Matsunaga, Estimating the number of independent motions for multibody motion segmentation, in: 5th Asian Conference on Computer Vision, 2002, pp. 7–9.