Revealing Community Structures by Ensemble Clustering using Group Diffusion

Elena Ivannikova[a], Hyunwoo Park[b], Timo Hämäläinen[a], Kichun Lee[c,*]

[a]Faculty of Information Technology, University of Jyväskylä, POBox 35 (Agora), 40014 Jyväskylä, Finland
[b]Management Sciences at the Fisher College of Business, The Ohio State University, 2100 Neil Ave, Columbus, OH 43210
[c]Industrial Engineering, Hanyang University, Engineering Center #705-1, 222 Wangsimni-ro Seongdong-gu Seoul 133-791, Korea

Abstract

We propose an ensemble clustering approach using group diffusion to reveal community structures in data. We represent data points as a directed graph and assume each data point belong to single cluster membership instead of multiple memberships. The method is based on the concept of ensemble group diffusion with a parameter to represent diffusion depth in clustering. The ability to modulate the diffusion-depth parameter by varying it within a certain interval allows for more accurate construction of clusters. Depending on the value of the diffusion-depth parameter, the presented approach can determine very well both local clusters and global structure of data. At the same time, the ability to combine single outcomes of the method results in better cluster segmentation. Due to this property, the proposed method performs well on data sets where other conventional clustering methods fail. We test the method with both simulated and real-world data sets. The results support our theoretical conjectures on improved accuracy compared to other selected methods.

Keywords: Clustering, Diffusion, Markov chain, Social network, Community structure

*Corresponding author at: Industrial Engineering, Hanyang University, Engineering Center #705-1, 222 Wangsimni-ro Seongdong-gu Seoul 133-791, Korea. Tel.: +82-2-2220-0478.
Email addresses: `elena.v.ivannikova@student.jyu.fi` (Elena Ivannikova), `park.2706@osu.edu` (Hyunwoo Park), `timo.t.hamalainen@jyu.fi` (Timo Hämäläinen), `skylee@hanyang.ac.kr` (Kichun Lee)

1. Introduction

Interest in the analysis of complex networks has rapidly grown over the past few years. Network models have been used in different areas including economics, biology, social sciences, and computer science, where systems are often represented as graphs. Analyzing network models in practice is a challenging task due to the complexity of the networks, particularly when the underlying community structure is unknown. There are two general approaches to reveal the community structure of networks. The first approach is graph partitioning when the number of clusters is known. The second approach, called community structure detection, is more challenging, as it divides a network into clusters or groups graph nodes when the number of clusters is unknown beforehand. For community structure detection, both identifying clusters and determining the number of clusters must be solved simultaneously.

The detection of community structures in an arbitrary graph is a challenging task. In recent years, several methods have been developed and applied, including min-cut based approaches, clique based approaches, modularity based approaches, clustering approaches, and so forth [1]. These approaches share an essential tool, clustering, in a sense to find good clusters of nodes in a graph that improve a certain criterion. Clustering, indeed, is a universal tool applied in many different fields of data analysis, such as data mining, statistics, marketing, and others [2]. The goal of cluster analysis is to partition data into groups or clusters based on pairwise similarity so that observations inside one cluster are more similar than the ones belonging to different clusters [3]. The dimensionality of the data set has a strong influence on the performance of clustering algorithms. Some methods work well for low-dimensional data, whereas they are unable to find structure in high-dimensional data sets. High-dimensional data pose a number of challenges for researchers and practitioners. First of all, high-dimensional data are more likely to be sparse, which makes it difficult for algorithms to find structures in the data. Moreover, in high-dimensional data sets, points may belong to diverse clusters in different subspaces. Capturing the geometric structure of the manifold from the data, whether low- or high-dimensional, plays an essential role in reliable clustering results. Clustering methods without considering such geometric structures can fail to produce accurate results and find mere local structures in sparse high-dimensional data.

In addition to geometric structures of real-world data, another challenge for clustering in community detection tasks is that the results cannot be validated as there is no ground truth available for the data sets, as in supervised learning. Furthermore, various clustering methods often generate different and biased structures in a data set due to different optimization criteria they adopt. To overcome these issues in different clustering results, combining multiple partitions can improve the quality of clustering results significantly. In this sense, a collective approach called clustering ensemble aims to provide more robust, stable, and novel solutions by leveraging a consensus of multiple clustering runs. The main goal of clustering ensemble algorithms is to define a clustering solution that maximizes a consensus function and to select a partition generation procedure. Partitioning can be performed using (1) data resampling [4, 5], (2) different parameter values or initializations [6, 7], and (3) different clustering algorithms such as k-means, density-based, graph-partitioning-based, and statistics-based [8].

In this paper, we propose an ensemble clustering algorithm based on the concept of ensemble group diffusion, denoted by Ensemble Group Diffusion, EGD. The proposed method takes into account not only the geometric structure of data using group distances, but also the diffusion of group distances across connectivity scales. Moreover, the presented method is able to produce more robust clustering results by collectively integrating views from individual diffusion scales. The method is also capable of reasonably regulating cluster sizes by an admitted group dependence level. In addition, it nicely handles directed graphs as opposed to other approaches. Further, we present a detailed analysis of the degree of the collective integration and propose a guideline for parameter settings. We demonstrate the efficacy of the proposed method using not only an illustrative simple example, demonstration test cases and simulation studies, but also real-world data sets such as the researcher collaboration network in a healthcare system from the National Institute of Health (NIH) in the U.S., a consumer behavioral pattern captured by the co-purchasing network from Amazon, a leading consumer online shopping place in the U.S. and a gene-expression microarray data sets.

The rest of the paper is organized as follows. Section 2 reviews clustering techniques used in community detection. Section 3 describes preliminary concepts for the proposed algorithm. Section 4 provides theoretical justification for the proposed clustering method. Section 5 compares the performance of the proposed method to popular and state-of-the-art clustering algorithms under different settings and data sets. Section 6 discusses the implications of our development of the algorithm and concludes the paper.

2. Related work

Many clustering algorithms have been proposed in the literature of community structure detection and clustering analysis. Modularity-based methods established by Newman [9] have shown exceptional performance in many cases [10, 11, 12]. These methods are nonparametric and are designed to maximize the modularity as an objective function. These methods, however, fail to detect smaller communities in some cases where such granular identification is desirable. It is hard to say whether the detected clusters are indeed single communities or clusters of smaller communities. For example, Fortunato and Barthélemy [13] analyzed modularity-based methods and their applicability in the area of community detection. Their research points out that the modularity function has a resolution limit. Communities that are smaller than a threshold in a certain criterion may not be revealed, even when the whole graph is identified as a single community. In addition, working with pairwise similarity between nodes, modularity-based methods are inherently unable to handle directional relationships commonly observed in reality.

Other clustering methods for community detection also exist. Hierarchical clustering [3], agglomerative or divisive, is another technique commonly used for community detection. Hierarchical clustering [3] first defines a similarity measure between clusters and computes a similarity matrix between vertices of a graph. Among the most critical disadvantages of hierarchical clustering is that the results can be different depending on the similarity measure used, although it is a universal phenomenon in most clustering methods. Besides, agglomerative hierarchical clustering does not scale well, which is crucial for clustering graphs [14]. In essence, approaches based on a predefined number of clusters require an additional important step that involves a decision criterion for the optimal number of clusters. Spectral clustering [3] refers to the group of methods based on eigenvalue decomposition of the similarity matrix or its derivative matrices for clustering data sets. This approach is good at finding non-convex clusters, able to take into account geometric structures of the data [3]. However, it works with similarity matrices, which reflect only bidirectional relationships among the nodes in a graph. The result depends on the number of selected eigenvectors. Along with spectral and density-based methods considering geometric structures of data, Park and Lee [15] proposed a group-dependence clustering approach. This approach is based on the idea of maximizing a measure called group dependence. The central idea of the method is that any two nodes in the graph can be considered as being connected through Markovian transitions. This conceptualization allows for the calculation of 'dependence distance' [16] between graph nodes in a certain evolution step, which can adjust the level of connectivity scale in group assignment. Though the method supports the ability to adjust the level of the connectivity scale in clustering, it fails to present a collective view of clusters according to the connectivity scale and was insufficient in coping with directional structures between nodes. Density-based methods detect clusters according to the local density of data points. Based on a density threshold, the points from disconnected regions of high density are assigned to different clusters when the rest are marked as noise. However, such methods, computationally expensive, are suitable only for data defined by a set of coordinates. To overcome these drawbacks, an alternative approach, called clustering by 'fast search and find of density peaks' (FSFDP) [17], defines the cluster centers as local density maxima that are relatively distant from any point of higher local density. After that, each remaining point is assigned to the same cluster as its nearest neighbor of higher density.

Attempts to improve the quality of clustering results brought forth developing a number of ensemble clustering approaches during recent years. Zheng *et al.* used aggregated distance matrices and combined both partitional clustering and hierarchical clustering results [18]. Wang *et al.* used a Bayesian graphical model to aggregate mixed cluster results and maximized an approximation of the posterior distribution [19]. The clustering approach, proposed in [8] as one of the state-of-the art approaches, addresses the problem of combining multiple partitions of a set of objects using the knowledge-reuse framework [20]. It formulates the cluster ensemble problem by introducing an objective function for combining multiple clustering solutions and by solving the corresponding optimization problem. This way the final consensus solution is obtained without accessing original features. The authors propose the following three consensus functions: cluster-based similarity partitioning algorithm (CSPA) based on a measure of pairwise similarity, HyperGraph Partitioning Algorithm (HGPA) based on approximation of the maximum mutual information objective, and meta-cLustering algorithm (MCLA) based on solving a cluster correspondence problem. The final solution is selected among the three consensus clusterings as the one with the highest average mutual information.

3. Preliminary Concepts

In this section, we briefly summarize the concept of group dependence that links data points by a Markov random walk and the concept of a clustering ensemble that combines several division outcomes. Then, in the next section, we propose the concept of ensemble group diffusion to measure a multiscale cohesion level for a group division in an integrative fashion. Ensemble group diffusion gives rise to a new clustering method for community detection, which will be discussed in detail in regards to its characteristics and parameters.

3.1. Group Dependence

The concept of group dependence is closely related to the Markov random walk and provides a general foundation for a distance measure between data points that considers the geometrical structure [21]. Group dependence proposed by Park and Lee [15] is a measure that quantifies the goodness-of-division of an undirected graph. In this paper let us suppose that a directed graph of data points (nodes) $x_1, \cdots, x_n \in \mathbb{R}^b$ is given. Denote the set of data points by $\Omega = \{x_1, \ldots, x_n\}$. We view the graph as a Markov chain, assuming the whole chain is ergodic and all transitions follow the Markovian property.

We start with a simple case of bisecting the graph, then providing instructions on how to divide it into more than two groups in the following section. Let $s_i = 1$, a decision variable, if data point i belongs to group 1 and $s_i = -1$ if it belongs to group 2. Observe that the quantity $(s_i s_j + 1)/2$ is 1 if i and j are in the same group and 0 otherwise. Denote the group assignment vector by $\mathbf{s} = [s_1, \ldots, s_n]$. Group dependence is defined as follows:

Definition 1. Group dependence D_t for a given group assignment $\mathbf{s}$ and connectivity scale parameter t is

$$D_t = \sum_{x_i, x_j \in \Omega} \big(Dep(X_0 = j, X_t = i) - 1\big) \frac{(s_i s_j + 1)}{2},$$

in which $Dep(X_0 = j, X_t = i)$, as dependence, is defined by $\frac{P(X_0=j, X_t=i)}{P(X_0=j)P(X_t=i)}$ and t, as an exogenously given parameter, means the t-step-wide neighborhood evolution in Ω.

Dependence is closely linked to the point-wise mutual information in information theory and the lift measure in association rule learning. Intuitively speaking, dependence captures how x_j in the initial state is inter-dependent with x_i at the t-th step: $Dep(X_0 = j, X_t = i) < 1$ means that j and i are negatively dependent; $Dep(X_0 = j, X_t = i) = 1$ means that they are independent; $Dep(X_0 = j, X_t = i) > 1$ means that they are positively dependent. The term, $Dep(X_0 = j, X_t = i) - 1$, represents the degree of relative dependence in comparison to the level of independence as the reference point. Accordingly, group dependence D_t measures the overall coherence of group assignment $\mathbf{s}$ in terms of dependence for the whole data set at the t-step transition. Based on group dependence, we propose another measure of ensemble group diffusion to reflect the multiscale dependence structure in a directed graph. We then look for a good group assignment $\mathbf{s}$ of all n points to maximize the measure.

3.2. Clustering Ensemble

As the idea of ensemble group diffusion closely relates to clustering ensembles, we briefly introduce the basic concept of a clustering ensemble. Cluster ensembles basically address the problem of combining multiple base clustering results for the same data set into a final consensus solution. Depending on how to reach a consensus solution, several approaches (such as graph-based, matrix-based, and probabilistic models [22]) exist in the literature. However, the problem formation in cluster ensembles is universal as follows. We start with a base clustering algorithm that generates the group assignment $\mathbf{s}$ of the data points in Ω. We prepare M base clustering results by supplying different parameters to one base algorithm. From them, we obtain M different group assignments $\mathbf{s}^{(1)}, \cdots, \mathbf{s}^{(M)}$. The results from the M base clustering algorithms can be stacked together to form an overall clustering matrix. Given the overall clustering matrix, the cluster ensemble problem is to combine the M base clustering results for the n data points to generate a consensus clustering, which should be more accurate and stable than the individual base clusterings.

In this paper, we calculate diffusion matrices of dependence for each parameter value of connectivity level t and aggregate the diffusion matrices. Specifically, we similarly start with a directed graph of n data points in Ω with probability matrix P. Having a set of possible parameters, denoted by T, we calculate $[Dep(X_0 = j, X_t = i)]_{i,j=1,\cdots,n}$ for every $t \in T$ and then obtain a cumulative matrix, the ensemble group diffusion, as the sum of the individual group diffusion results:

$$D(\mathbf{s}) = \sum_{t \in T} D_t = \sum_{t \in T} \sum_{x_i, x_j \in \Omega} \big(Dep(X_0 = j, X_t = i) - 1\big) \frac{(s_i s_j + 1)}{2}. \quad (1)$$

The final clustering is obtained through solving the maximization problem for the cumulative matrix $D(\mathbf{s})$. Combining individual diffusion matrices to ensemble group diffusion can be viewed as a peer regularization. Diffusion matrices obtained with a small t value pull the overall solution towards having smaller clusters. Similarly, diffusion matrices from a large t value shift the optimal solution towards coarser and larger scale representations. The construction of ensemble group diffusion brings stable clustering results under various degrees of resolution and heterogeneous structures in the data set. Thus, it aims to solve the resolution limit issue in which an obvious small-sized community is rarely detected when the whole graph is sufficiently large.

4. Clustering with Ensemble Group Diffusion

We incorporate group dependence and ensemble clustering to present a new approach of ensemble group diffusion that finds the community structure in a directed graph. Given a transition matrix P, we denote the one-step backward transition matrix by P_B, which is calculated from P. Then by backward Markovian transitions, the t-step transition matrix is P_B^t: $P_{B;i,j}^t = P(X_0 = j | X_t = i)$. We observe that if x_i and x_j are close in the geometric structure of the data, the backward transition probability should be large. The posterior transition probability involves a backward Markov chain, representing the probability of the initial state j after reaching state i at the t-th step transition as a measure of the difference between the two states i and j in the directed graph.

In particular, the backward transition probability $P_{B;i,j}^t$ should be at least greater than the probability that x_i and x_j are connected by chance among all data points. Thus, the greater $P_{B;i,j}^t$ is among the data points in a cluster, the better the cluster is. Also, a partition of the data set is meaningful when a whole connectivity level by the partition should increase more than that by a random configuration. The quantity $\sum_{i,j}(P^t_{B;i,j} - 1/n)$ should be great for all x_i and x_j pairs in the same cluster, where the threshold probability $1/n$ represents the random probability among n data points. If one seeks a tight configuration, one may use a value larger than $1/n$. In quantifying the connectivity level in detail, we use not only the concept of geometric diffusion, but also modulate the diffusion depth parameter t by varying t in a certain interval. Thus, we define the collective geometric diffusion to be

$$\sum_{t \in T} \sum_{i,j} (P^t_{B;i,j} - 1/n),$$

where T is the set of diffusion depth values.

We denote the group assignment vector by $\mathbf{s} = [s_1, \ldots, s_n]$. Let $s_i = 1$ if data point i belongs to group 1 and $s_i = -1$ if it belongs to group 2. To obtain a good partition in terms of collective geometric diffusion, we solve the following programming:

$$\arg\max_{\mathbf{s}} \sum_{t \in T} \sum_{i,j} (P^t_{B;i,j} - 1/n) \frac{(s_i s_j + 1)}{2}. \tag{2}$$

We show that collective geometric diffusion is in proportion to ensemble group diffusion when all initial states have equal probability as non-informative prior because

$$\begin{aligned}
\sum_{t \in T} \sum_{i,j} (P^t_{B;i,j} - 1/n) &= 1/n \sum_{t \in T} \sum_{i,j} (P(X_0 = j | X_t = i)n - 1) \\
&= 1/n \sum_{t \in T} \sum_{i,j} \left(\frac{P(X_0 = j, X_t = i)}{P(X_t = i)\frac{1}{n}} - 1 \right) \\
&= 1/n \sum_{t \in T} \sum_{i,j} \left(\frac{P(X_0 = j, X_t = i)}{P(X_t = i) P(X_0 = j)} - 1 \right) \\
&= 1/n \sum_{t \in T} \sum_{i,j} (Dep(X_0 = j, X_t = i) - 1).
\end{aligned}$$

Thus, the programming in (2) is equivalent to maximizing the ensemble group diffusion

$$\arg\max_{\mathbf{s}} \sum_{t \in T} \sum_{i,j} (Dep(X_0 = j, X_t = i) - 1) \frac{(s_i s_j + 1)}{2}.$$

Note again that the quantity $(s_i s_j + 1)/2$ is 1 if i and j are in the same group and 0 otherwise. Thus, an optimal clustering scheme is achievable through maximizing the collective geometric diffusion measure by varying the group assignment $\mathbf{s}$ of all n points. To express the level of closeness to a group, the group identity s_i is extended from discrete to continuous with the norm of $\mathbf{s}$ fixed. By the set of diffusion depths T, we can effectively adjust the level of the connectivity scale for which two points are associated. When infinite diffusion steps are taken, for the infinite value of t, the Markov chain converges to the stationary distribution and the collective geometric connectivity becomes trivial. For instance, one can set T to be $\{1, 2\}$ as a short-range scale or $\{5, 6, 7, 8\}$ as a mid-range scale. In practice, one could start with T as a long-range scale such as $\{1, \cdots, 8\}$ and, depending on the result, shrink it, or vice versa (start with T as a short-range scale and expand it). We will see the effect of T by varying it in the experiment section. We express the maximization of the ensemble group diffusion with respect to $\mathbf{s}$ subject to $\|\mathbf{s}\| = 1$ as follows:

$$\arg\max_{\mathbf{s}} \quad \mathbf{s}^\top \sum_{t \in T} \left(P_B^t - 1/n \mathbf{1}\mathbf{1}^\top \right) \mathbf{s} := \arg\max_{\mathbf{s}} \quad \mathbf{s}^\top G \mathbf{s}, \tag{3}$$

where

$$G = \sum_{t \in T} \left(P_B^t - 1/n \mathbf{1}\mathbf{1}^\top \right). \tag{4}$$

We numerically find the eigenvalues and eigenvectors of G. The existence of the largest and positive eigenvalue and its eigenvector $\mathbf{s}_1$ implies that the ensemble group diffusion is maximally increased by adjusting a division of the data set on the direction of the corresponding eigenvector $\mathbf{s}_1$. We mention the computational hurdle is computing eigenvalues and eigenvectors of G, and we compare its running times in the experiment section. Moreover, the division of the data points is based on the signs of $\mathbf{s}_1$. In fact, $\mathbf{s}_1$ is a one-dimensional representation of the data points, which can be used for a general purpose such as visualization and classification because the sign and magnitude of $\mathbf{s}_1$ relate to the degree of closeness to one group against the other. We note that the eigenvector associated with the zero eigenvalue represents assigning all data points to just one group. On the contrary, nonexistence of a positive eigenvalue implies any further division of the data yields no gain.

4.1. Detecting More Than Two Groups

The procedure explained so far either divides a graph into two groups or decides not to divide further. It is natural to consider a network with more than two groups latent in its community structure. To obtain more than two clusters, we adopt a standard approach to subsequently divide the groups found [9, 15]. We look for a possible division for each group found in the previous step by constructing a new backward transition matrix $P_{B|g}$ for a detected group g as a subset of the data set: $P_{B|g}$ with size $|g| \times |g|$, defined by

$$P^t_{B|g;i,j} = \{P(X_0 = j|X_t = i)|\forall i, j \in g\}.$$

Following the same procedure based on $P_{B|g}$, the solution of $\arg\max_{\|\mathbf{s}\|=1} G^{(g)}$ in (3) enables us to decide whether a division is possible or not.

It is important to note that the new group configuration with a new division found does not always result in an increase in the ensemble group diffusion of the whole graph because the new similarity matrix $P_{B|g}$ reflects only a part of the whole data set without considering connections to the nodes belonging to other groups. Hence, among the possible divisions, we look for only the division which causes the ensemble group diffusion in (3) for the whole data set to increase most when the new division is applied.

Furthermore, we require that the increase is at least by certain margin. Specifically, for the purpose of regularizing the solution, we introduce a dependence gain parameter $\delta_d \in [0, 1]$. This parameter is used for calculating the minimal dependence gain value required to split a cluster into two sub-clusters and is defined as

$$\Delta_d = \delta_d \sum_{g_{i,j}>0} G,$$

where $G = [g_{i,j}]$ is defined in (4). For setting the value of δ_d in practice, we recommend starting with quite a small value, close to zero, and increasing it depending on the results. One can verify that $\sum_{g_{i,j}>0} G$ is the upper bound for cumulative collective geometric diffusion. Thus, the division into sub-clusters proceeds if the difference between the collective geometric diffusion values corresponding to the group configurations before and after the division is higher than the dependence gain:

$$D(\mathbf{s}') - D(\mathbf{s}) > n\Delta_d,$$

where D is defined in (1), $\mathbf{s}$ and $\mathbf{s}'$ denote the group configurations before and after the division into sub-clusters, respectively, and n stands for the size of the set of data points Ω. The dependence gain parameter prevents the algorithm from identifying clusters that are too small or not clear enough, which allows controlling the desired level of clarity in finding clusters. In summary, we stop dividing the group when we find no positive eigenvalues from group ensemble matrix $G^{(g)}$ or the dependence gain fail to exceed $n\Delta_d$.

4.2. Illustrative Examples

This section demonstrates the performance and steps of the proposed algorithm. For illustration, we construct a simulated similarity matrix, which is generated as follows. We randomly generate 500 points in two dimensional space where every point belongs to one of three clusters. One cluster is generated from a Gaussian distribution with identity covariance matrix $\mathbf{I}$ and contains 200 points. The second cluster is formed by a Gaussian distribution with covariance matrix $0.25 \times \mathbf{I}$ and consists of 100 points. It is shifted from the center of the first cluster by approximately 2.7 units. The third cluster is constructed from a uniform distribution in the interval $(0, 2)$. It consists of 200 points and is shifted from the center of the first cluster by approximately 2.8 units. The obtained data is illustrated in Figure 1(a), where clusters are colored differently. The data set has quite a visible underlying community structure, although cluster membership for some of the points on the border is not clear.

We construct a distance matrix using the Euclidean distance. The distance between two points x and y is calculated as $\|\mathbf{x} - \mathbf{y}\| = \sqrt{\sum_i (x_i - y_i)^2}$, where i

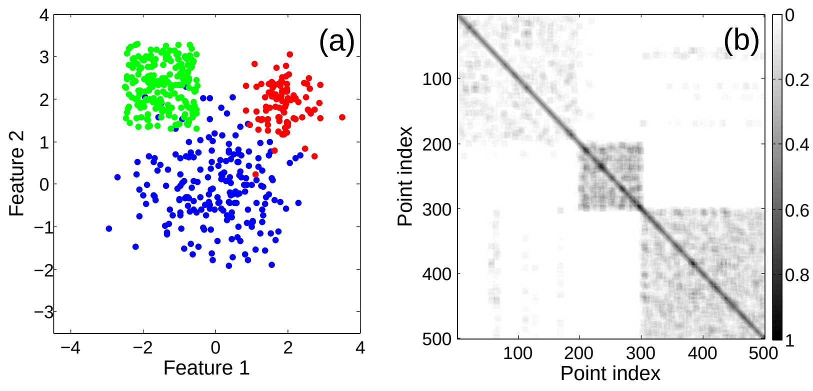


Figure 1: (a) The original data set and true cluster division from illustrative example, (b) similarity matrix corresponding to the data in (a).

denotes a dimension. The similarity matrix shown in Figure 1(b) is formed from the distance matrix using the general form of a Gaussian radial basis function

$$h(x, y) = \exp(-\|x - y\|^2/\sigma^2), \tag{5}$$

where $\sigma > 0$. In this example, considering the units of $x_i - y_i$, we empirically set the parameter σ to 0.2 and δ_d to 0.12. We notice that three standard deviations of $x_i - y_i$, $3\sigma_{x_i - y_i}$, is 0.212 and the choice of δ_d from the interval [0.05, 0.12] brings no change in the clustering result. Therefore, we choose the upper limit of this interval $\delta_d = 0.12$.

We apply the proposed method (EGD) clustering to the obtained similarity matrix. We consider the diffusion depth parameter values $T = \{3\}$, $T = \{8\}$ and their combination $T = \{3, 8\}$. Figure 2(a)-(c) shows clustering results with EGD for the data and parameters described above. Every cluster is marked in its own color. The clusters corresponding to original data clusters are displayed in similar colors. The algorithm with $T = \{3\}$ fails to find the underlying data structure and assigns nearly all points to one cluster (see Figure 2(a)). For $T = \{8\}$, the method successfully determines one cluster marked green (see Figure 2(b)), but shuffles the two remaining clusters. For $T = \{3, 8\}$, the proposed EGD approach discovers all three clusters, except for a few elements

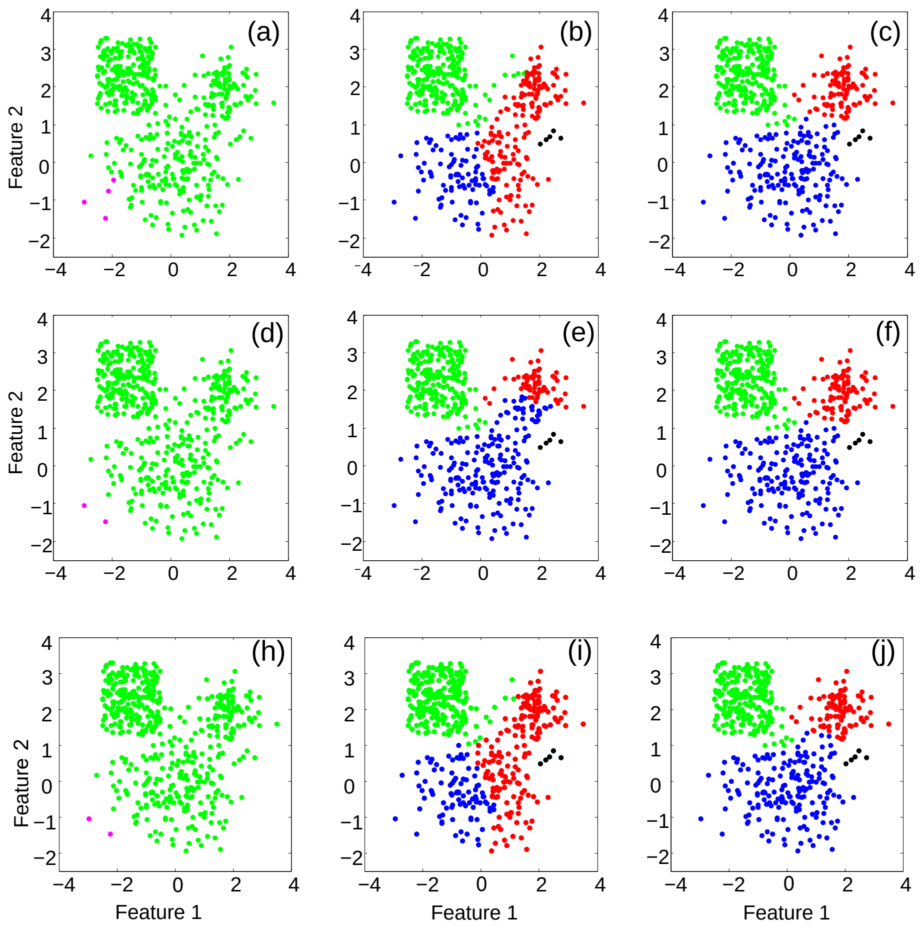


Figure 2: EGD clustering results for illustrative example: (a) $T = \{3\}$, (b) $T = \{8\}$, (c) $T = \{3, 8\}$, (d) $T = \{2\}$, (e) $T = \{7\}$, (f) $T = \{2, 7\}$, (h) $T = \{1\}$, (i) $T = \{9\}$, (j) $T = \{1, 9\}$.

near the border and a small group of points treated as a separate cluster. We note that the border between clusters in this data set is not obvious and the problem of clustering such points belonging to the border will be addressed further in Section 6. Apparently, the collective nature of the proposed method leverages the benefits of runs with a single t (see Figure 2(c)).

Furthermore, using the data set with ground-truth, we show the effectiveness of the EGD algorithm by providing clustering results under various settings of the parameter T. Figures 2(d), (e), and (f) display clustering results for

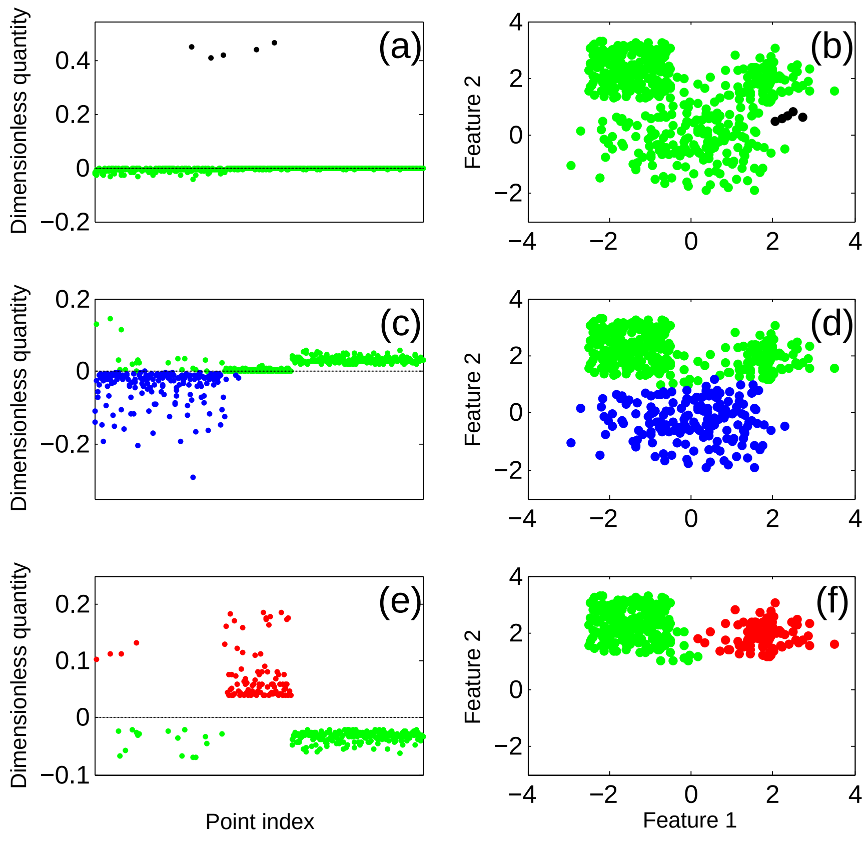


Figure 3: EGD clustering steps in illustrative example for $T = \{3, 8\}$. (a) The first eigenvector of similarity matrix at the first iteration, (b) the first division, (c) the first eigenvector of similarity matrix at the second iteration, (d) the second division, (e) the first eigenvector of similarity matrix at the third iteration, (f) the third division.

$T = \{2\}$, $T = \{7\}$ and their combination $T = \{2, 7\}$, respectively. The algorithm with $T = \{2\}$ assigns nearly all points to one cluster (see Figure 2(d)). For $T = \{7\}$, the method determines two clusters marked green and blue, but mixes points in the third cluster (see Figure 2(e)). For $T = \{2, 7\}$, the EGD algorithm discovers all three clusters, except for a few elements near the border and a small group of points treated as a separate cluster. Clustering results for $T = \{1\}$, $T = \{9\}$ and their combination $T = \{1, 9\}$, displayed in Figures 2(h), (i), and (j), are similar to the ones obtained for $T = \{3\}$, $T = \{8\}$, and $T = \{3, 8\}$, and consistently show the benefit of collective diffusion depths.

To demonstrate the functioning of the ensemble algorithm, we display how the data set is split by the eigenvectors of the ensemble group diffusion matrix G in (4) in every iteration. We consider the case when $T = \{3, 8\}$. The first eigenvector corresponding to the first iteration of the algorithm splits the data set into two clusters, green and black (see Figures 3(a)-(b)). At the second iteration, its own first eigenvector splits the green cluster into two parts, green and blue (see Figures 3(c)-(d)). Last, the first eigenvector corresponding to the third iteration separates the green cluster from the previous step into two subclusters, marked green and red (see Figures 3(e)-(f)). In Figure 3 values of the eigenvectors and the corresponding points on the scatter plots are displayed in the same color. In addition, we add Figures 10 and 11 in Supplementary Materials to show how the data set is split by the eigenvectors of the matrix G in every iteration for $T = \{2, 7\}$ and $T = \{1, 9\}$, respectively. The results demonstrate that the algorithm is stable under various parameter settings and provides reasonable separation into groups. Then, we set $\delta_d = 0$ to promote cluster splits. We run EGD by varying t from small to bigger values to show how cluster structure evolves by changing t values. Figure 4 demonstrates the effect of increasing t on clustering results, evolution from local to global structure.

For comparison, we provide the results of the modularity, spectral, and hierarchical clustering methods used further in this work (see Figure 5). As parameter values for spectral and hierarchical methods we provide true ($k = 3$) and wrong numbers of clusters ($k = 2, 4$). Hierarchical clustering method places all data points mainly in one cluster for all tested parameter values (see Figures 5(a)-(c)). Spectral clustering correctly determines one cluster for $k = 5$ but shuffles points belonging to the other two clusters. However, for $k = 3$, spectral clustering performs relatively well, which is quite natural in that the true number of clusters was provided in this case. The results of the spectral clustering approach can be seen in Figures 5(d)-(f). The modularity approach fails for this data set as it discovers too many clusters (see Figure 5(g)).

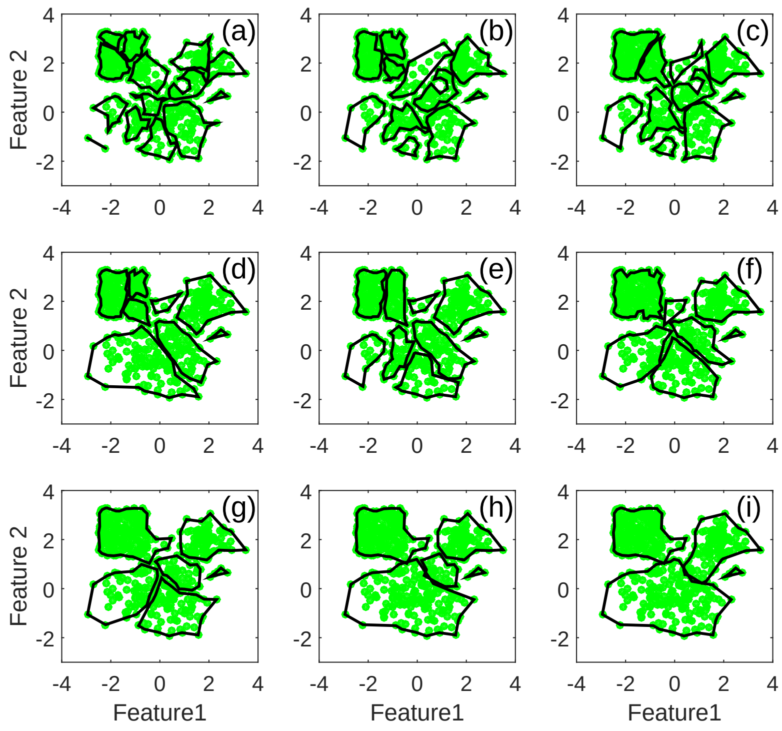


Figure 4: Evolving cluster structure by changing t (number of defined clusters is shown in brackets): (a) $T = \{1\}$ (14), (b) $T = \{5\}$ (11), (c) $T = \{9\}$ (10), (d) $T = \{12\}$ (8), (e) $T = \{15\}$ (9), (f) $T = \{25\}$ (7), (g) $T = \{29\}$ (6), (h) $T = \{35\}$ (5), (i) $T = \{40\}$ (4).

Next, we applied the EGD algorithm to the test cases in which underlying manifold structures exist, as presented in Figure 6. Figures 6(a)-(c) refer to artificial data sets representing classes of different shapes [23]. Figure 6(d) displays the test case which the FLAME approach [24] used as a challenging test case. For computing distance matrices we adopted Manhattan distance measure for the two spirals data set displayed in Figure 6(a) and standardized Euclidean distance measure for the remaining test cases. Similarity matrices were calculated from distance matrices using Gaussian radial basis function as in 5. EGD successfully determined true clusters for the test cases as shown in Figures 6(a)-(c). In particular, the results for the test case (d) are comparable

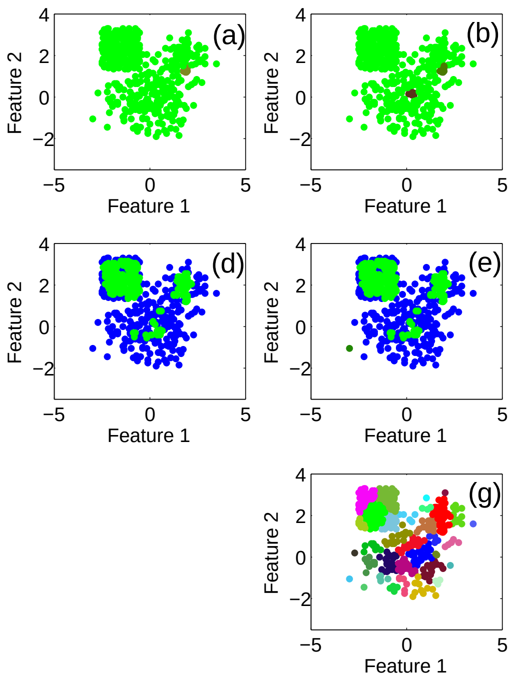


Figure 5: Clustering results for illustrative example: (a) hierarchical clustering ($k = 2$), (b) hierarchical clustering ($k = 3$), (c) hierarchical clustering ($k = 4$), (d) spectral clustering ($k = 2$), (e) spectral clustering ($k = 3$), (f) spectral clustering ($k = 4$), (g) modularity clustering.

to the original method. Note that unlike original method, EGD assigned two outlier points displayed in red to a separate cluster that looks natural in this case and shows a potential ability to reveal a community that is small in scale.

The illustrative and demo examples show that the proposed EGD clustering approach can determine the underlying geometry of the data, even for those data sets where some of the common clustering methods fail. We attribute it to the property of ensemble group diffusion that inherently combines individual outcomes to result in better cluster segmentation.

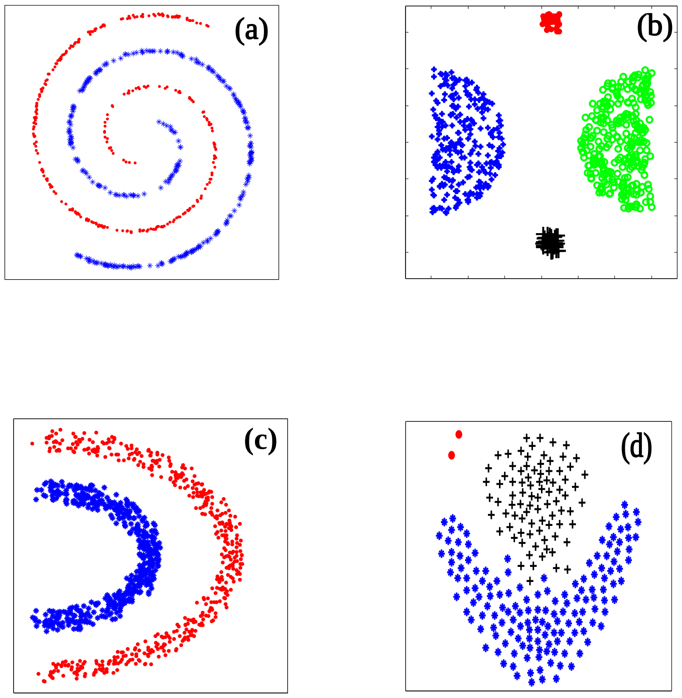


Figure 6: EGD clustering results for the data sets: (a) two spirals, (b) outlier, (c) half kernel, (d) FLAME.

5. Experimental and Empirical Results

5.1. Benchmark Methods

We compare the EGD clustering with other well-known methods frequently used for community structure detection. Among those methods are agglomerative hierarchical clustering, spectral clustering, modularity clustering [9], density-based clustering and a knowledge reuse framework-based (KRF) clustering ensemble approach proposed in [8]. We apply the KRF approach to the results obtained by Metis [25] and graph partitioning (GP) [26] algorithms by varying the number of clusters.

As a density-based method we refer to clustering by ‘fast search and find of density peaks’ (FSFDP) by Rodriguez and Laio [17]. We use a Matlab implementation of FSFDP as was expounded in [27]. In order to define cluster centers the original method adopted a manual setting by supervised analysis of a decision graph that displays local density ρ_i versus distance δ_i from points of higher density for each data point i. Then one finds a rectangular region where both δ_i and ρ_i are high [27]. As the procedure was quite inefficient and deteriorated in the multiple data sets used, we designed a heuristic for automatic selection of cluster centers by binning the data points into k equally spaced intervals along the axes and marking points with maximal δ in each bin as cluster centers.

We employ both simulated and real-life data sets to compare performance of the clustering algorithms, including the proposed EGD method in this paper. In order to evaluate the performance of these algorithms, we need to have ”true clustering labels” for each data set. The simulated data set clearly has one, as we simulate the data set from a predefined correlation matrix structure. For other real-world data sets, we deliberately chose empirical settings where we can define such true clustering for all nodes.

5.2. Performance Evaluation

Given true clustering labels, we measure the performance of the clustering results using two approaches: Rand measure [28] and normalized mutual information (NMI) [8]. The Rand measure is based on the dyad-level accuracy of clustering results, counting the number of pairs in which an algorithm’s clustering result and the true clustering specification agree. In essence, it combines the ratio of correctly clustered pairs (CC) and the ratio of correctly separated pairs (CS). CC and CS quantify the performance of clustering algorithms in terms of what percentage of pairs are correctly clustered or separated given the true clustering results. They represent two extremes of a clustering algorithm’s performance. If an algorithm tends to cluster aggressively by putting too many nodes into the same cluster, it will score high in CC but low in CS, and vice versa. Thus, a desirable clustering algorithm should score high in the Rand measure, balancing CC and CS.

These three measures are computed as follows:

$$CC = \frac{\sum_{i,j} 1_{\{x_i = x_j\}} 1_{\{y_i = y_j\}}}{\sum_{i,j} 1_{\{x_i = x_j\}} 1_{\{y_i = y_j\}} + \sum_{i,j} 1_{\{x_i \neq x_j\}} 1_{\{y_i = y_j\}}},$$

$$CS = \frac{\sum_{i,j} 1_{\{x_i \neq x_j\}} 1_{\{y_i \neq y_j\}}}{\sum_{i,j} 1_{\{x_i = x_j\}} 1_{\{y_i \neq y_j\}} + \sum_{i,j} 1_{\{x_i \neq x_j\}} 1_{\{y_i \neq y_j\}}},$$

$$Rand = \frac{\sum_{i,j} 1_{\{x_i = x_j\}} 1_{\{y_i = y_j\}} + \sum_{i,j} 1_{\{x_i \neq x_j\}} 1_{\{y_i \neq y_j\}}}{\sum_{i \neq j} 1},$$

where $X = \{x_i\}, i = 1, \cdots, n$ is the clustering result under evaluation and $Y = \{y_i\}, i = 1, \cdots, n$ is the true cluster labels. n represents the number of nodes in the graph.

On the other hand, we employ another measure, NMI, to capture similarity between the true and test clustering results in a holistic way. Mutual information is a concept from information theory and increases as two input sequences are similar to each other. The normalized variant that we use in this paper scales it into the range between zero and one. The normalization process is similar to that of the Pearson correlation coefficient. In this case, the information-theoretic entropy serves as a normalization factor. The information entropy measures how random each input sequence is. Following Strehl and Ghosh [8], NMI is calculated as follows:

$$\text{NMI}(X, Y) = \frac{I(X, Y)}{\sqrt{H(X)H(Y)}},$$

where $I(X, Y)$ denotes mutual information between X and Y, and $H(X)$ and $H(Y)$ denote the information entropy of X and Y, respectively.

We show the performance summary of the proposed method and other methods in terms of NMI and RND across 9 data sets in Figure 7. We give detailed description of the data sets and performance comparisons in the following sections.

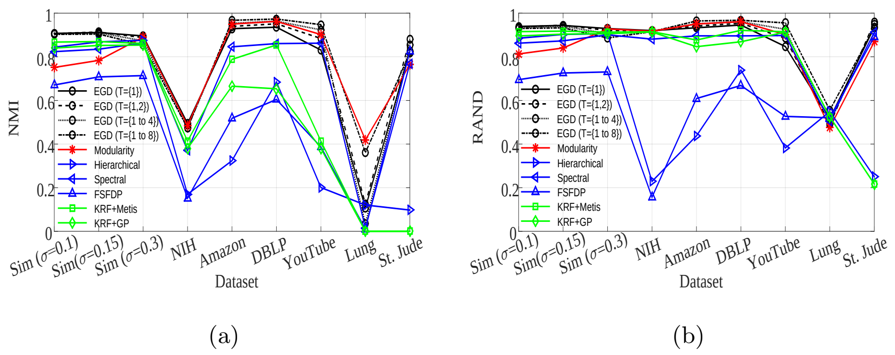


Figure 7: Performance summary of the proposed method and other methods in terms of (a) NMI, (b) RAND.

5.3. Simulation Tests

Our evaluation starts with the tests on synthetically generated data sets. We use a simulated correlation matrix with four evident groups, which represents a graph of 12 nodes. The group structure imposed into the correlation matrix is {1,2}, {3,4,5,6}, {7,8,9}, {10,11,12}. Within-group true correlation coefficients are 0.9, 0.7, 0.6, and 0.8. Nodes in groups 1 and 2 are positively correlated by 0.2 and those in groups 3 and 4 are negatively correlated by -0.4. All other inter-group correlations are set to zero. Each node represents a random variable and the edges of the graph are Pearson sample correlation coefficients $r_{i,j}$ in the range from -1 to 1. The distance between two points i and j is calculated as $\|x_i - x_j\| = 1/(r_{i,j}+1)$. For calculating the similarity matrix W, a general form of the Gaussian radial basis function in Equation (5) was used. More detailed information about the data can be found in [29] and [15].

We evaluate each clustering method as an average over 10,000 replications. For each replication, we build a sample correlation matrix from random realizations from the true correlation matrix. Following Stone and Ayroles [29], we extract nine observations from the true correlation matrix using a multivariate normal distribution. In order to see the effects of the width parameter σ, we vary σ from 0.1 to 0.15 and 0.3.

We tested EGD by varying the diffusion depth parameter set as $T = \{1\}$, {1,2}, {1,2,3,4}, and {1,2,3,4,5,6,7,8} from short-range to long-range scales. Our benchmark methods include agglomerative hierarchical clustering, spectral clustering, modularity clustering, density-based clustering and KRF applied to the results of Metis and GP. We informed benchmark methods other than Modularity about the number of clusters with both the true value ($k = 4$) and misleading values ($k = 3, 5$). Table 4 in Supplementary Materials shows the results of simulation experiments using the nine clustering methods. Boldface values denote the highest number in each column. EGD ouperforms other methods for all values of σ. The method demonstrates more accurate results for relatively low values of $\sigma = 0.1, 0.15$ and low values of the parameter δ_d, with the highest NMI and Rand scores of 0.91 and 0.94, correspondingly for both σ values.

To verify the performance differences between the proposed method and the other methods, we applied post statistical analysis using repeated measures ANOVA. The tests showed that the proposed method outperformed the other methods in the simulation experiments with 95% confidence levels. Please refer to Supplementary Materials (Table 2) to see the p-values of the tests.

5.4. Co-PI Network from the NIH Research Funding Data

To benchmark the performance of the proposed method in the real-world context, we constructed a principal-investigator (PI) network from the funding data of the National Institute of Health (NIH) of the United States. The NIH is a collective body of 27 institutes and centers (ICs), disbursing $25-30B every year for biomedical research. This accounts for a significant portion of the total biomedical research funding of the U.S. and the NIH is the single largest public entity in the picture. As a public agency, the NIH keeps track of detailed funding information for each grant including grant application abstract, activity type, amount of grant, list of co-PIs, and institution of the head PI.

For each grant, one or more researchers are in charge of carrying out the proposed research project. Among them, one person is designated as head PI (or contact PI), who is meant to be the primary corresponding agent for the project. Each project record also contains the head PI's associated institution (university, research institute, or private company) and its address. Although most grants are executed by a single PI, a few projects are led by multiple PIs, in which case the project is run by a head PI and co-PIs.

We focus on the projects having multiple co-PIs to construct the collaboration network of researchers in biomedical research. By counting the number of co-occurrences of PIs, we obtain the weighted undirected graph of the co-PI network.

In order to test the clustering algorithms, we need not only a network but also true labels of the nodes. We prepared the true labels based on the location of the affiliated institution of a PI. In essence, we infer whether PIs are co-located from the collaboration network structure among the co-PIs. Reasoning behind this inference is that researchers in the same geographical region are more likely to collaborate on research project supported by NIH funding.

We first collected all grant data between 2000 and 2012 from NIH's data retrieval interface called ExPORTER. The NIH publishes funding records not only for its 27 ICs but also for some other related agencies. Then, a small number of research grants are awarded to non-US institutions. Last, some large projects are broken into subproject records occasionally. In such cases, we only consider the ultimate parent project record. Since our focus is on the NIH's U.S. funding records, we remove non-U.S. projects from our sample. After filtering out non-NIH, non-US grants, and subproject records, we are left with 707,496 grants. 14,093 projects among them have more than one PI and the number of unique PIs is 11,999. As institution information is only available for the head PI, we removed PIs for which we cannot identify the institution, and 9,769 PIs remain. The collaboration network is extremely sparse because of a myriad of isolated cliques of two or three PIs. We extracted the connected components of size greater than or equal to 10 from the entire landscape. At last, we are left with 993 PIs from 217 institutions in 44 states. Assuming that investigators affiliated with institutions that are geographically close to each other have a higher chance to collaborate as co-PIs, we use the state as a true clustering label for each PI based on the location of their institution.

Table 5 in Supplementary Materials shows the performance comparison with the NIH data set. The EGD clustering with $\delta_d = 0$ and $T = \{1\}$ outperforms all other configurations and algorithms, including modularity clustering. We observe a high Rand measure and low NMI consistently across all methods. Low CC and high CS scores explain why Rand measure is higher than NMI. This implies that the co-PI network is highly fragmented and it has a fairly low chance that two PIs are associated with the institutions in the same state. When algorithms place nodes into the same cluster, it is more likely to be wrong than when they separate out nodes into different clusters. Thus, under this sparse clustering structure, we see that NMI is a more robust measure than the Rand measure, although these two measures were close to each other in the simulation study in the previous section. Last, the hierarchical clustering and FSFDP clustering scores are low in both Rand and NMI, which suggests that the methods clearly failed to correctly identify the latent community structure. Spectral clustering, Metis, GP, and KRF produced better results, but still fell short of the results from modularity clustering and the EGD clustering.

5.5. Social Network Data with Ground Truth Membership Records

Social networks have gained a significant number of users over the past decade. Various social network services operate in the web with a different focus, such as for friendship or professional career networks. This trend led to an explosion of availability of social network data that could be used for academic research. Indeed, various fields such as marketing and psychology have used data sets from real world social network services to address a specific research question. Clustering algorithm development is one of the fields that can immediately benefit from using these social network data sets. One hurdle that prevents one from doing such research is that raw social network data usually does not provide ground truth membership of the nodes. In order to test clustering algorithm performance, we need a true clustering that can be compared against as a benchmark. The notion of ground truth membership depends on how you frame the clustering task. For instance, suppose that we are interested in clustering people in a professional career social network. Depending on our research interest, community structure may be defined by age group or the industry that they are working in. In this case, both age group and industry code can serve as the ground truth membership label for each person in the network.

This section is devoted to the analysis using ground truth networks provided by Yang and Leskovec [30], who constructed a large set of networks with explicit ground truth community structure from a number of different domains. We apply the EGD algorithm to the Stanford Network Analysis Project (SNAP) data consisting of three data sets and compare its performance with the benchmark methods in the same way as before. SNAP data used in this section are in fact undirected graphs with binary edge weights describing three well known real world networks.

The first data set is Amazon's product co-purchasing network. The data set is constructed based on the feature which lists corresponding products (goods) under the tag "Customers Who Bought This Item Also Bought" [30]. The ground truth community is constructed in a way that all its members share a common purpose. Amazon-defined product categories (e.g., electronics, beauty & health, or clothing) serve as the ground-truth communities. The second data set is from DBLP, which is a widely known bibliography repository archiving archiving publication records particularly focusing on the field of computer science. Yang and Leskovec [30] extracted authors' collaboration network from publication data. The authors are connected if they have a joint publication. The publication venues serve as ground-truth communities for the authors. Last, the third data set comes from YouTube, an online video sharing community. It acts as a social network where users can form friendships, create own groups, and join other groups. Such group membership provides ground-truth communities of the users.

In the original data set, Yang and Leskovec [30] provided the list of the top 5,000 largest communities along with network data (i.e., nodes and edges). Since the size of the networks is too large, we first need to reduce the data to check how the clustering algorithms work with smaller data sets. We preprocessed the data in a way that we randomly select top 10 mutually exclusive communities. This guarantees that each node belongs to only a single community, which clears the ambiguity concern of multiple membership. Second, in order to lift the computational burden, we reject a sample containing more than 300 nodes. Last, we randomly choose 100 samples to construct the final set of samples. The network and community statistics averaged over 100 samples are as follows. The average number of nodes and edges in three data sets (Amazon, DBLP, YouTube) are (132, 107, 103) and (387, 314, 171), respectively. All data sets have similar number of nodes, but samples from YouTube are much more sparse networks, as they have about half the number of edges compared to the other two data sets. The average clustering coefficients are (0.69, 0.88, 0.30); DBLP exhibits the highest level of clustering coefficients. In sum, these three sets of samples have different network-level characteristics, which allows us to examine the sensitivity of algorithm performance by comparing the algorithms in these three different settings.

Table 6 in Supplementary Materials shows the performance comparison among the nine clustering algorithms with various configurations. In these results, the EGD method predominantly outperformed all other benchmark algorithms. $\delta_d = 0$ and 0.001 produce the best outcome and a larger set of t led to a better outcome than the smaller set, such as $T = \{1\}$. The overall accuracy scores measured in NMI are in the descending order of DBLP (95.74%), Amazon (94.05%), and YouTube (88.81%), which suggests that higher average clustering coefficient is associated with more accurate clustering outcomes. The statistical testing for the three data sets showed that the proposed method outperformed the other methods in the SNAP experiments with 95% confidence levels except for comparisons with spectral clustering. Notice that for DBLP and YouTube we informed spectral clustering of the correct number of clusters. To see the p-values of the tests, refer to Supplementary Materials (Table 3).

5.6. Amazon Co-purchasing Relationships

Since our proposed method and the dependence clustering works on the adjacency matrix of the network, it is not limited to undirected graphs. Rather, we surmise that our method may work better on directed graphs compared to other favored choices of clustering methods. We construct a set of directed graph samples from Amazon's co-purchasing relationship between products. If product i is purchased together with product j frequently, we denote the relationship as a directed edge from i to j. Note that this relationship is not necessarily reflexive because the absolute level of demand for the two products may starkly differ. This data set is also compiled by SNAP [31]. SNAP collected the co-purchasing network data at multiple points in time. The version we used to construct our samples was collected by SNAP on June 1, 2003. The original population data set consists of 403,394 nodes and 3,387,388 directed edges.

Product co-purchasing networks can serve our purpose of testing the clustering algorithms only if we also have true labels of all nodes. SNAP also provides the metadata for each node such as the product name, product group, and optional subcategories that the product belongs to. SNAP collected the metadata in summer 2006, approximately three years after the co-purchasing network was collected. However, we argue that this gap in data collection time does not affect our samples and results in a significant way because product group and subcategories do not change frequently over time. Most of the nodes fall into one of four product groups: books, music CDs, videos, and DVDs. We decide to select one product group and choose books only for our final samples for three reasons. First, we narrow down to a single product group because we suspect co-purchasing links exist extremely sparsely across different product groups. Second, books are highly standardized products and have a well-defined classification scheme based on the topical subject. Third, books represent more than 70% of the original SNAP data set, thus, choosing books does not undermine the representativeness of our samples. In order to uniquely assign each book to a single true label, we pick the most frequent subject category for a book when the book is tagged with multiple subject areas.

With these settings in place, it is impractical for us to run various clustering algorithms on the full data set. We thus create samples from the full data set with which we test and compare the performance of different clustering algorithms within a reasonable amount of time. The detailed sampling steps are as follows. First, we choose a random book and the randomly chosen book then becomes the only member of the seed set. Second, for each book in the seed set, we look up books frequently co-purchased with the focal book. Third, we add the co-purchased books to the seed set. Fourth, we repeat Step 2 and 3 until the size of the seed set reaches the previously defined threshold. We set the threshold at 100 for our sampling process, so all of our sampled networks have at least 100 nodes. Last, we extract all directed edges between the nodes in the final seed set. In essence, this sampling process generates multiple layers of egonetworks superposed to each other. The adjacency matrix resulting from the sampling process is binary and asymmetric. We create 10 sample networks and labels using this sampling process. The performance metrics are averaged across the 10 samples when reported in the results section.

Each sampled network, on average, contains 165.3 nodes and 774.8 edges, which results in an average network density of 0.03132. 34% of the directed edges are reciprocal, which means that the two nodes have a bidirectional relationship in such cases. The frequently co-purchased relationship is not reflexive by itself, but a significant portion of the relationship in our samples is indeed bidirectional, largely because of our sampling process relying on egonetworks. Still, more than 60% of the relationships are unidirectional. The average number of subject categories for each sampled network is 22.3. One may suspect that most of the nodes in a sampled network belong to a single category also because of our reliance on egonetworks for sampling. However, category membership turns out to be quite evenly distributed. The most frequent category accounts for only 16% of the nodes in a network and the average Herfindahl-Hirschman Index, representing the concentration of proportions, is 0.0873, which is not particularly high.

Table 7 in Supplementary Materials shows the performance comparison between the proposed EGD and modularity clustering methods. After the previous tests we decided to narrow down the comparison analysis to these two methods, as they show the most stable results and both methods do not require a parameter specifying the number of clusters. EGD with low values of t and δ_d outperforms modularity clustering in terms of NMI. Moreover, our method is able to inherently handle the directed relationship by a transition matrix, whereas the modularity approach forces the directed relationship to be symmetric. Thus, compared to modularity clustering, our approach can better handle problems where data sets with inherent directed nature are involved.

5.7. Gene-expression data

In this section we consider two high-dimensional data sets as regards to gene-expression profiles. One is the lung cancer data set [32] including four known classes of speciments: 139 adenocarcinomas (AD), 21 squamous cell carcinomas (SQ), 20 pulmonary carcinoids (COID), and 17 normal lung (NL). The other is St. Jude leukemia data set [33] that contains samples from pediatric acute lymphoblastic leukemia patients. The data set includes six leukemia subtypes: 43 T-lineage (T-ALL), 27 E2A-PBX1, 15 BCR-ABL, 79 TEL-AML1, 20 MLL rearrangements, and 64 hyperdiploid karyotype (i.e., > 50 chromosomes). More detailed description of the data sets can be found in Table 1.

Table 1: Description of the data sets

Data set	# classes	# samples	# features	Distance/Similarity
Lung cancer	4	197	1000	Euclidean/ Gaussian ($\sigma = 0.1$)
St. Jude leukemia	6	248	985	Standardized Euclidean/ $\lvert \mathrm{D} - \max(\mathrm{D}) \rvert^{*}$

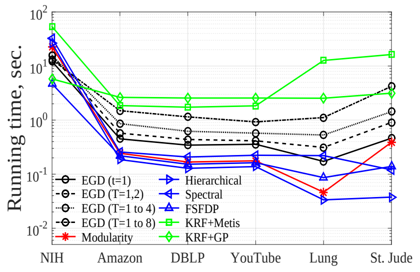
* D refers to distance matrix

For both data sets similarity matrices are obtained from the distance matrices using the measures described in Table 1. For the lung cancer data set the test results show that EGD outperforms other approaches demonstrating the highest NMI and Rand measure scores, as shown in Table 8 (Supplementary Materials). Note that though Modularity clustering method performs better in terms of NMI, it assigns every point to a separate cluster, which is hardly practical. Moreover, hierarchical clustering that provides the highest Rand measure score allocates nearly all the points to a single cluster with an exception of only a few samples. The same happens to FSFDP and KRF. Spectral clustering neither succeeds.

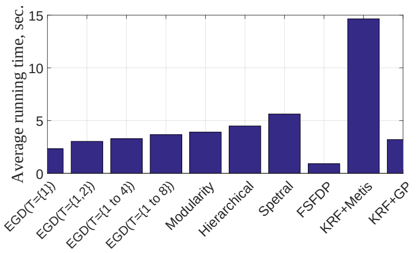
For the St. Jude leukemia data set EGD performs best with the highest NMI value of 0.88, as shown in Table 9 (Supplementary Materials). The highest Rand measure score of 0.96 is provided by spectral clustering. However, compared to spectral clustering the loss of EGD in Rand measure is not significant. Hierarchical and KRF approaches fail by assigning the majority of the points to a single cluster which is verified by significantly low values of CS and high values of CC.

5.8. Running Times

Finally, the running time for the methods and the data sets used in this work are displayed in Figure 8. The experiments were conducted on a system with the following characteristics: 64-bit Windows 10 operating system, Intel(R) Core(TM) i5-3317U CPU 1.70 GHz, 8 GB RAM, and MATLAB (R2014b). MATLAB implementations of the KRF approach, Metis, and GP used in the experiments are available at [34] and [35]. For the methods which require the number of clusters as a parameter, only iterations when the true parameter values are provided were considered during the running-time evaluation. Figure 8(a) displays running times on the logarithmic scale grouped by the data sets and averaged over the data sets, correspondingly. Figure 8(b) shows running times in seconds for each method averaged over the data sets. The proposed EGD demonstrated good performance and proved to be the most efficient for the majority of the data sets among the ensemble methods used for comparison in this work. In addition, we show running times according to the length of input data in Figure 9. The data set consists of randomly generated points in two dimensional space where every point belongs to one of three clusters (n points in a cluster), similarly to the example from Figure 1(a). In the tests, n varies in the interval from 50 to 6000. The experiments were conducted on a system with the following characteristics: 1TB of RAM, 64 CPU cores (8 cores Intel(R) Xeon(R) E7-8837 2.67GHz per CPU), and MATLAB (R2017a). The proposed EGD showed consistent performance among the tested methods. One can see that EGD is comparable to Modularity, outperforming it for large input lengths ($n > 2000$) and KRF+GP for all the values of T.



(a)



(b)

Figure 8: Running times of the methods (a) grouped by the data sets (logarithmic scale), (b) averaged over the data sets.

6. Discussions and Conclusion

Cluster ensemble algorithms recently became popular in the field of data analysis because of the growing capabilities of computing technologies. With careful selection of consensus procedure, they prove to be more accurate compared to individual clustering results. Ensemble strategies benefit from combining individual runs of a component algorithm diversified in a randomized or systematic fashion. Randomized diversity is usually achieved by manipulating data (e.g., bagging and nonparametric bootstrapping), whereas systematic diversity is the result of varying parameters (e.g., parametric bootstrapping). In

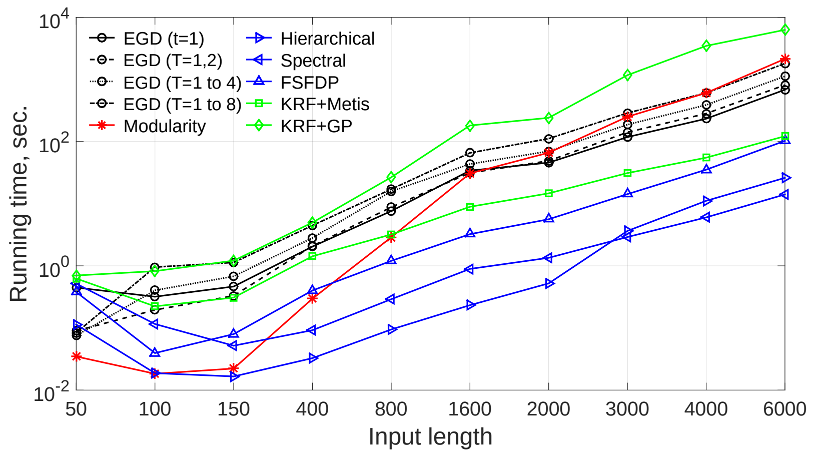


Figure 9: Running times of the methods against data input length (logarithmic scale).

this paper, we employ the latter approach.

Our method, EGD, proposed in this paper is an ensemble approach that maximizes the group diffusion measure. Ensemble diversity is achieved by varying the diffusion depth parameter t. This way, the combined effect of both bias and variance error components reduction is expected. Cluster size can be bounded below by proper settings of the dependence gain parameter δ_d ranging in the interval $[0, 1]$. We suggest using small values of δ_d for the data sets with sparse clustering structure. On the other hand, for the data sets with dense structure and expected unclear boundaries, we recommend using higher values of δ_d. We must note that the cluster size can be bounded below by setting a hard threshold on the minimal number of points in the cluster. This provides direct intuition to setting the threshold according to the expected size of the smallest cluster. The solution with a dependence gain parameter is more flexible and acts as a form of soft threshold. The intuition for setting the dependence gain parameter is, however, less straightforward.

For evaluating the algorithm, we use both simulated and real-world data sets. In the simulated data set, EGD outperforms modularity clustering with respect to Rand and NMI measures. When small values of the width parameter

σ are used, better performance is achieved for larger sets of t values. Similarly, structures of the SNAP and gene-expression data sets are best revealed by EGD with a larger set of t values. This is likely due to the non-uniform density of the underlying data, which requires consideration at different scales. On the other hand, for the NIH data set with sparse clustering structure, EGD outperforms all other methods including modularity clustering for small t values. This implies that this data set has a fine-grained structure which is better discovered by small diffusion depths. In general, this zooming mechanism is ensured by the parameter t in particular. The method is able to determine local clusters with smaller values of t, whereas higher t values allow for determining the global structure. The combination of its values allows for defining clusters more accurately.

Therefore, we conclude that EGD is suitable for solving structure discovery problems for data sets covering a wide spectrum of underlying structural and density properties thanks to flexibility in the tuning of the parameters. Diffusion depth and dependence gain parameters serve the purpose of selectively addressing data analysis at different scales, whereas the ensemble binding provides integration over the scales. The benefit of the proposed method, however, presents the questions of how one should set the parameters and of what are the theoretical interpretations, and of how one can optimally set the gain parameter depending on cluster depths, which will be a future research direction.

Despite accurate results shown in our tests, the proposed EGD algorithm can be improved further by a number of advances. In particular, we will give more detailed attention to the regularization of the algorithm's optimization criterion and the ability to efficiently handle large-sized data in the next phase of our research. Additionally, it is highly demanded to extend the method by adding the ability to determine overlapped clusters, where each instance may belong to multiple clusters simultaneously. This problem is particularly important in the field of community detection in social networks, where multiple membership is a natural attribute.

Acknowledgements

This research was supported by a grant from the HPY Research Foundation funded by Elisa, Finland. This research was also supported by the National Safety Promotion Technology Development Program (201600000002094, Smart crime prevention solution development through machine learning based on Image Big Data), funded by the Ministry of Trade, Industry and Energy(MOTIE). This work was also supported by the Ministry of Education of the Republic of Korea and the National Research Foundation of Korea (NRF-2015S1A5A2A03047963). This research was also supported by the grant (C0514059) funded by Small and Medium Business Administration (SMBA) and AURI in the Republic of Korea.

References

[1] S. Fortunato, C. Castellano, Community structure in graphs, in: Encyclopedia of Complexity and Systems Science, 2009, pp. 1141–1163.

[2] A. K. Jain, M. N. Murty, P. J. Flynn, Data clustering: A review, ACM Computing Surveys 31 (3) (1999) 264–323.

[3] T. J. Hastie, R. J. Tibshirani, J. H. Friedman, The elements of statistical learning : data mining, inference, and prediction, Springer series in statistics, Springer, New York, 2009.

[4] B. Minaei-Bidgoli, A. Topchy, W. Punch, Ensembles of partitions via data resampling, in: Proceedings of International Conference on Information Technology: Coding and Computing, Vol. 2, 2004, pp. 188–192.

[5] R. Dudoit, J. Fridly, Bagging to improve the accuracy of a clustering procedure, Bioinformatics (2003) 1090–1099.

[6] A. L. N. Fred, A. K. Jain, Combining multiple clusterings using evidence accumulation, IEEE Transaction on Pattern Analysis and Machine Intelligence 27 (2005) 835–850.

[7] J. Jia, X. Xiao, B. Liu, Similarity-based spectral clustering ensemble selection, in: Proceedings of 9th International Conference on Fuzzy Systems and Knowledge Discovery, 2012, pp. 1071–1074.

[8] A. Strehl, J. Ghosh, Cluster ensembles-a knowledge reuse framework for combining multiple partitions, Journal of Machine Learning Research 3 (2003) 583–617.

[9] M. E. J. Newman, Modularity and community structure in networks, Proceedings of the National Academy of Sciences 103 (23) (2006) 8577–8582.

[10] M. Girvan, M. E. J. Newman, Community structure in social and biological networks, Proceedings of the National Academy of Sciences 99 (12) (2002) 7821–7826.

[11] D. Lai, H. Lu, C. Nardini, Enhanced modularity-based community detection by random walk network preprocessing, Physical Review E 81 (6).

[12] A. Arenas, A. Fernandez, S. Gomez, Analysis of the structure of complex networks at different resolution levels, New Journal of Physics 10 (5) (2008) 053039.

[13] S. Fortunato, M. Barthélemy, Resolution limit in community detection, Proceedings of the National Academy of Sciences 104 (1) (2007) 36–41.

[14] S. Fortunato, Community Detection in Graphs, Physics reports, Elsevier, 2010.

[15] H. Park, K. Lee, Dependence clustering, a method revealing community structure with group dependence, Knowledge-Based Systems 60 (2014) 58–72.

[16] K. Lee, A. Gray, H. Kim, Dependence maps, a dimensionality reduction with dependence distance for high-dimensional data, Data Mining and Knowledge Discovery 26 (3) (2013) 512–532.

[17] A. Rodriguez, A. Laio, Clustering by fast search and find of density peaks, Science 344 (6191) (2014) 1492–1496.

[18] L. Zheng, T. Li, C. Ding, A framework for hierarchical ensemble clustering, ACM Transactions on Knowledge Discovery from Data 9 (2).

[19] H. Wang, H. Shan, A. Banerjee, Bayesian cluster ensembles, in: Proceedings of the 9th SIAM International Conference on Data Mining, 2009.

[20] K. D. Bollacker, J. Ghosh, Effective supra-classifiers for knowledge base construction, Pattern Recognition Letters 20 (11-13) (1999) 1347–1352.

[21] R. R. Coifman, S. Lafon, A. B. Lee, M. Maggioni, F. Warner, S. Zucker, Geometric diffusions as a tool for harmonic analysis and structure definition of data: Diffusion maps, in: Proceedings of the National Academy of Sciences, 2005, pp. 7426–7431.

[22] J. Ghosh, A. Acharya, Cluster ensembles, Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery 1 (4) (2011) 305–315.

[23] J. Kools, Six functions for generating artificial datasets, `https://se.mathworks.com/matlabcentral/fileexchange/41459-6-functions-for-generating-artificial-datasets` (accessed 13 October 2016).

[24] L. Fu, E. Medico, FLAME, a novel fuzzy clustering method for the analysis of DNA microarray data, BMC Bioinformatics 8 (1).

[25] G. Karypis, V. Kumar, A fast and high quality multilevel scheme for partitioning irregular graphs, SIAM Journal on Scientific Computing 20 (1) (1998) 359–392.

[26] J. P. Hespanha, An efficient matlab algorithm for graph partitioning, Tech. rep. (2004).

[27] Q. Duan, Density-based clustering, `https://se.mathworks.com/matlabcentral/fileexchange/53922-densityclust` (accessed 20 October 2016).

[28] W. Rand, Objective criteria for the evaluation of clustering methods, Journal of the American Statistical Association 66 (336) (1971) 846–850.

[29] E. A. Stone, J. F. Ayroles, Modulated modularity clustering as an exploratory tool for functional genomic inference, PLOS Genetics 5 (5) (2009) 1–13.

[30] J. Yang, J. Leskovec, Defining and evaluating network communities based on ground-truth, in: Proceedings of the ACM SIGKDD Workshop on Mining Data Semantics, MDS '12, ACM, New York, NY, USA, 2012.

[31] J. Leskovec, L. A. Adamic, B. A. Huberman, The dynamics of viral marketing, ACM Transactions on the Web (TWEB) 1 (1) (2007) 5.

[32] A. Bhattacharjee, W. G. Richards, J. Staunton, C. Li, S. Monti, P. Vasa, C. Ladd, J. Beheshti, R. Bueno, M. Gillette, M. Loda, G. Weber, E. J. Mark, E. S. Lander, W. Wong, B. E. Johnson, T. R. Golub, D. J. Sugarbaker, M. Meyerson, Classification of human lung carcinomas by mrna expression profiling reveals distinct adenocarcinoma subclasses, Proceedings of the National Academy of Sciences of the United States of America 98 (24) (2001) 13790–13795.

[33] E.-J. Yeoh, M. E. Ross, S. A. Shurtleff, W. Williams, D. Patel, R. Mahfouz, F. G. Behm, S. C. Raimondi, M. V. Relling, A. Patel, C. Cheng, D. Campana, D. Wilkins, X. Zhou, J. Li, H. Liu, C.-H. Pui, W. E. Evans, C. Naeve, L. Wong, J. R. Downing, Classification, subtype discovery, and prediction of outcome in pediatric acute lymphoblastic leukemia by gene expression profiling, Cancer Cell 1 (2) (2002) 133 – 143.

[34] A. Strehl, Clusterpack matlaboctave toolbox, Available at `http://strehl.com/download/ClusterPackV20.zip` (accessed 20 October 2016).

[35] J. P. Hespanha, `grPartition` a MATLAB function for graph partitioning, Available at `http://www.ece.ucsb.edu/~hespanha` (accessed 20 October 2016).

Supplementary Materials

Table 2: Results (p-values) of repeated measure ANOVA tests for Simulation data (the null hypothesis is $\mu_1 = \mu_2$, in which μ_i means the performance of method i, and the alternative is $\mu_1 \neq \mu_2$)

Metod1	Method2	NMI	Rand
Simulation data, $\sigma = 0.1$			
EGD ($\delta_d = 0.01$,$T = \{1\}$)	Modularity	<0.001	<0.001
	Hierarchical (k=5)	<0.001	<0.001
	Spectral (k=5)	<0.001	<0.001
	FSFDP (k=5)	<0.001	<0.001
	KRF+Metis	<0.001	<0.001
	KRF+GP	<0.001	<0.001
Simulation data, $\sigma = 0.15$			
EGD ($\delta_d = 0$,$T = \{1, 2\}$)	Modularity	<0.001	<0.001
	Hierarchical (k=5)	<0.001	<0.001
	Spectral (k=5)	<0.001	<0.001
	FSFDP (k=5)	<0.001	<0.001
	KRF+Metis	<0.001	<0.001
	KRF+GP	<0.001	<0.001
Simulation data, $\sigma = 0.3$			
EGD ($\delta_d = 0$,$T = \{1\}$)	Modularity	<0.001	<0.001
	Hierarchical (k=4)	<0.001	<0.001
	Spectral (k=4)	<0.001	<0.001
	FSFDP (k=5)	<0.001	<0.001
	KRF+Metis	<0.001	<0.001
	KRF+GP	<0.001	<0.001

Table 3: Results (p-values) of repeated measure ANOVA tests for SNAP data (the null hypothesis is $\mu_1 = \mu_2$, in which μ_i means the performance of method i, and the alternative is $\mu_1 \neq \mu_2$)

Metod1	Method2	NMI	Rand
Amazon			
EGD ($\delta_d = 0$,T={1 to 8})	Modularity	<0.001	<0.001
	Hierarchical (k=11)	<0.001	<0.001
	Spectral (k=9)	<0.001	0.131
	FSFDP (k=11)	<0.001	<0.001
	KRF+Metis	<0.001	<0.001
	KRF+GP	<0.001	<0.001
DBLP			
EGD ($\delta_d = 0.001$,T={1 to 8})	Modularity	<0.001	<0.001
	Hierarchical (k=11)	<0.001	<0.001
	Spectral (k=10)	0.300	0.043
	FSFDP (k=11)	<0.001	<0.001
	KRF+Metis	<0.001	<0.001
	KRF+GP	<0.001	<0.001
YouTube			
EGD ($\delta_d = 0$,T={1 to 8})	Modularity	<0.001	<0.001
	Hierarchical (k=11)	<0.001	<0.001
	Spectral (k=10)	0.200	0.977
	FSFDP (k=11)	<0.001	<0.001
	KRF+Metis	<0.001	<0.001
	KRF+GP	<0.001	<0.001

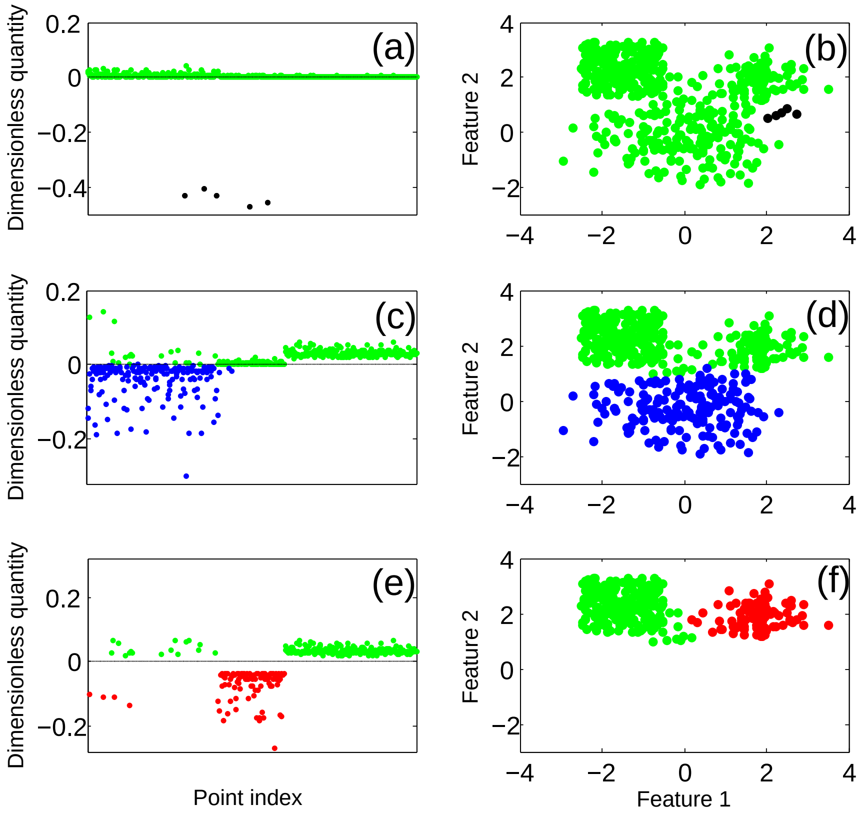


Figure 10: EGD clustering steps in illustrative example for $T = \{2, 7\}$. (a) The first eigenvector of similarity matrix at the first iteration, (b) the first division, (c) the first eigenvector of similarity matrix at the second iteration, (d) the second division, (e) the first eigenvector of similarity matrix at the third iteration, (f) the third division.

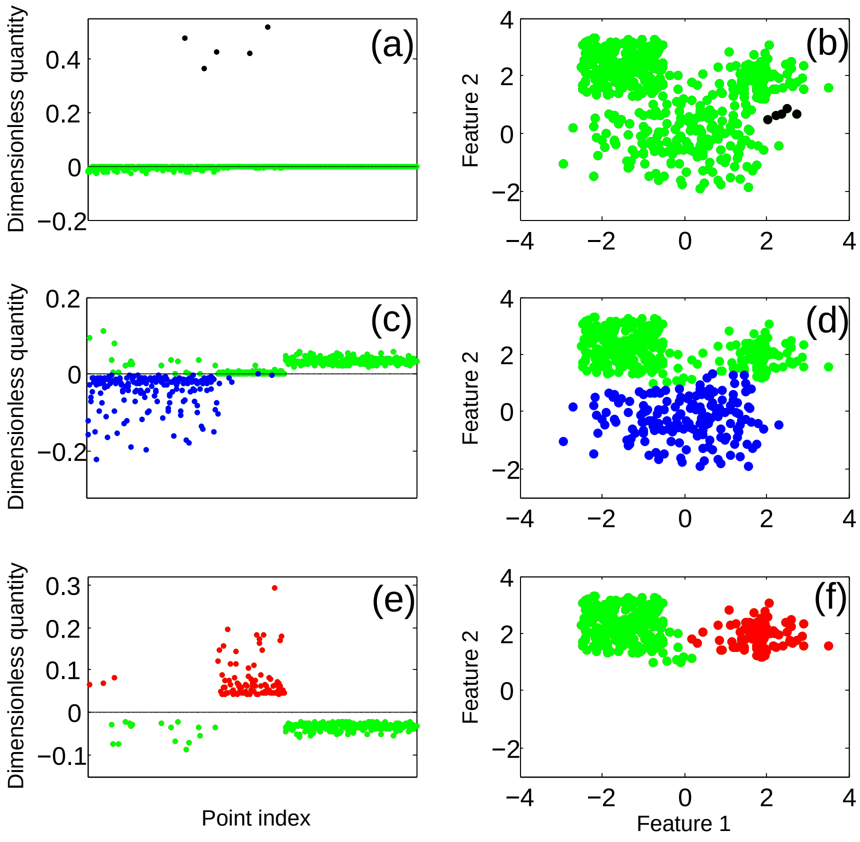


Figure 11: EGD clustering steps in illustrative example for $T = \{1, 9\}$. (a) The first eigenvector of similarity matrix at the first iteration, (b) the first division, (c) the first eigenvector of similarity matrix at the second iteration, (d) the second division, (e) the first eigenvector of similarity matrix at the third iteration, (f) the third division.

Table 4: Simulation

			$\sigma = .1$				$\sigma = .15$				$\sigma = .3$			
Method	δ_d	Parameter	CC	CS	Rand	NMI	CC	CS	Rand	NMI	CC	CS	Rand	NMI
EGD	0	$T = \{1\}$	0.8220	0.9680	0.9392	0.9073	0.8107	0.9764	0.9438	0.9115	0.8687	0.9445	**0.9295**	**0.8956**
		$T = \{1, 2\}$	0.8434	0.9598	0.9369	0.9064	0.8433	0.9695	**0.9446**	**0.9135**	0.8826	0.9305	0.9210	0.8880
		$T = \{1 \text{ to } 4\}$	0.8623	0.9509	0.9335	0.9050	0.8696	0.9578	0.9405	0.9102	0.8956	0.9083	0.9058	0.8734
		$T = \{1 \text{ to } 8\}$	0.8751	0.9432	0.9298	0.9030	0.8902	0.9434	0.9329	0.9043	0.9031	0.8824	0.8865	0.8547
	0.01	$T = \{1\}$	0.8310	0.9661	**0.9395**	**0.9079**	0.8238	0.9732	0.9438	0.9112	0.8729	0.9392	0.9261	0.8915
		$T = \{1, 2\}$	0.8511	0.9578	0.9367	0.9067	0.8520	0.9657	0.9433	0.9119	0.8863	0.9253	0.9176	0.8841
		$T = \{1 \text{ to } 4\}$	0.8677	0.9493	0.9332	0.9049	0.8753	0.9544	0.9388	0.9085	0.8984	0.9024	0.9016	0.8687
		$T = \{1 \text{ to } 8\}$	0.8807	0.9412	0.9293	0.9027	0.8936	0.9404	0.9312	0.9025	0.9067	0.8756	0.8817	0.8495
	0.2	$T = \{1\}$	0.9327	0.8520	0.8679	0.8382	0.9285	0.8393	0.8568	0.8261	0.9199	0.8179	0.8379	0.7984
		$T = \{1, 2\}$	0.9317	0.8565	0.8713	0.8421	0.9296	0.8493	0.8651	0.8352	0.9223	0.8158	0.8368	0.7989
		$T = \{1 \text{ to } 4\}$	0.9338	0.8529	0.8689	0.8403	0.9334	0.8495	0.8660	0.8374	0.9278	0.7986	0.8240	0.7887
		$T = \{1 \text{ to } 8\}$	0.9362	0.8482	0.8655	0.8377	0.9371	0.8452	0.8633	0.8358	0.9339	0.7743	0.8057	0.7745
Modularity			0.0516	0.9999	0.8131	0.7512	0.1925	0.9993	0.8404	0.7835	0.6826	0.9881	0.9279	0.8902
Hierarchical		$k = 3$	0.8861	0.7338	0.7638	0.7433	0.9236	0.7611	0.7931	0.7805	0.9487	0.7879	0.8196	0.8075
		$k = 4$	0.8131	0.9028	0.8851	0.8441	0.8484	0.9158	0.9026	0.8683	0.8690	0.9153	0.9062	0.8757
		$k = 5$	0.7156	0.9576	0.9100	0.8654	0.7356	0.9582	0.9143	0.8722	0.7408	0.9568	0.9142	0.8718
Spectral		$k = 3$	0.9148	0.7404	0.7747	0.7523	0.9175	0.7511	0.7839	0.7614	0.9271	0.7875	0.8150	0.7905
		$k = 4$	0.8361	0.8689	0.8624	0.8236	0.8406	0.8811	0.8731	0.8348	0.8530	0.9124	0.9007	0.8632
		$k = 5$	0.7276	0.9309	0.8908	0.8421	0.7308	0.9407	0.8993	0.8515	0.7271	0.9550	0.9101	0.8624

Continued on next page

Table 4: Simulation – continued from previous page

			$\sigma = .1$				$\sigma = .15$				$\sigma = .3$			
Method	δ_d	Parameter	CC	CS	Rand	NMI	CC	CS	Rand	NMI	CC	CS	Rand	NMI
FSFDP		$k = 3$	0.8750	0.6123	0.6640	0.6482	0.8935	0.6454	0.6942	0.6835	0.9281	0.6488	0.7038	0.6923
		$k = 4$	0.8315	0.6618	0.6952	0.6707	0.8493	0.6954	0.7257	0.7078	0.8984	0.6894	0.7306	0.7138
		$k = 5$	0.7958	0.6935	0.7137	0.6828	0.8044	0.7249	0.7405	0.7167	0.8703	0.7130	0.7440	0.7235
Metis		$k = 3$	0.8467	0.8491	0.8486	0.7748	0.8476	0.8511	0.8504	0.7777	0.8444	0.8506	0.8494	0.7739
		$k = 4$	0.8148	0.9532	0.9260	0.8752	0.8085	0.9554	0.9264	0.8763	0.7946	0.9528	0.9216	0.8688
		$k = 5$	0.6647	0.9754	0.9142	0.8643	0.6680	0.9782	0.9171	0.8685	0.6566	0.9744	0.9118	0.8584
KRF+Metis			0.6781	0.9738	0.9155	0.8675	0.6774	0.9767	0.9177	0.8697	0.6675	0.9733	0.9131	0.8612
GP		$k = 3$	0.9115	0.7460	0.7786	0.7532	0.9125	0.7586	0.7889	0.7627	0.9139	0.7808	0.8070	0.7778
		$k = 4$	0.8269	0.8726	0.8636	0.8217	0.8268	0.8855	0.8739	0.8319	0.8296	0.8996	0.8858	0.8428
		$k = 5$	0.7174	0.9362	0.8931	0.8422	0.7153	0.9450	0.8997	0.8491	0.7104	0.9487	0.9017	0.8495
KRF+GP			0.6948	0.9446	0.8954	0.8402	0.7013	0.9536	0.9039	0.8520	0.6952	0.9581	0.9063	0.8537

Table 5: NIH

Method	δ_d	Parameter	CC	CS	Rand	NMI
EGD	0	$T = \{1\}$	0.3422	0.9306	**0.9201**	**0.4946**
		$T = \{1, 2\}$	0.3253	0.9305	0.9192	0.4868
		$T = \{1 \text{ to } 4\}$	0.2907	0.9311	0.9156	0.4743
		$T = \{1 \text{ to } 8\}$	0.2915	0.9312	0.9155	0.4726
	0.0001	$T = \{1\}$	0.3313	0.9306	0.9195	0.4881
		$T = \{1, 2\}$	0.3253	0.9305	0.9192	0.4868
		$T = \{1 \text{ to } 4\}$	0.2907	0.9311	0.9156	0.4743
		$T = \{1 \text{ to } 8\}$	0.2915	0.9312	0.9155	0.4726
Modularity			0.3048	0.9308	0.9173	0.4843
Hierarchical		$k = 43$	0.0753	0.9303	0.2287	0.1690
		$k = 44$	0.0753	0.9303	0.2287	0.1699
		$k = 45$	0.0754	0.9307	0.2305	0.1717
Spectral		$k = 43$	0.1322	0.9299	0.8767	0.3571
		$k = 44$	0.1424	0.9303	0.8810	0.3705
		$k = 45$	0.1309	0.9295	0.8797	0.3523
FSFDP		$k = 43$	0.0777	0.9612	0.1559	0.1686
		$k = 44$	0.0771	0.9547	0.1565	0.1519
		$k = 45$	0.0773	0.9504	0.1745	0.1612
Metis		$k = 43$	0.2436	0.9296	0.9142	0.4093
		$k = 44$	0.2703	0.9300	0.9157	0.4296
		$k = 45$	0.2533	0.9296	0.9152	0.4189
KRF+Metis			0.2505	0.9295	0.9150	0.4134
GP		$k = 43$	0.1568	0.9297	0.8938	0.3676
		$k = 44$	0.1430	0.9294	0.8893	0.3613
		$k = 45$	0.1694	0.9299	0.8975	0.3788
KRF+GP			0.2150	0.9287	0.9135	0.3800

Table 6: SNAP

			Amazon				DBLP				YouTube			
Method	δ_d	Parameter	CC	CS	Rand	NMI	CC	CS	Rand	NMI	CC	CS	Rand	NMI
EGD	0	$T = \{1\}$	0.7056	1	0.9331	0.9290	0.7269	0.9998	0.9456	0.9357	0.4743	0.9978	0.8470	0.8311
		$T = \{1, 2\}$	0.7479	1	0.9410	0.9394	0.7959	0.9999	0.9537	0.9513	0.6607	0.9960	0.8851	0.8841
		$T = \{1 \text{ to } 4\}$	0.8042	1	0.9516	0.9535	0.8518	0.9999	0.9609	0.9635	0.8117	0.9942	0.9261	0.9235
		$T = \{1 \text{ to } 8\}$	0.8661	1	**0.9643**	**0.9673**	0.8935	1	0.9674	0.9732	0.9031	0.9886	0.9554	**0.9473**
	0.001	$T = \{1\}$	0.7135	1	0.9346	0.9315	0.7447	0.9998	0.9481	0.9408	0.4949	0.9963	0.8511	0.8346
		$T = \{1, 2\}$	0.7603	0.9999	0.9432	0.9425	0.8093	0.9998	0.9554	0.9545	0.6706	0.9955	0.8878	0.8845
		$T = \{1 \text{ to } 4\}$	0.8122	0.9999	0.9528	0.9550	0.8565	0.9999	0.9617	0.9648	0.8150	0.9935	0.9267	0.9204
		$T = \{1 \text{ to } 8\}$	0.8661	0.9999	**0.9643**	0.9667	0.8957	1	0.9679	**0.9740**	0.9062	0.9872	**0.9557**	0.9446
	0.01	$T = \{1\}$	0.7884	0.9853	0.9366	0.9117	0.8441	0.9946	0.9569	0.9497	0.6091	0.9787	0.8648	0.8229
		$T = \{1, 2\}$	0.8084	0.9877	0.9419	0.9200	0.8663	0.9949	0.9601	0.9547	0.7580	0.9770	0.9004	0.8638
		$T = \{1 \text{ to } 4\}$	0.8543	0.9865	0.9507	0.9274	0.8984	0.9949	0.9650	0.9608	0.8538	0.9797	0.9308	0.8925
		$T = \{1 \text{ to } 8\}$	.8846	0.9899	0.9599	0.9424	0.9122	0.9958	**0.9682**	0.9658	0.9252	0.9729	0.9508	0.9083
Modularity			0.7916	1	0.9504	0.9501	0.8455	0.9999	0.9610	0.9624	0.7238	0.9937	0.9017	0.9019
Hierarchical		$k = 9$	0.7843	0.3300	0.4209	0.3085	0.7966	0.6917	0.7157	0.6687	0.7609	0.2146	0.3662	0.1839
		$k = 10$	0.7600	0.3553	0.4376	0.3239	0.7728	0.7227	0.7396	0.6847	0.744	0.2396	0.3822	0.2006
		$k = 11$	0.7306	0.3881	0.4598	0.3444	0.7457	0.7537	0.7627	0.6996	0.7269	0.2666	0.4001	0.2182
Spectral		$k = 9$	0.7907	0.9282	0.8951	0.8473	0.8234	0.9041	0.8800	0.8385	0.7029	0.9336	0.8621	0.8063
		$k = 10$	0.7243	0.9449	0.8960	0.8458	0.8088	0.9279	0.8959	0.8606	0.7102	0.9767	0.8977	0.8625
		$k = 11$	0.6320	0.9479	0.8837	0.8271	0.7175	0.9333	0.8918	0.8449	0.6043	0.9775	0.8718	0.833

Continued on next page

Table 6: SNAP – continued from previous page

			Amazon				DBLP				YouTube			
Method	δ_d	Parameter	CC	CS	Rand	NMI	CC	CS	Rand	NMI	CC	CS	Rand	NMI
FSFDP		$k = 9$	0.6960	0.5464	0.5781	0.4942	0.7418	0.6198	0.6398	0.5845	0.6863	0.4495	0.5133	0.3788
		$k = 10$	0.6531	0.5977	0.6087	0.5182	0.7203	0.6572	0.6686	0.6044	0.6616	0.4728	0.5268	0.3866
		$k = 11$	0.6454	0.6201	0.6280	0.5341	0.7020	0.6751	0.6818	0.6115	0.6398	0.5103	0.5480	0.4086
KRF+Metis		$k = 9$	0.5117	0.9716	0.8803	0.7931	0.6836	0.9707	0.9174	0.8422	0.2436	0.9296	0.9142	0.4093
		$k = 10$	0.4744	0.9779	0.8793	0.7953	0.6454	0.9781	0.9192	0.8482	0.2703	0.9300	0.9157	0.4296
		$k = 11$	0.4396	0.9822	0.8773	0.7948	0.6221	0.9852	0.9225	0.8619	0.2533	0.9296	0.9152	0.4189
			0.4442	0.9808	0.8769	0.7883	0.6261	0.9835	0.9213	0.8543	0.2505	0.9295	0.9150	0.4134
KRF+GP		$k = 9$	0.4818	0.8880	0.8095	0.6497	0.5280	0.8435	0.7908	0.6252	0.1568	0.9297	0.8938	0.3676
		$k = 10$	0.4500	0.9086	0.8203	0.6619	0.4882	0.8547	0.7964	0.6169	0.1430	0.9294	0.8893	0.3613
		$k = 11$	0.4137	0.9229	0.8263	0.6667	0.4800	0.8820	0.8182	0.6474	0.1694	0.9299	0.8975	0.3788
			0.3492	0.9605	0.8464	0.6652	0.3922	0.9527	0.8689	0.6534	0.2150	0.9287	0.9135	0.3800

Table 7: Amazon co-purchasing relationships

Method	δ_d	Parameter	CC	CS	Rand	NMI
EGD	0	$T = \{1\}$	0.0772	0.9184	0.8317	**0.2971**
		$T = \{1, 2\}$	0.0813	0.9192	0.7697	0.2563
		$T = \{1 \text{ to } 4\}$	0.0810	0.9185	0.8190	0.2746
		$T = \{1 \text{ to } 8\}$	0.0808	0.9188	0.7680	0.2326
	0.001	$T = \{1\}$	0.0800	0.9185	0.8179	0.2564
		$T = \{1, 2\}$	0.0816	0.9192	0.7616	0.2253
		$T = \{1 \text{ to } 4\}$	0.0804	0.9184	0.8116	0.2381
		$T = \{1 \text{ to } 8\}$	0.0804	0.9187	0.7641	0.2065
Modularity			0.0778	0.9182	**0.8441**	0.2872

Table 8: Lung cancer

Method	δ_d	Parameter	CC	CS	Rand	NMI
EGD	0.09	$T = \{1\}$	0.2928	0.7630	0.5164	0.1069
		$T = \{1, 2\}$	0.2172	0.7902	0.4896	0.1259
		$T = \{1 \text{ to } 4\}$	0.2643	0.7477	0.4941	0.0345
		$T = \{1 \text{ to } 8\}$	0.3346	0.7997	**0.5557**	**0.3609**
Modularity			0	1	0.4754	**0.4192**
Hierarchical		$k = 3$	0.9963	0.0386	0.5410	0.0974
		$k = 4$	0.9947	0.0578	0.5493	0.1206
		$k = 5$	0.9931	0.0771	**0.5576**	0.1408
Spectral		$k = 3$	0.4330	0.6147	0.5194	0.0188
		$k = 4$	0.4162	0.6056	0.5063	0.0153
		$k = 5$	0.5231	0.4420	0.4845	0.0508
FSFDP		$k = 3$	0.9864	0.0063	0.5204	0.0103
		$k = 4$	0.9864	0.0063	0.5204	0.0103
		$k = 5$	0.9864	0.0063	0.5204	0.0103
Metis		$k = 3$	0.3281	0.6680	0.4897	0.0061
		$k = 4$	0.2435	0.7508	0.4847	0.0083
		$k = 5$	0.1952	0.8032	0.4843	0.0160
KRF+Metis			1	0	0.5246	0
GP		$k = 3$	0.3292	0.6675	0.4901	0.0137
		$k = 4$	0.2512	0.7543	0.4904	0.0246
		$k = 5$	0.1986	0.7994	0.4843	0.0493
KRF+GP			1	0	0.5246	0

Table 9: St. Jude leukemia

Method	δ_d	Parameter	CC	CS	Rand	NMI
EGD	0.06	$T = \{1\}$	0.9469	0.9319	0.9351	0.8262
		$T = \{1, 2\}$	0.9469	0.9623	0.9589	**0.8800**
		$T = \{1 \text{ to } 4\}$	0.8882	0.9648	0.9482	0.8539
		$T = \{1 \text{ to } 8\}$	0.7514	0.9827	0.9325	0.8172
Modularity			0.9686	0.8411	0.8688	0.7640
Hierarchical		$k = 3$	0.9512	0.0475	0.2436	0.0845
		$k = 4$	0.9485	0.0568	0.2503	0.0974
		$k = 5$	0.9482	0.0568	0.2503	0.0953
Spectral		$k = 3$	0.9418	0.9646	**0.9596**	0.8663
		$k = 4$	0.7265	0.9640	0.9125	0.7675
		$k = 5$	0.8488	0.9879	0.9578	0.8640
FSFDP		$k = 3$	0.9449	0.7988	0.8305	0.7428
		$k = 4$	0.9233	0.8833	0.8920	0.8253
		$k = 5$	0.9175	0.8761	0.8851	0.8038
Metis		$k = 3$	0.9497	0.3772	0.5015	0.4488
		$k = 4$	0.7265	0.7412	0.7380	0.4956
		$k = 5$	0.6348	0.6824	0.6721	0.4129
KRF+Metis			1	0	0.2170	0
GP		$k = 3$	0.6333	0.9242	0.8611	0.6845
		$k = 4$	0.5557	0.9454	0.8608	0.6704
		$k = 5$	0.4956	0.9593	0.8587	0.6764
KRF+GP			1	0	0.2170	0