A Mathematical Theory for Clustering in Metric Spaces

Cheng-Shang Chang, *Fellow, IEEE,* Wanjiun Liao, *Fellow, IEEE,* Yu-Sheng Chen, and Li-Heng Liou

***Abstract*—Clustering is one of the most fundamental problems in data analysis and it has been studied extensively in the literature. Though many clustering algorithms have been proposed, clustering theories that justify the use of these clustering algorithms are still unsatisfactory. In particular, one of the fundamental challenges is to address the following question:**

What is a cluster in a set of data points?

In this paper, we make an attempt to address such a question by considering a set of data points associated with a distance measure (metric). We first propose a new *cohesion* measure in terms of the distance measure. Using the cohesion measure, we define a cluster as a set of points that are cohesive to themselves. For such a definition, we show there are various equivalent statements that have intuitive explanations. We then consider the second question:

How do we find clusters and good partitions of clusters under such a definition?

For such a question, we propose a hierarchical agglomerative algorithm and a partitional algorithm. Unlike standard hierarchical agglomerative algorithms, our hierarchical agglomerative algorithm has a specific stopping criterion and it stops with a partition of clusters. Our partitional algorithm, called the K-sets algorithm in the paper, appears to be a new iterative algorithm. Unlike the Lloyd iteration that needs two-step minimization, our K-sets algorithm only takes one-step minimization.

One of the most interesting findings of our paper is the duality result between a distance measure and a cohesion measure. Such a duality result leads to a dual K-sets algorithm for clustering a set of data points with a cohesion measure. The dual K-sets algorithm converges in the same way as a sequential version of the classical kernel K-means algorithm. The key difference is that a cohesion measure does not need to be positive semi-definite.

***Index Terms*—Clustering, hierarchical algorithms, partitional algorithms, convergence, K-sets, duality**

I. Introduction

Clustering is one of the most fundamental problems in data analysis and it has a lot of applications in various fields, including Internet search for information retrieval, social network analysis for community detection, and computation biology for clustering protein sequences. The problem of clustering has been studied extensively in the literature (see e.g., the books [1], [2] and the historical review papers [3], [4]). For a clustering problem, there is a set of data points (or objects) and a similarity (or dissimilarity) measure that measures how similar two data points are. The aim of a clustering algorithm is to cluster these data points so that data points within the same cluster are similar to each other and data points in different clusters are dissimilar.

As stated in [4], clustering algorithms can be divided into two groups: *hierarchical* and *partitional*. Hierarchical algorithms can further be divided into two subgroups: *agglomerative* and *divisive*. Agglomerative hierarchical algorithms, starting from each data point as a sole cluster, recursively merge two *similar* clusters into a new cluster. On the other hand, divisive hierarchical algorithms, starting from the whole set as a single cluster, recursively divide a cluster into two *dissimilar* clusters. As such, there is a hierarchical structure of clusters from either a hierarchical agglomerative clustering algorithm or a hierarchical divisive clustering algorithm.

Partitional algorithms do not have a hierarchical structure of clusters. Instead, they find all the clusters as a partition of the data points. The K-means algorithm is perhaps the simplest and the most widely used partitional algorithm for data points in a Euclidean space, where the Euclidean distance serves as the natural dissimilarity measure. The K-means algorithm starts from an initial partition of the data points into K clusters. It then repeatedly carries out the Lloyd iteration [5] that consists of the following two steps: (i) generate a new partition by assigning each data point to the closest cluster center, and (ii) compute the new cluster centers. The Lloyd iteration is known to reduce the sum of squared distance of each data point to its cluster center in each iteration and thus the K-means algorithm converges to a local minimum. The new cluster centers can be easily found if the data points are in a Euclidean space (or an inner product space). However, it is in general much more difficult to find the representative points for clusters, called medoids, if data points are in a non-Euclidean space. The refined K-means algorithms are commonly referred as the K-medoids algorithm (see e.g., [6], [1], [7], [8]). As the K-means algorithm (or the K-medoids algorithm) converges to a local optimum, it is quite sensitive to the initial choice of the partition. There are some recent works that provide various methods for selecting the initial partition that might lead to performance guarantees [9], [10], [11], [12], [13]. Instead of using the Lloyd iteration to minimize the sum of squared distance of each data point to its cluster center, one can also formulate a clustering problem as an optimization problem with respect to a certain objective function and then solve the optimization problem by other methods. This then leads to kernel and spectral clustering methods (see e.g., [14], [15], [16], [17], [18] and [19], [20] for reviews of the papers in this area). Solving the optimization problems formulated from

C. S. Chang and L. H. Liou are with the Institute of Communications Engineering, National Tsing Hua University, Hsinchu 300, Taiwan, R.O.C. email: cschang@ee.nthu.edu.tw, dacapo1142@gmail.com

W. Liao and Y.-S. Chen are with Department of Electrical Engineering, National Taiwan University, Taipei, Taiwan, R.O.C. email: {wjliao,r01921042}@ntu.edu.tw.

the clustering problems are in general NP-hard and one has to resort to approximation algorithms [21]. In [21], Balcan et al. introduced the concept of approximation stability that assumes all the partitions (clusterings) that have the objective values close to the optimum ones are close to the target partition. Under such an assumption, they proposed efficient algorithms for clustering large data sets.

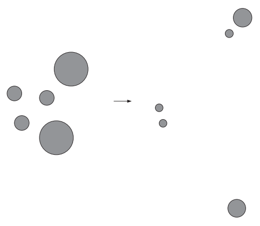

Fig. 1. A consistent change of 5 clusters.

Though there are already many clustering algorithms proposed in the literature, clustering theories that justify the use of these clustering algorithms are still unsatisfactory. As pointed out in [22], there are three commonly used approaches for developing a clustering theory: (i) an axiomatic approach that outlines a list of axioms for a clustering function (see e.g., [23], [24], [25], [26], [27], [28]), (ii) an objective-based approach that provides a specific objective for a clustering function to optimize (see e.g., [29], [21], and (iii) a definition-based approach that specifies the definition of clusters (see e.g, [30], [31], [32]). In [26], Kleinberg adopted an axiomatic approach and showed an impossibility theorem for finding a clustering function that satisfies the following three axioms:

(i) Scale invariance: if we scale the dissimilarity measure by a constant factor, then the clustering function still outputs the same partition of clusters.

(ii) Richness: for any specific partition of the data points, there exists a dissimilarity measure such that the clustering function outputs that partition.

(iii) Consistency: for a partition from the clustering function with respect to a specific dissimilarity measure, if we increase the dissimilarity measure between two points in different clusters and decrease the dissimilarity measure between two points in the same cluster, then the clustering function still outputs the same partition of clusters. Such a change of a dissimilarity measure is called a *consistent* change.

The impossibility theorem is based on the fundamental result that the output of any clustering function satisfying the scale invariance property and the consistency property is in a collection of *antichain* partitions, i.e., there is no partition in that collection that in turn is a *refinement* of another partition in that collection. As such, the richness property cannot be satisfied. In [29], it was argued that the impossibility theorem is *not* an inherent feature of clustering. The key point in [29] is that the consistency property may not be a desirable property for a clustering function. This can be illustrated by considering a consistent change of 5 clusters in Figure 1. The figure is redrawn from Figure 1 in [29] that originally consists of 6 clusters. On the left hand side of Figure 1, it seems reasonable to have a partition of 5 clusters. However, after the consistent change, a new partition of 3 clusters might be a better output than the original partition of 5 clusters. As such, they abandoned the three axioms for *clustering functions* and proposed another three similar axioms for Clustering-Quality Measures (CGM) (for measuring the quality of a partition). They showed the existence of a CGM that satisfies their three axioms for CGMs.

As for the definition-based approach, most of the definitions of a single cluster in the literature are based on loosely defined terms [1]. One exception is [30], where Ester et al. provided a precise definition of a single cluster based on the concept of density-based reachability. A point p is said to be directly density-reachable from another point q if point p lies within the ϵ-neighborhood of point q and the ϵ-neighborhood of point q contains at least a minimum number of points. A point is said to be density-reachable from another point if they are connected by a sequence of directly density-reachable points. Based on the concept of density-reachability, a cluster is defined as a maximal set of points that are density-reachable from each other. An intuitive way to see such a definition for a cluster in a set of data points is to convert the data set into a graph. Specifically, if we put a directed edge from one point p to another point q if point p is directly density-reachable from point q, then a cluster simply corresponds to a *strongly connected component* in the graph. One of the problems for such a definition is that it requires specifying two parameters, ϵ and the minimum number of points in a ϵ-neighborhood. As pointed out in [30], it is not an easy task to determine these two parameters.

In this paper, we make an attempt to develop a clustering theory in metric spaces. In Section II, we first address the question:

What is a cluster in a set of data points in metric spaces?

For this, we first propose a new *cohesion* measure in terms of the distance measure. Using the cohesion measure, we define a cluster as a set of points that are cohesive to themselves. For such a definition, we show in Theorem 7 that there are various equivalent statements and these statements can be explained intuitively. We then consider the second question:

How do we find clusters and good partitions of clusters under such a definition?

For such a question, we propose a hierarchical agglomerative algorithm in Section III and a partitional algorithm Section IV. Unlike standard hierarchical agglomerative algorithms, our hierarchical agglomerative algorithm has a specific stopping criterion. Moreover, we show in Theorem 9 that our hierarchical agglomerative algorithm returns a partition of clusters when it stops. Our partitional algorithm, called the K-sets algorithm in the paper, appears to be a new iterative algorithm. Unlike the Lloyd iteration that needs two-step minimization, our K-sets algorithm only takes one-step minimization. We further show in Theorem 14 that the K-sets algorithm converges in a finite number of iterations. Moreover, for $K = 2$, the K-sets algorithm returns two clusters when the algorithm converges.

TABLE I
LIST OF NOTATIONS

$\Omega = \{x_1, x_2, \ldots, x_n\}$	The set of all data points
n	The total number of data points
$d(x, y)$	The distance between two points x and y
$\bar{d}(S_1, S_2)$	The *average* distance between two sets S_1 and S_2 in (10)
$\mathrm{RD}(x\|\|y) = d(x, y) - \bar{d}(\{x\}, \Omega)$	The *relative* distance from x to y
$\mathrm{RD}(y) = \bar{d}(\Omega, \{y\}) - \bar{d}(\Omega, \Omega)$	The *relative* distance from a random point to y
$\mathrm{RD}(S_1\|\|S_2) = \bar{d}(S_1, S_2) - \bar{d}(S_1, \Omega)$	The *relative distance* from a set S_1 to another set S_2
$\gamma(x, y) = \mathrm{RD}(y) - \mathrm{RD}(x\|\|y)$	The *cohesion measure* between two points x and y
$\gamma(S_1, S_2) = \sum_{x \in S_1} \sum_{y \in S_2} \gamma(x, y)$	The *cohesion measure* between two sets S_1 and S_2
$\Delta(x, S) = 2\bar{d}(\{x\}, S) - \bar{d}(S, S)$	The *triangular distance* from a point x to a set S
$Q = \sum_{k=1}^{K} \gamma(S_k, S_k)$	The *modularity* for a partition $S_1, S_2, \ldots, S_K$ of Ω
$R = \sum_{k=1}^{K} \gamma(S_k, S_k)/\|S_k\|$	The *normalized modularity* for a partition $S_1, S_2, \ldots, S_K$ of Ω

One of the most interesting findings of our paper is the duality result between a distance measure and a cohesion measure. In Section V, we first provide a general definition of a cohesion measure. We show that there is an induced distance measure, called the *dual distance measure*, for each cohesion measure. On the other hand, there is also an induced cohesion measure, called the *dual cohesion measure*, for each distance measure. In Theorem 18, we further show that the dual distance measure of a dual cohesion measure of a distance measure is the distance measure itself. Such a duality result leads to a dual K-sets algorithm for clustering a set of data points with a cohesion measure. The dual K-sets algorithm converges in the same way as a sequential version of the classical kernel K-means algorithm. The key difference is that a cohesion measure does not need to be positive semi-definite.

In Table I, we provide a list of notations used in this paper.

II. CLUSTERS IN METRIC SPACES

A. What is a cluster?

As pointed out in [4], one of the fundamental challenges associated with clustering is to address the following question:

What is a cluster in a set of data points?

In this paper, we will develop a clustering theory that formally defines a cluster for data points in a metric space. Specifically, we consider a set of n data points, $\Omega = \{x_1, x_2, \ldots, x_n\}$ and a distance measure $d(x, y)$ for any two points x and y in Ω. The distance measure $d(\cdot, \cdot)$ is assumed to be a *metric* and it satisfies

(D1) $d(x, y) \geq 0$;
(D2) $d(x, x) = 0$;
(D3) (Symmetric) $d(x, y) = d(y, x)$;
(D4) (Triangular inequality) $d(x, y) \leq d(x, z) + d(z, y)$.

Such a metric assumption is stronger than the usual dissimilarity (similarity) measures [33], where the triangular inequality in general does not hold. We also note that (D2) is usually stated as a necessary and sufficient condition in the literature, i.e., $d(x, y) = 0$ if and only if $x = y$. However, we only need the sufficient part in this paper. Our approach begins with a definition-based approach. We first give a specific definition of what a cluster is (without the need of specifying any parameters) and show those axiom-like properties are indeed satisfied under our definition of a cluster.

B. Relative distance and cohesion measure

One important thing that we learn from the consistent change in Figure 1 is that a good partition of clusters should be looked at a global level and the *relative* distances among clusters should be considered as an important factor. The distance measure between any two points only gives an absolute value and it does not tell us how close these two points are relative to the whole set of data points. The key idea of defining the relative distance from one point x to another point y is to choose another random point z as a reference point and compute the relative distance as the average of $d(x, y) - d(x, z)$ for all the points z in Ω. This leads to the following definition of relative distance.

Definition 1 (Relative distance) *The* relative distance *from a point x to another point y, denoted by RD$(x||y)$, is defined as follows:*

$$\begin{aligned} \mathit{RD}(x||y) &= \frac{1}{n} \sum_{z \in \Omega} (d(x, y) - d(x, z)) \\ &= d(x, y) - \frac{1}{n} \sum_{z \in \Omega} d(x, z). \end{aligned} \quad (1)$$

The relative distance (from a random point) to a point y, denoted by RD(y), is defined as the average relative distance from a random point to y, i.e.,

$$\begin{aligned} \mathit{RD}(y) &= \frac{1}{n} \sum_{z \in \Omega} \mathit{RD}(z||y) \\ &= \frac{1}{n} \sum_{z_2 \in \Omega} d(z_2, y) - \frac{1}{n^2} \sum_{z_2 \in \Omega} \sum_{z_1 \in \Omega} d(z_2, z_1). \end{aligned} \quad (2)$$

Note from (1) that in general $\mathrm{RD}(x||y)$ is not symmetric, i.e., $\mathrm{RD}(x||y) \neq \mathrm{RD}(y||x)$. Also, $\mathrm{RD}(x||y)$ may not be *nonnegative*. In the following, we extend the notion of relative distance from one point to another point to the relative distance from one set to another set.

Definition 2 (Relative distance) *The* relative distance *from a set of points S_1 to another set of points S_2, denoted by RD$(S_1||S_2)$, is defined as the average relative distance from a random point in S_1 to another random point in S_2, i.e.,*

$$\mathit{RD}(S_1||S_2) = \frac{1}{|S_1| \cdot |S_2|} \sum_{x \in S_1} \sum_{y \in S_2} \mathit{RD}(x||y). \quad (3)$$

Based on the notion of relative distance, we define a cohesion measure for two points x and y below.

Definition 3 (Cohesion measure between two points) *Define the* cohesion measure *between two points x and y, denoted by $\gamma(x,y)$, as the difference of the relative distance to y and the relative distance from x to y, i.e.,*

$$\gamma(x,y) = \textit{RD}(y) - \textit{RD}(x||y). \tag{4}$$

Two points x and y are said to be cohesive *(resp.* incohesive*) if $\gamma(x,y) \geq 0$ (resp. $\gamma(x,y) \leq 0$).*

In view of (4), two points x and y are cohesive if the relative distance from x to y is not larger than the relative distance (from a random point) to y.

Note from (1) and (2) that

$$\begin{aligned}\gamma(x,y) &= \text{RD}(y) - \text{RD}(x||y) \\ &= \frac{1}{n}\sum_{z_2\in\Omega} d(z_2,y) + \frac{1}{n}\sum_{z_1\in\Omega} d(x,z_1) \\ &\qquad -\frac{1}{n^2}\sum_{z_2\in\Omega}\sum_{z_1\in\Omega} d(z_2,z_1) - d(x,y) \quad (5) \\ &= \frac{1}{n^2}\sum_{z_2\in\Omega}\sum_{z_1\in\Omega}\Big(d(x,z_1)+d(z_2,y) \\ &\qquad -d(z_1,z_2)-d(x,y)\Big). \quad (6)\end{aligned}$$

Though there are many ways to define a cohesion measure for a set of data points in a metric space, our definition of the cohesion measure in Definition 3 has the following four desirable properties. Its proof is based on the representations in (5) and (6) and it is given in Appendix A.

Proposition 4 (i) *(Symmetry) The cohesion measure is symmetric, i.e., $\gamma(x,y) = \gamma(y,x)$.*

(ii) *(Self-cohesiveness) Every data point is cohesive to itself, i.e., $\gamma(x,x) \geq 0$.*

(iii) *(Self-centredness) Every data point is more cohesive to itself than to another point, i.e., $\gamma(x,x) \geq \gamma(x,y)$ for all $y \in \Omega$.*

(iv) *(Zero-sum) The sum of the cohesion measures between a data point to all the points in Ω is zero, i.e., $\sum_{y\in\Omega}\gamma(x,y) = 0$.*

These four properties can be understood intuitively by viewing a cohesion measure between two points as a "binding force" between those two points. The symmetric property ensures that the binding force is reciprocated. The self-cohesiveness property ensures that each point is self-binding. The self-centredness property further ensures that the self binding force is always stronger than the binding force to the other points. In view of the zero-sum property, we know for every point x there are points that are incohesive to x and each of these points has a negative binding force to x. Also, there are points that are cohesive to x (including x itself from the self-cohesiveness property) and each of these points has a positive force to x. As such, the binding force will naturally "push" points into "clusters."

To further understand the intuition of the cohesion measure, we can think of z_1 and z_2 in (6) as two random points that are used as reference points. Then two points x and y are cohesive if $d(x,z_1) + d(z_2,y) \geq d(z_1,z_2) + d(x,y)$ for two reference points z_1 and z_2 that are randomly chosen from Ω. In Figure 2, we show an illustrating example for such an intuition in $\mathcal{R}^2$. In Figure 2(a), point x is close to one reference point z_1 and point y is close to the other reference point z_2. As such, $d(x,z_1) + d(z_2,y) \leq d(z_1,z_2)$ and thus these two points x and y are incohesive. In Figure 2(b), point x is not that close to z_1 and point y is not that close to z_2. However, x and y are on the two opposite sides of the segment between the two reference points z_1 and z_2. As such, there are two triangles in this graph: the first triangle consists of the three points x, z_1, and w, and the second triangle consists of the three points y, z_2, and w. From the triangular inequality, we then have $d(w,z_1) + d(x,w) \geq d(x,z_1)$ and $d(y,w) + d(w,z_2) \geq d(y,z_2)$. Since $d(w,z_1) + d(w,z_2) = d(z_1,z_2)$ and $d(x,w) + d(y,w) = d(x,y)$, it then follows that $d(z_1,z_2) + d(x,y) \geq d(x,z_1) + d(y,z_2)$. Thus, points x and y are also incohesive in Figure 2(b). In Figure 2(c), point x is not that close to z_1 and point y is not that close to z_2 as in Figure 2(b). Now x and y are on the same side of the segment between the two reference points z_1 and z_2. There are two triangles in this graph: the first triangle consists of the three points x, y, and w, and the second triangle consists of the three points z_1, z_2, and w. From the triangular inequality, we then have $d(x,w) + d(w,y) \geq d(x,y)$ and $d(w,z_1) + d(w,z_2) \geq d(z_1,z_2)$. Since $d(x,w) + d(w,z_1) = d(x,z_1)$ and $d(w,y) + d(w,z_2) = d(z_2,y)$, it then follows that $d(x,z_1) + d(y,z_2) \geq d(z_1,z_2) + d(x,y)$. Thus, points x and y are cohesive in Figure 2(c). In view of Figure 2(c), it is intuitive to see that two points x and y are cohesive if *they both are far away from the two reference points and they both are close to each other*.

The notions of relative distance and cohesion measure are also related to the notion of relative centrality in our previous work [34]. To see this, suppose that we sample two points x and y from Ω according to the following bivariate distribution:

$$p(x,y) = \frac{e^{-\theta d(x,y)}}{\sum_{u\in\Omega}\sum_{v\in\Omega} e^{-\theta d(u,v)}}, \quad \theta > 0. \tag{7}$$

Let $P_X(x) = \sum_{y\in\Omega} p(x,y)$ and $P_Y(y) = \sum_{x\in\Omega} p(x,y)$ be the two marginal distributions. Then one can verify that the covariance $p(x,y) - P_X(x)P_Y(y)$ is proportional to the cohesion measure $\gamma(x,y)$ when $\theta \downarrow 0$. Intuitively, two points x and y are cohesive if they are positively correlated according to the sampling in (7) when θ is very small.

Now we extend the cohesion measure between two points to the cohesion measure between two sets.

Definition 5 (Cohesion measure between two sets) *Define the* cohesion measure *between two sets S_1 and S_2, denoted by $\gamma(S_1,S_2)$, as the sum of the cohesion measures of all the pairs of two points (with one point in S_1 and the other point in S_2), i.e.,*

$$\gamma(S_1,S_2) = \sum_{x\in S_1}\sum_{y\in S_2}\gamma(x,y). \tag{8}$$

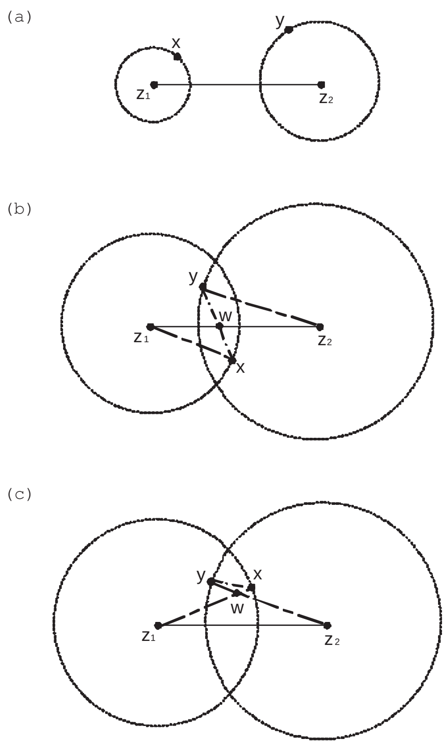

Fig. 2. Illustrating examples of the cohesion measure in $\mathcal{R}^2$: (a) incohesive as $d(x,z_1)+d(y,z_2) \leq d(z_1,z_2)$, (b) incohesive as $d(w,z_1)+d(x,w) \geq d(x,z_1)$ and $d(y,w)+d(w,z_2) \geq d(y,z_2)$, and (c) cohesive $d(x,w)+d(w,y) \geq d(x,y)$ and $d(w,z_1)+d(w,z_2) \geq d(z_1,z_2)$.

Two sets S_1 and S_2 are said to be cohesive *(resp.* incohesive*) if $\gamma(S_1,S_2) \geq 0$ (resp. $\gamma(S_1,S_2) \leq 0$).*

C. Equivalent statements of clusters

Now we define what a cluster is in terms of the cohesion measure.

Definition 6 (Cluster) *A nonempty set S is called a* cluster *if it is cohesive to itself, i.e.,*

$$\gamma(S,S) \geq 0. \tag{9}$$

In the following, we show the first main theorem of the paper. Its proof is given in Appendix B.

Theorem 7 *Consider a* nonempty *set S that is not equal to Ω. Let $S^c = \Omega\backslash S$ be the set of points that are not in S. Also, let $\bar{d}(S_1,S_2)$ be the average distance between two randomly selected points with one point in S_1 and another point in S_2, i.e.,*

$$\bar{d}(S_1,S_2) = \frac{1}{|S_1| \times |S_2|} \sum_{x \in S_1} \sum_{y \in S_2} d(x,y). \tag{10}$$

The following statements are equivalent.

(i) *The set S is a cluster, i.e., $\gamma(S,S) \geq 0$.*
(ii) *The set S^c is a cluster, i.e., $\gamma(S^c,S^c) \geq 0$.*
(iii) *The two sets S and S^c are incohesive, i.e., $\gamma(S,S^c) \leq 0$.*
(iv) *The set S is more cohesive to itself than to S^c, i.e., $\gamma(S,S) \geq \gamma(S,S^c)$.*
(v) $2\bar{d}(S,\Omega) - \bar{d}(\Omega,\Omega) - \bar{d}(S,S) \geq 0$.
(vi) *The relative distance from Ω to S is not smaller than the relative distance from S to S, i.e., RD$(\Omega||S) \geq$ RD$(S||S)$.*
(vii) *The relative distance from S^c to S is not smaller than the relative distance from S to S, i.e., RD$(S^c||S) \geq$ RD$(S||S)$.*
(viii) $2\bar{d}(S,S^c) - \bar{d}(S,S) - \bar{d}(S^c,S^c) \geq 0$.
(ix) *The relative distance from S to S^c is not smaller than the relative distance from Ω to S^c, i.e., RD$(S||S^c) \geq$ RD$(\Omega||S^c)$.*
(x) *The relative distance from S^c to S is not smaller than the relative distance from Ω to S, i.e., RD$(S^c||S) \geq$ RD$(\Omega||S)$.*

One surprise finding in Theorem 7(ii) is that the set S^c is also a cluster. This shows that the points inside S are cohesive and the points outside S are also cohesive. Thus, there seems a boundary between S and S^c from the cohesion measure. Another surprise finding is in Theorem 7(viii). One usually would expect that a cluster S should satisfy $\bar{d}(S,S) \leq \bar{d}(S,S^c)$. But it seems our definition of a cluster is much weaker than that. Regarding the scale invariance property, it is easy to see from Theorem 7(viii) that the inequality there is still satisfied if we scale the distance measure by a constant factor. Thus, a cluster of data points is still a cluster after scaling the distance measure by a constant factor. Regarding the richness property, we argue that there exists a distance measure such that any subset of points in Ω is a cluster. To see this, we simply let the distance between any two points in the subset be equal to 0 and the distance between a point outside the subset to a point in the subset be equal to 1. Since a point x itself is a cluster, i.e., $\gamma(x,x) \geq 0$, we then have $\gamma(x,y) = \gamma(x,x) \geq 0$ for any two points x and y in the subset. From (9), the subset is a cluster under such a choice of the distance measure. Furthermore, one can also see from Theorem 7(vii) that for a cluster S, if we decrease the relative distance between two points in S and increase the relative distance between one point in S and another point in S^c, then the set S is still a cluster under such a "consistent" change.

We also note that in our proof of Theorem 7 we only need $d(\cdot,\cdot)$ to be symmetric. As such, the results in Theorem 7 also hold even when the triangular inequality is not satisfied.

III. A HIERARCHICAL AGGLOMERATIVE ALGORITHM

Once we define what a cluster is, our next question is

How do we find clusters and good partitions of clusters?

For this, we turn to an objective-based approach. We will show that clusters can be found by optimizing two specific objective functions by a hierarchical algorithm in Section III and a partitional algorithm in Section IV.

In the following, we first define a quality measure for a partition of Ω.

Definition 8 (Modularity) *Let S_k, $k = 1,2,\ldots,K$, be a partition of $\Omega = \{x_1,x_2,\ldots,x_n\}$, i.e., $S_k \cap S_{k'}$ is an empty set for $k \neq k'$ and $\cup_{k=1}^K S_k = \Omega$. The modularity index Q*

ALGORITHM 1: The Hierarchical Agglomerative Algorithm

```
Input: A data set $\Omega = \{x_1, x_2, \ldots, x_n\}$ and a distance measure $d(\cdot, \cdot)$.
Output: A partition of clusters $\{S_1, S_2, \ldots, S_K\}$.
Initially, $K = n$; $S_i = \{x_i\}$, $i = 1, 2, \ldots, n$;
Compute the cohesion measures $\gamma(S_i, S_j) = \gamma(x_i, x_j)$ for all $i, j = 1, 2, \ldots, n$;
while there exists some $i$ and $j$ such that $\gamma(S_i, S_j) > 0$ do
    Merge $S_i$ and $S_j$ into a new set $S_k$, i.e., $S_k = S_i \cup S_j$;
    $\gamma(S_k, S_k) = \gamma(S_i, S_i) + 2\gamma(S_i, S_j) + \gamma(S_j, S_j)$;
    for each $\ell \neq k$ do
        $\gamma(S_k, S_\ell) = \gamma(S_\ell, S_k) = \gamma(S_i, S_\ell) + \gamma(S_j, S_\ell)$;
    end
    $K = K - 1$;
end
Reindex the $K$ remaining sets to $\{S_1, S_2, \ldots, S_K\}$;
```

with respect to the partition S_k, $k = 1, 2, \ldots, K$, is defined as follows:

$$Q = \sum_{k=1}^{K} \gamma(S_k, S_k). \tag{11}$$

Based on such a quality measure, we can thus formulate the clustering problem as an optimization problem for finding a partition $S_1, S_2, \ldots, S_K$ (for some unknown K) that maximizes the modularity index Q. Note that

$$Q = \sum_{k=1}^{K} \gamma(S_k, S_k) = \sum_{k=1}^{K} \sum_{x \in S_k} \sum_{y \in S_k} \gamma(x, y) = \sum_{x \in \Omega} \sum_{y \in \Omega} \gamma(x, y) \delta_{c(x), c(y)}, \tag{12}$$

where $c(x)$ is the cluster of x and $\delta_{c(x),c(y)} = 1$ if x and y are in the same cluster. In view of (12), another way to look at the optimization problem is to find the assignment of each point to a cluster. However, it was shown in [35] that finding the optimal assignment for modularity maximization is NP-complete in the strong sense and thus heuristic algorithms, such as hierarchical algorithms and partitional algorithms are commonly used in the literature for solving the modularity maximization problem.

In Algorithm 1, we propose a hierarchical agglomerative clustering algorithm that converges to a local optimum of this objective. The algorithm starts from n clusters with each point itself as a cluster. It then recursively merges two disjoint cohesive clusters to form a new cluster until either there is a single cluster left or all the remaining clusters are incohesive. There are two main differences between a standard hierarchical agglomerative clustering algorithm and ours:

(i) Stopping criterion: in a standard hierarchical agglomerative clustering algorithm, such as single linkage or complete linkage, there is no stopping criterion. Here our algorithm stops when all the remaining clusters are incohesive.

(ii) Greedy selection: our algorithm only needs to select a pair of cohesive clusters to merge. It does not need to be the most cohesive pair. This could potentially speed up the algorithm in a large data set.

In the following theorem, we show that the modularity index Q in (11) is non-decreasing in every iteration of the hierarchical agglomerative clustering algorithm and it indeed produces clusters. Its proof is given in Appendix C.

Theorem 9 (i) *Every set returned by the hierarchical agglomerative clustering algorithm is indeed a cluster.*

(ii) *For the hierarchical agglomerative clustering algorithm, the modularity index is non-decreasing in every iteration and thus converges to a local optimum.*

As commented before, our algorithm only requires to find a pair of cohesive clusters to merge in each iteration. This is different from the greedy selection in [1], Chapter 13, and [36]. Certainly, our hierarchical agglomerative clustering algorithm can also be operated in a greedy manner. As in [36], in each iteration we can merge the two clusters that result in the largest increase of the modularity index, i.e., the most cohesive pair. It is well-known (see e.g., the book [2]) that a näive implementation of a greedy hierarchical agglomerative clustering algorithm has $O(n^3)$ computational complexity and the computational complexity can be further reduced to $O(n^2 \log(n))$ if priority queues are implemented for the greedy selection. We also note that there are several hierarchical agglomerative clustering algorithms proposed in the literature for community detection in networks (see e.g., [37], [38], [39], [40]). These algorithms are also based on "modularity" maximization. Among them, the fast unfolding algorithm in [38] is the fast one as there is a second phase of building a new (and much smaller) network whose nodes are the communities found during the previous phase. The Newman and Girvan modularity in [37] is based on a probability measure from a random selection of an edge in a network (see [34] for more detailed discussions) and this is different from the distance metric used in this paper.

In the following, we provide an illustrating example for our hierarchical agglomerative clustering algorithm by using the greedy selection of the most cohesive pair.

Example 10 (Zachary's karate club) As in [37], [41], we consider the Zachary's karate club friendship network [42]. The set of data was observed by Wayne Zachary [42] over the course of two years in the early 1970s at an American university. During the course of the study, the club split into two groups because of a dispute between its administrator (node 34) and its instructor (node 1).

In Figure 3, we show the dendrogram generated by using our hierarchical agglomerative clustering algorithm with the greedy selection of the most cohesive pair in each iteration. The distance measure is the geodesic distance of the graph in [42]. The algorithm stops when there are three incohesive clusters left, one led by the administrator (node 34), one led

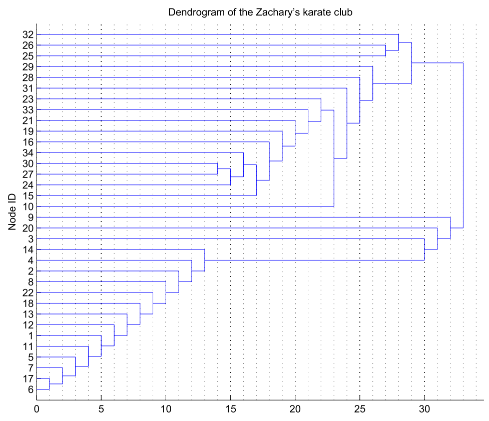

Fig. 3. The dendrogram from our greedy hierarchical agglomerative clustering algorithm for the Zachary karate club friendship network.

by the instructor (node 1), and person number 9 himself. According to [42], there was an interesting story for person number 9. He was a weak supporter for the administrator. However, he was only three weeks away from a test for black belt (master status) when the split of the club occurred. He would have had to give up his rank if he had joined the administrator's club. He ended up with the instructor's club. We also run an additional step for our algorithm (to merge the pair with the largest cohesive measure) even though the remaining three clusters are incohesive. The additional step reveals that person number 9 is clustered into the instructor's club.

IV. A PARTITIONAL ALGORITHM

A. Triangular distance

In this section, we consider another objective function.

Definition 11 (normalized modularity) *Let S_k, $k = 1, 2, \dots, K$, be a partition of $\Omega = \{x_1, x_2, \dots, x_n\}$, i.e., $S_k \cap S_{k'}$ is an empty set for $k \neq k'$ and $\cup_{k=1}^K S_k = \Omega$. The normalized modularity index R with respect to the partition S_k, $k = 1, 2, \dots, K$, is defined as follows:*

$$R = \sum_{k=1}^{K} \frac{1}{|S_k|} \gamma(S_k, S_k). \tag{13}$$

Unlike the hierarchical agglomerative clustering algorithm in the previous section, in this section we assume that K is fixed and known in advance. As such, we may use an approach similar to the classical K-means algorithm by iteratively assigning each point to the nearest set (until it converges). Such an approach requires a measure that can measure how close a point x to a set S is. In the K-means algorithm, such a measure is defined as the square of the distance between x and the centroid of S. However, there is no centroid for a set in a non-Euclidean space and we need to come up with another measure.

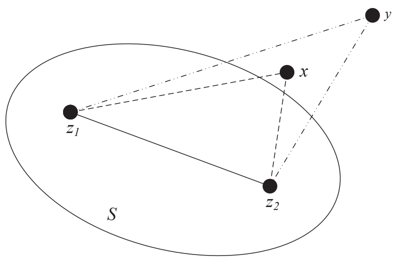

Fig. 4. An illustration of the triangular distance in $\mathcal{R}^2$.

Our idea for measuring the distance from a point x to a set S is to randomly choose two points z_1 and z_2 from S and consider the three sides of the triangle x, z_1 and z_2. Note that the triangular inequality guarantees that $d(x, z_1) + d(x, z_2) - d(z_1, z_2) \geq 0$. Moreover, if x is close to z_1 and z_2, then $d(x, z_1) + d(x, z_2) - d(z_1, z_2)$ should also be small. We illustrate such an intuition in Figure 4, where there are two points x and y and a set S in $\mathcal{R}^2$. Such an intuition leads to the following definition of triangular distance from a point x to a set S.

Definition 12 (Triangular distance) *The* triangular distance *from a point x to a set S, denoted by $\Delta(x, S)$, is defined as follows:*

$$\Delta(x, S) = \frac{1}{|S|^2} \sum_{z_1 \in S} \sum_{z_2 \in S} \Big(d(x, z_1) + d(x, z_2) - d(z_1, z_2) \Big). \tag{14}$$

In the following lemma, we show several properties of the triangular distance and its proof is given in Appendix D.

Lemma 13 (i)

$$\Delta(x, S) = 2\bar{d}(\{x\}, S) - \bar{d}(S, S) \geq 0. \tag{15}$$

(ii)

$$\Delta(x, S) = \gamma(x, x) - \frac{2}{|S|}\gamma(\{x\}, S) + \frac{1}{|S|^2}\gamma(S, S). \tag{16}$$

(iii) *Let S_k, $k = 1, 2, \dots, K$, be a partition of $\Omega = \{x_1, x_2, \dots, x_n\}$. Then*

$$\sum_{k=1}^{K} \sum_{x \in S_k} \Delta(x, S_k) = \sum_{x \in \Omega} \gamma(x, x) - R. \tag{17}$$

(iv) *Let S_k, $k = 1, 2, \dots, K$, be a partition of $\Omega = \{x_1, x_2, \dots, x_n\}$ and $c(x)$ be the index of the set to which x belongs, i.e., $x \in S_{c(x)}$. Then*

$$\begin{aligned} \sum_{k=1}^{K} \sum_{x \in S_k} \Delta(x, S_k) &= \sum_{k=1}^{K} \sum_{x \in S_k} \bar{d}(\{x\}, S_k) \\ &= \sum_{x \in \Omega} \bar{d}(\{x\}, S_{c(x)}). \end{aligned} \tag{18}$$

ALGORITHM 2: The K-sets Algorithm

```
Input: A data set $\Omega = \{x_1, x_2, \ldots, x_n\}$, a distance
       measure $d(\cdot, \cdot)$, and the number of sets $K$.
Output: A partition of sets $\{S_1, S_2, \ldots, S_K\}$.
(0) Initially, choose arbitrarily $K$ disjoint nonempty sets
$S_1, \ldots, S_K$ as a partition of $\Omega$.
(1) for $i = 1, 2, \ldots, n$ do
    Compute the triangular distance $\Delta(x_i, S_k)$ for each
    set $S_k$ by using (15).
    Find the set to which the point $x_i$ is closest in terms
    of the triangular distance.
    Assign point $x_i$ to that set.
end
(2) Repeat from (1) until there is no further change.
```

The first property of this lemma is to represent triangular distance by the average distance. The second property is to represent the triangular distance by the cohesion measure. Such a property plays an important role for the duality result in Section V. The third property shows that the optimization problem for maximizing the normalized modularity R is equivalent to the optimization problem that minimizes the sum of the triangular distance of each point to its set. The fourth property further shows that such an optimization problem is also equivalent to the optimization problem that minimizes the sum of the average distance of each point to its set. Note that $\bar{d}(\{x\}, S_k) = \frac{1}{|S_k|}\sum_{y \in S_k} d(x, y)$. The objective for maximizing the normalized modularity R is also equivalent to minimize

$$\sum_{k=1}^{K} \frac{1}{|S_k|} \sum_{x \in S_k} \sum_{y \in S_k} d(x, y).$$

This is different from the K-median objective, the K-means objective and the min-sum objective addressed in [21].

B. The K-sets algorithm

In the following, we propose a partitional clustering algorithm, called the K-sets algorithm in Algorithm 2, based on the triangular distance. The algorithm is very simple. It starts from an arbitrary partition of the data points that contains K disjoint sets. Then for each data point, we assign the data point to the closest set in terms of the triangular distance. We repeat the process until there is no further change. Unlike the Lloyd iteration that needs two-step minimization, the K-sets algorithm only takes one-step minimization. This might give the K-sets algorithm the computational advantage over the K-medoids algorithms [6], [1], [7], [8].

In the following theorem, we show the convergence of the K-sets algorithm. Moreover, for $K = 2$, the K-sets algorithm yields two clusters. Its proof is given in Appendix E.

Theorem 14 (i) *In the K-sets algorithm based on the triangular distance, the normalized modularity is increasing when there is a change, i.e., a point is moved from one set to another. Thus, the algorithm converges to a local optimum of the normalized modularity.*

(ii) *Let $S_1, S_2, \ldots, S_K$ be the K sets when the algorithm converges. Then for all $i \neq j$, the two sets S_i and S_j are two clusters if these two sets are viewed in isolation (by removing the data points not in $S_i \cup S_j$ from Ω).*

An immediate consequence of Theorem 14 (ii) is that for $K = 2$, the two sets S_1 and S_2 are clusters when the algorithm converges. However, we are not able to show that for $K \geq 3$ the K sets, $S_1, S_2, \ldots, S_K$, are clusters in Ω. On the other hand, we are not able to find a counterexample either. All the numerical examples that we have tested for $K \geq 3$ yield K clusters.

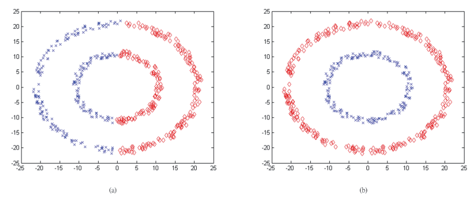

Fig. 5. Two rings: (a) a clustering result by the K-means algorithm, and (b) a clustering result by the K-sets algorithm.

C. Experiments

In this section, we report several experimental results for the K-sets algorithm: including the dataset with two rings in Section IV-C1, the stochastic block model in Section IV-C2, and the mixed National Institute of Standards and Technology dataset in Section IV-C3.

1) Two rings: In this section, we first provide an illustrating example for the K-sets algorithm.

In Figure 5, we generate two rings by randomly placing 500 points in $\mathcal{R}^2$. The outer (resp. inner) ring consists of 300 (resp. 200) points. The radius of a point in the outer (resp. inner) ring is uniformly distributed between 20 and 22 (resp. 10 and 12). The angle of each point is uniformly distributed between 0 and 2π. In Figure 5(a), we show a typical clustering result by using the classical K-means algorithm with $K = 2$. As the centroids of these two rings are very close to each other, it is well-known that the K-means algorithm does not perform well for the two rings. Instead of using the Euclidean distance as the distance measure for our K-sets algorithm, we first convert the two rings into a graph by adding an edge between two points with the Euclidean distance less than 5. Then the distance measure between two points is defined as the geodesic distance of these two points in the graph. By doing so, we can then easily separate these rings by using the K-sets algorithm with $K = 2$ as shown in Figure 5(b).

The purpose of this example is to show the limitation of the applicability of the K-means algorithm. The data points for the K-means algorithm need to be in some Euclidean space. On the other hand, the data points for the K-sets algorithms only need to be in some metric space. As such, the distance matrix constructed from a graph cannot be directly applied by the K-means algorithm while it is still applicable for the K-sets algorithm. We note that one can still apply the K-means algorithm by embedding the data points in a non-Euclidean space into a Euclidean space. One common approach of doing that is known as *spectral clustering* in the literature (see e.g., [14], [43], [44], [45], [19]).

2) Stochastic block model: The stochastic block model (SBM), as a generalization of the Erdös-Rényi random graph [46], is a commonly used method for generating random graphs that can be used for benchmarking community detection algorithms [47], [48]. In a stochastic block model with q blocks (communities), the total number of nodes in the random graph are evenly distributed to these q blocks. The probability that there is an edge between two nodes within the same block is p_{in} and the probability that there is an edge between two nodes in two different blocks is p_{out}. These edges are generated independently. Let $c_{in} = n \cdot p_{in}$, $c_{out} = n \cdot p_{out}$. Then it is known (see e.g., [49], [50], [47]) that these q communities can be detected (in theory for a large network) if

$$|c_{in} - c_{out}| > q\sqrt{\text{mean degree}}. \tag{19}$$

In this paper, we use MODE-NET [48] to run SBM. Specifically, we consider a stochastic block model with two blocks. The number of nodes in the stochastic block model is 1,000 with 500 nodes in each of these two blocks. The average degree of a node is set to be 3. The values of $c_{in} - c_{out}$ of these graphs are in the range from 2.5 to 5.9 with a common step of 0.1. We generate 20 graphs for each $c_{in} - c_{out}$. Isolated vertices are removed. Thus, the exact numbers of vertices used in this experiment are slightly less than than 1,000.

We compare the K-sets algorithm (in Algorithm 2) and the hierarchical agglomerative algorithm (in Algorithm 1) with some other community detection algorithms, such as OSLOM2 [51], infomap [52], [53], and fast unfolding [38]. The metric used for the K-sets algorithm and the hierarchical agglomerative algorithm for each sample of the random graph is the resistance distance, and this is pre-computed by NumPy [54]. The resistance distance matrix (denoted by $R = (R_{i,j})$) can be derived from the pseudo inverse of the Laplacian matrix (denoted by $\Gamma = (\Gamma_{i,j})$) as follows: [55]:

$$R_{i,j} = \begin{cases} 0, & \text{if } i = j, \\ \Gamma_{i,i} + \Gamma_{j,j} - \Gamma_{i,j} - \Gamma_{j,i}, & \text{otherwise.} \end{cases}$$

The K-sets algorithm, the hierarchical agglomerative algorithm and OSLOM2 are implemented in C++, and the others are all taken from igraph [56] and are implemented in C with python wrappers. In Table II, we show the average running times for these four algorithms over 700 trials. The pre-computation time for the K-sets algorithm is the time to compute the distance matrix. Except infomap, the other three algorithms are very fast. In Figure 6, we compute the normalized mutual information measure (NMI) by using a built-in function in igraph [56] for the results obtained from these four algorithms. Each point is averaged over 20 random graphs from the stochastic block model. The error bars are the 95% confidence intervals. In this stochastic block model, the theoretical phase transition threshold from (19) is $c_{in} - c_{out} = 3.46$. It seems that the K-sets algorithm is able to detect these two blocks when $c_{in} - c_{out} \geq 4.5$. Its performance in that range is better than the hierarchical agglomerative algorithm, infomap [52], [53], fast unfolding [38] and OSLOM2 [51]. We note that the comparison is not exactly fair as the hierarchical agglomerative algorithm and the other three algorithms do not have the information of the number of blocks (communities). We also note that our hierarchical agglomerative algorithm performs better than fast unfolding [38] in terms of NMI. A detailed examination of our experiments reveals that our hierarchical agglomerative algorithm ends up with only two or three clusters when $c_{in} - c_{out}$ is large. On the other hand, fast unfolding could produce more than 10 clusters in the same range. As we use greedy selection in our hierarchical agglomerative algorithm, its computation time is substantially longer than fast unfolding.

For the K-sets algorithm, we also conduct an experiment that uses the line graph (the edge-to-vertex dual graph) of the original graph in the stochastic block model. For this experiment, we use the K-sets algorithm to label the edges into two clusters according to the resistance distance of the line graph. Each node is then labelled by the majority vote of the labels of its edges. As shown in Figure 6, the K-sets algorithm using the line graph (the curve marked with K-sets (line graph)) performs better than the K-sets algorithm using the original graph. This shows that the choice of the right feature (and the associated distance metric) plays an important role in the clustering problem. In fact, it is known (see e.g., [57]) that the second eigenvector of the non-backtracking matrix is correlated with the communities in the stochastic block model and it can be used for producing nearly optimal community detection results in such a model. Feature selection (and the associated distance metric) is beyond the scope of this paper.

TABLE II
AVERAGE RUNNING TIME (IN SECONDS).

	pre-comp.	running	total
Infomap	0	0.7634	0.7634
Fast unfolding	0	0.0074	0.0074
OSLOM2	0	0.0059	0.0059
K-sets	2.3096	0.0060	2.3156
K-sets (line graph)	7.0140	0.0367	7.0507
Hierarchical	3.0076	4.9082	7.9158

3) Mixed National Institute of Standards and Technology dataset: In this section, we consider a real-world dataset, the mixed National Institute of Standards and Technology dataset (the MNIST dataset) [58]. The MNIST dataset contains 60,000 samples of hand-written digits. These samples are 28×28 pixels grayscale images (i.e., each of the image is a 784 dimensional data point). For our experiments, we select the first 1,000 samples from each set of the digit 0 to 9 to create a total number of 10,000 samples.

To fairly evaluate the performance of the K-sets algorithm, we compare the K-sets algorithm with two clustering algorithms in which the number of clusters is also known a priori, i.e., the K-means++ algorithm [59] and the K-medoids algorithm [60]. For the MNIST dataset, the number of clusters is 10 (for the ten digits, $0, 1, 2, \ldots, 9$). The K-

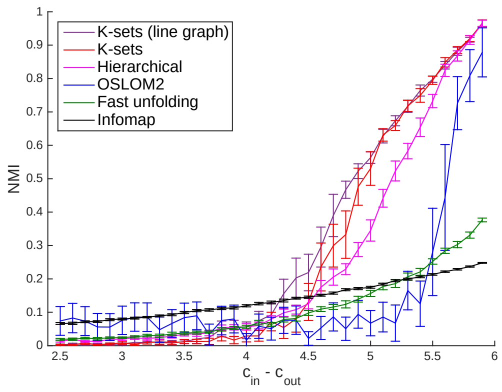

Fig. 6. Comparison of infomap [52], [53], fast unfolding [38], OSLOM2 [51], Hierarchical (Algorithm 1) and K-sets (Algorithm 2) for the stochastic block model with two blocks. Each point is averaged over 20 such graphs. The error bars are the 95% confidence intervals. The theoretical phase transition threshold in this case is 3.46.

means++ algorithm is an improvement of the standard K-means algorithm with a specific method to choose the initial centroids of the K clusters. Like the K-sets algorithm, the K-medoids algorithm is also a clustering algorithm that uses a distance measure. The key difference between the K-sets algorithm and the K-medoids algorithm is that we use the triangular distance to a set for the assignment of each data point and the K-medoids algorithm uses the distance to a medoid for such an assignment. The Euclidean distance between two data points (samples from the MNIST dataset) for the K-medoids algorithm and the K-sets algorithm are pre-computed by NumPy [54]. The K-sets algorithm is implemented in C++, and the others are implemented in C with python wrappers. All the programs are executed on an Acer Altos-T350-F2 machine with two Intel(R) Xeon(R) CPU E5-2690 v2 processors. In order to have a fair comparison of their running times, the parallelization of each program is disabled, i.e., only one core is used in these experiments. We assume that the input data is already stored in the main memory and the time consumed for I/O is not recorded.

In Table III, we show the average running times for these three algorithms over 100 trials. Both the K-medoids algorithm and the K-sets algorithm need to compute the distance matrix and this is shown in the row marked with the pre-computation time. The total running times for these three algorithms are roughly the same for this experiment. In Figure 7, we compute the normalized mutual information measure (NMI) by using a built-in function in igraph [56] for the results obtained from these three algorithms. Each point is averaged over 100 trials. The error bars are the 95% confidence intervals. In view of Figure 7, the K-sets algorithm outperforms the K-means++ algorithm and the K-medoids algorithm for the MNIST dataset. One possible explanation for this is that both the the K-means++ algorithm and the K-medoids algorithm only select a single representative data point for a cluster and that representative data point may not be able to represent the whole cluster well enough. On the other hand, the K-sets algorithm uses the triangular distance that takes the distance to every point in a cluster into account.

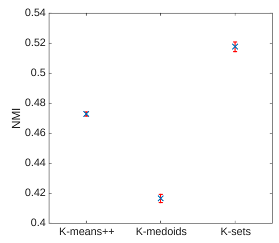

Fig. 7. Comparison of K-means++ [59], K-medoids [60] and K-sets for the MNIST dataset. Each point is averaged over 100 trials. The error bars are the 95% confidence intervals.

TABLE III
AVERAGE RUNNING TIME (IN SECONDS).

	K-means++	K-medoids	K-sets
pre-computation	0	36.981	36.981
running	49.940	1.228	1.801
total	49.940	38.209	38.782

V. DUALITY BETWEEN A COHESION MEASURE AND A DISTANCE MEASURE

A. The duality theorem

In this section, we show the duality result between a cohesion measure and a distance measure. In the following, we first provide a *general* definition for a cohesion measure.

Definition 15 *A measure between two points x and y, denoted by $\beta(x, y)$, is called a* cohesion measure *for a set of data points Ω if it satisfies the following three properties:*

(C1) *(Symmetry) $\beta(x, y) = \beta(y, x)$ for all $x, y \in \Omega$.*
(C2) *(Zero-sum) For all $x \in \Omega$, $\sum_{y\in\Omega} \beta(x, y) = 0$.*
(C3) *(Triangular inequality) For all x, y, z in Ω,*

$$\beta(x, x) + \beta(y, z) - \beta(x, z) - \beta(x, y) \geq 0. \quad (20)$$

In the following lemma, we show that the specific cohesion measure defined in Section II indeed satisfies (C1)–(C3) in Definition 15. Its proof is given in Appendix F.

Lemma 16 *Suppose that $d(\cdot, \cdot)$ is a distance measure for Ω, i.e., $d(\cdot, \cdot)$ satisfies (D1)–(D4). Let*

$$\begin{aligned}\beta(x, y) \quad = \quad & \frac{1}{n} \sum_{z_2\in\Omega} d(z_2, y) + \frac{1}{n} \sum_{z_1\in\Omega} d(x, z_1) \\ & -\frac{1}{n^2} \sum_{z_2\in\Omega} \sum_{z_1\in\Omega} d(z_2, z_1) - d(x, y). \quad (21)\end{aligned}$$

Then $\beta(x, y)$ is a cohesion measure for Ω.

We know from (5) that the cohesion measure $\gamma(\cdot, \cdot)$ defined in Section II has the following representation:

$$\begin{aligned}\gamma(x, y) = & \frac{1}{n} \sum_{z_2\in\Omega} d(z_2, y) + \frac{1}{n} \sum_{z_1\in\Omega} d(x, z_1) \\ & -\frac{1}{n^2} \sum_{z_2\in\Omega} \sum_{z_1\in\Omega} d(z_2, z_1) - d(x, y).\end{aligned}$$

As a result of Lemma 16, it also satisfies (C1)–(C3) in Definition 15. As such, we call the cohesion measure $\gamma(\cdot,\cdot)$ defined in Section II the *dual cohesion measure* of the distance measure $d(\cdot,\cdot)$.

On the other hand, if $\beta(x,y)$ is a cohesion measure for Ω, then there is an induced distance measure and it can be viewed as the *dual distance measure* of the cohesion measure $\beta(x,y)$. This is shown in the following lemma and its proof is given in Appendix G.

Lemma 17 *Suppose that $\beta(\cdot,\cdot)$ is a cohesion measure for Ω. Let*

$$d(x,y) = (\beta(x,x)+\beta(y,y))/2 - \beta(x,y). \tag{22}$$

Then $d(\cdot,\cdot)$ is a distance measure that satisfies (D1)–(D4).

In the following theorem, we show the duality result. Its proof is given in Appendix H.

Theorem 18 *Consider a set of data points Ω. For a distance measure $d(\cdot,\cdot)$ that satisfies (D1)–(D4), let*

$$d^*(x,y) = \frac{1}{n}\sum_{z_2\in\Omega} d(z_2,y) + \frac{1}{n}\sum_{z_1\in\Omega} d(x,z_1) - \frac{1}{n^2}\sum_{z_2\in\Omega}\sum_{z_1\in\Omega} d(z_2,z_1) - d(x,y) \tag{23}$$

be the dual cohesion measure of $d(\cdot,\cdot)$. On the other hand, For a cohesion measure $\beta(\cdot,\cdot)$ that satisfies (C1)–(C3), let

$$\beta^*(x,y) = (\beta(x,x)+\beta(y,y))/2 - \beta(x,y) \tag{24}$$

be the dual distance measure of $\beta(\cdot,\cdot)$. Then $d^{**}(x,y) = d(x,y)$ and $\beta^{**}(x,y)=\beta(x,y)$ for all $x,y\in\Omega$.

B. The dual K-sets algorithm

For the K-sets algorithm, we need to have a distance measure. In view of the duality theorem between a cohesion measure and a distance measure, we propose the dual K-sets algorithm in Algorithm 3 that uses a cohesion measure. As before, for a cohesion measure $\gamma(\cdot,\cdot)$ between two points, we define the cohesion measure between two sets S_1 and S_2 as

$$\gamma(S_1,S_2) = \sum_{x\in S_1}\sum_{y\in S_2}\gamma(x,y). \tag{25}$$

Also, note from (16) that the triangular distance from a point x to a set S can be computed by using the cohesion measure as follows:

$$\Delta(x,S) = \gamma(x,x) - \frac{2}{|S|}\gamma(\{x\},S) + \frac{1}{|S|^2}\gamma(S,S). \tag{26}$$

As a direct result of the duality theorem in Theorem 18 and the convergence result of the K-sets algorithm in Theorem 14, we have the following convergence result for the dual K-sets algorithm.

Corollary 19 *As in (13), we define the normalized modularity as $\sum_{k=1}^{K}\frac{1}{|S_k|}\gamma(S_k,S_k)$. For the dual K-sets algorithm, the normalized modularity is increasing when there is a change, i.e., a point is moved from one set to another. Thus, the algorithm converges to a local optimum of the normalized modularity. Moreover, for $K=2$, the dual K-sets algorithm yields two clusters when the algorithm converges.*

ALGORITHM 3: The dual K-sets Algorithm

```
Input: A data set $\Omega = \{x_1, x_2, \ldots, x_n\}$, a cohesion
       measure $\gamma(\cdot,\cdot)$, and the number of sets $K$.
Output: A partition of sets $\{S_1, S_2, \ldots, S_K\}$.
(0) Initially, choose arbitrarily $K$ disjoint nonempty sets
$S_1, \ldots, S_K$ as a partition of $\Omega$.
(1) for $i = 1, 2, \ldots, n$ do
    Compute the triangular distance $\Delta(x_i, S_k)$ for each
    set $S_k$ by using (26).
    Find the set to which the point $x_i$ is closest in terms
    of the triangular distance.
    Assign point $x_i$ to that set.
end
(2) Repeat from (1) until there is no further change.
```

C. Connections to the kernel K-means algorithm

In this section, we show the connection between the dual K-sets algorithm and the kernel K-means algorithm in the literature (see e.g., [61], [62]). In [62], it was shown that a general weighted kernel K-means objective is mathematically equivalent to a weighted graph clustering objective, including the ratio cut, normalized cut, and ratio association criteria.

Let us consider the $n\times n$ matrix $\Gamma = (\gamma_{i,j})$ with $\gamma_{i,j} = \gamma(x_i,x_j)$ being the cohesion measure between x_i and x_j. Call the matrix Γ the cohesion matrix (corresponding to the cohesion measure $\gamma(\cdot,\cdot)$). Since $\gamma(x_i,x_j)=\gamma(x_j,x_i)$, the matrix Γ is symmetric and thus has real eigenvalues λ_k, $k=1,2,\ldots,n$. Let $\mathbf{I}$ be the $n\times n$ identity matrix and $\sigma \ge -\min_{1\le k\le n}\lambda_k$. Then the matrix $\tilde{\Gamma} = \sigma\mathbf{I}+\Gamma$ is positive semi-definite as its n eigenvalues $\tilde{\lambda}_k = \sigma+\lambda_k$, $k=1,2,\ldots,N$ are all nonnegative. Let $v_k = (v_{k,1},v_{k,2},\ldots,v_{k,n})^T$, $k=1,2,\ldots,n$ be the eigenvector of Γ corresponding to the eigenvalue λ_k. Then v_k is also the eigenvector of $\tilde{\Gamma}$ corresponding to the eigenvalue $\tilde{\lambda}_k$. Thus, we can decompose the matrix $\tilde{\Gamma}$ as follows:

$$\tilde{\Gamma} = \sum_{k=1}^{n}\tilde{\lambda}_k v_k v_k^T, \tag{27}$$

where v_k^T is the transpose of v_k. Now we choose the mapping $\phi : \Omega \mapsto \mathcal{R}^n$ as follows:

$$\phi(x_i) = \left(\sqrt{\tilde{\lambda}_1}v_{1,i}, \sqrt{\tilde{\lambda}_2}v_{2,i}, \ldots, \sqrt{\tilde{\lambda}_n}v_{n,i}\right)^T, \tag{28}$$

for $i = 1,2\ldots,n$. Note that

$$\begin{aligned}\phi(x_i)^T\cdot\phi(x_j) &= \sum_{k=1}^{n}\tilde{\lambda}_k v_{k,i}v_{k,j} \\ &= (\tilde{\Gamma})_{i,j} = \sigma\delta_{i,j} + \gamma(x_i,x_j), \end{aligned}\tag{29}$$

where $\delta_{i,j}=1$ if $i=j$ and 0 otherwise.

The "centroid" of a set S can be represented by the corresponding centroid in $\mathcal{R}^n$, i.e.,

$$\frac{1}{|S|}\sum_{y\in S}\phi(y), \tag{30}$$

and the square of the "distance" between a point x and the "centroid" of a set S is

$$\Big(\phi(x) - \frac{1}{|S|}\sum_{y\in S}\phi(y)\Big)^T \cdot \Big(\phi(x) - \frac{1}{|S|}\sum_{y\in S}\phi(y)\Big)$$
$$= (1 - \frac{2}{|S|}1_{\{x\in S\}} + \frac{1}{|S|})\sigma + \gamma(x,x) - \frac{2}{|S|}\gamma(\{x\},S) + \frac{1}{|S|^2}\gamma(S,S), \quad (31)$$

where $1_{\{x\in S\}}$ is the indicator function that has value 1 if x is in S and 0 otherwise. In view of (16), we then have

$$\Big(\phi(x) - \frac{1}{|S|}\sum_{y\in S}\phi(y)\Big)^T \cdot \Big(\phi(x) - \frac{1}{|S|}\sum_{y\in S}\phi(y)\Big)$$
$$= (1 - \frac{2}{|S|}1_{\{x\in S\}} + \frac{1}{|S|})\sigma + \Delta(x,S), \quad (32)$$

where $\Delta(x,S)$ is the triangular distance from a point x to a set S. Thus, the square of the "distance" between a point x and the "centroid" of a set S is $(1 - \frac{1}{|S|})\sigma + \Delta(x,S)$ for a point $x \in S$ and $(1 + \frac{1}{|S|})\sigma + \Delta(x,S)$ for a point $x \notin S$. In particular, when $\sigma = 0$, the dual K-sets algorithm is the same as the sequential kernel K-means algorithm for the kernel $\tilde{\Gamma}$. Unfortunately, the matrix $\tilde{\Gamma}$ may not be positive semi-definite if σ is chosen to be 0. As indicated in [61], a large σ decreases (resp. increases) the distance from a point x to a set S that contains (resp. does not contain) that point. As such, a point is more unlikely to move from one set to another set and the kernel K-means algorithm is thus more likely to be trapped in a local optimum.

To summarize, the dual K-sets algorithm operates in the same way as a sequential version of the classical kernel K-means algorithm by viewing the matrix Γ as a kernel. However, there are two key differences between the dual K-sets algorithm and the classical kernel K-means algorithm: (i) the dual K-sets algorithm guarantees the convergence even though the matrix Γ from a cohesion measure is not positive semi-definite, and (ii) the dual K-sets algorithm can only be operated *sequentially* and the kernel K-means algorithm can be operated in batches. To further illustrate the difference between these two algorithms, we show in the following two examples that a cohesion matrix may not be positive semi-definite and a positive semi-definite matrix may not be a cohesion matrix.

Example 20 In this example, we show there is a cohesion matrix Γ that is not a positive semi-definite matrix.

$$\Gamma = \begin{pmatrix} 0.44 & 0.04 & 0.04 & 0.04 & -0.56 \\ 0.04 & 0.64 & -0.36 & -0.36 & 0.04 \\ 0.04 & -0.36 & 0.64 & -0.36 & 0.04 \\ 0.04 & -0.36 & -0.36 & 0.64 & 0.04 \\ -0.56 & 0.04 & 0.04 & 0.04 & 0.44 \end{pmatrix}. \quad (33)$$

The eigenvalues of this matrix are -0.2, 0, 1, 1, and 1.

Example 21 In this example, we show there is a positive semi-definite matrix $M = (m_{i,j})$ that is not an cohesion matrix.

$$M = \begin{pmatrix} 0.375 & -0.025 & -0.325 & -0.025 \\ -0.025 & 0.875 & -0.025 & -0.825 \\ -0.325 & -0.025 & 0.375 & -0.025 \\ -0.025 & -0.825 & -0.025 & 0.875 \end{pmatrix}. \quad (34)$$

The eigenvalues of this matrix are 0, 0.1, 0.7, and 1.7. Even though the matrix M is symmetric and has all its row sums and column sums being 0, it is still not a cohesion matrix as $m_{1,1} - m_{1,2} - m_{1,4} + m_{2,4} = -0.4 < 0$.

D. Constructing a cohesion measure from a similarity measure

A similarity measure is in general defined as a bivariate function of two *distinct* data points and it is often characterized by a square matrix without specifying the diagonal elements. In the following, we show how one can construct a cohesion measure from a symmetric bivariate function by further specifying the diagonal elements. Its proof is given in Appendix I.

Proposition 22 *Suppose a bivariate function $\beta_0 : \Omega\times\Omega \mapsto \mathcal{R}$ is symmetric, i.e., $\beta_0(x,y) = \beta_0(y,x)$. Let $\beta_1(x,y) = \beta_0(x,y)$ for all $x \neq y$ and specify $\beta_1(x,x)$ such that*

$$\beta_1(x,x) \geq \max_{x\neq y\neq z}[\beta_1(x,z) + \beta_1(x,y) - \beta_1(y,z)]. \quad (35)$$

Also, let

$$\beta(x,y) = \beta_1(x,y) - \frac{1}{n}\sum_{z_1\in\Omega}\beta_1(z_1,y) - \frac{1}{n}\sum_{z_2\in\Omega}\beta_1(x,z_2) + \frac{1}{n^2}\sum_{z_1\in\Omega}\sum_{z_2\in\Omega}\beta_1(z_1,z_2). \quad (36)$$

Then $\beta(x,y)$ is a cohesion measure for Ω.

We note that one simple choice for specifying $\beta_1(x,x)$ in (35) is to set

$$\beta_1(x,x) = 2\beta_{\max} - \beta_{\min}, \quad (37)$$

where

$$\beta_{\max} = \max_{x\neq y}\beta(x,y), \quad (38)$$

and

$$\beta_{\min} = \min_{x\neq y}\beta(x,y). \quad (39)$$

In particular, if the similarity measure $\beta(x,y)$ only has values 0 and 1 as in the adjacency matrix of a simple undirected graph, then one can simply choose $\beta_1(x,x) = 2$ for all x.

Example 23 (A cohesion measure for a graph) As an illustrating example, suppose $A = (a_{i,j})$ is the $n\times n$ adjacency matrix of a simple undirected graph with $a_{i,j} = 1$ if there is an edge between node i and node j and 0 otherwise. Let $k_i = \sum_{j=1}^n a_{i,j}$ be the degree of node i and $m = \frac{1}{2}\sum_{i=1}^n k_i$ be the total number of edges in the graph. Then one can simply let $\beta_1(i,j) = 2\delta_{i,j} + a_{i,j}$, where $\delta_{i,j} = 1$ if $i = j$ and 0 otherwise. By doing so, we then have the following cohesion measure

$$\beta(i,j) = 2\delta_{i,j} + a_{i,j} - \frac{2+k_i}{n} - \frac{2+k_j}{n} + \frac{2m+2n}{n^2}. \quad (40)$$

We note that such a cohesion measure is known as the deviation to indetermination null model in [63].

VI. Conclusions

In this paper, we developed a mathematical theory for clustering in metric spaces based on distance measures and cohesion measures. A cluster is defined as a set of data points that are cohesive to themselves. The hierarchical agglomerative algorithm in Algorithm 1 was shown to converge with a partition of clusters. Our hierarchical agglomerative algorithm differs from a standard hierarchical agglomerative algorithm in two aspects: (i) there is a stopping criterion for our algorithm, and (ii) there is no need to use the greedy selection. We also proposed the K-sets algorithm in Algorithm 2 based on the concept of triangular distance. Such an algorithm appears to be new. Unlike the Lloyd iteration, it only takes one-step minimization in each iteration and that might give the K-sets algorithm the computational advantage over the K-medoids algorithms. The K-sets algorithm was shown to converge with a partition of two clusters when $K = 2$. Another interesting finding of the paper is the duality result between a distance measure and a cohesion measure. As such, one can perform clustering either by a distance measure or a cohesion measure. In particular, the dual K-sets algorithm in Algorithm 3 converges in the same way as a sequential version of the kernel K-means algorithm without the need for the cohesion matrix to *positive semi-definite*.

There are several possible extensions for our work:

(i) *Asymmetric distance measure*: One possible extension is to remove the symmetric property in (D3) for a distance measure. Our preliminary result shows that one only needs $d(x,x) = 0$ in (D2) and the triangular inequality in (D4) for the K-sets algorithm to converge. The key insight for this is that one can replace the original distance measure $d(x,y)$ by a new distance measure $\tilde{d}(x,y) = d(x,y) + d(y,x)$. By doing so, the new distance measure is symmetric.

(ii) *Distance measure without the triangular inequality*: Another possible extension is to remove the triangular inequality in (D4). However, the K-sets algorithm does not work properly in this setting as the triangular distance is no longer nonnegative. In order for the K-sets algorithm to converge, our preliminary result shows that one can adjust the value of the triangular distance based on a weaker notion of cohesion measure. Results along this line will be reported separately.

(iii) *Performance guarantee*: Like the K-means algorithm, the output of the K-sets algorithm also depends on the initial partition. It would be of interest to see if it is possible to derive performance guarantee for the K-sets algorithm (or the optimization problem for the normalized modularity). In particular, the approach by approximation stability in [21] might be applicable as their threshold graph lemma seems to hold when one replaces the distance from a point x to its center c, i.e., $d(x,c)$, by the average distance of a point x to its set, i.e., $\bar{d}(x,S)$.

(iv) *Local clustering*: The problem of local clustering is to find a cluster that contains a specific point x. Since we already define what a cluster is, we may use the hierarchical agglomerative algorithm in Algorithm 1 to find a cluster that contains x. One potential problem of such an approach is the output cluster might be very big. Analogous to the concept of community strength in [34], it would be of interest to define a concept of *cluster strength* and stop the agglomerative process when the desired cluster strength can no longer be met.

(v) *Reduction of computational complexity*: Note that the computation complexity for each iteration within the FOR loop of the K-sets algorithm is $O(Kn^2)$ as it takes $O(Kn)$ steps to compute the triangular distance for each point and there are n points that need to be assigned in each iteration. To further reduce the computational complexity for such an algorithm, one might exploit the idea of "sparsity" and this can be done by the transformation of distance measure.

Acknowledgement

This work was supported in part by the Excellent Research Projects of National Taiwan University, under Grant Number AE00-00-04, and in part by National Science Council (NSC), Taiwan, under Grant Numbers NSC102-2221-E-002-014-MY2 and 102-2221-E-007-006-MY3.

References

[1] Sergios Theodoridis and Konstantinos Koutroumbas. *Pattern Recognition*. Elsevier Academic press, USA, 2006.

[2] Anand Rajaraman, Jure Leskovec, and Jeffrey David Ullman. *Mining of massive datasets*. Cambridge University Press, 2012.

[3] Anil K Jain, M Narasimha Murty, and Patrick J Flynn. Data clustering: a review. *ACM computing surveys (CSUR)*, 31(3):264–323, 1999.

[4] Anil K Jain. Data clustering: 50 years beyond k-means. *Pattern Recognition Letters*, 31(8):651–666, 2010.

[5] Stuart Lloyd. Least squares quantization in pcm. *Information Theory, IEEE Transactions on*, 28(2):129–137, 1982.

[6] Leonard Kaufman and Peter J Rousseeuw. *Finding groups in data: an introduction to cluster analysis*, volume 344. John Wiley & Sons, 2009.

[7] Mark Van der Laan, Katherine Pollard, and Jennifer Bryan. A new partitioning around medoids algorithm. *Journal of Statistical Computation and Simulation*, 73(8):575–584, 2003.

[8] Hae-Sang Park and Chi-Hyuck Jun. A simple and fast algorithm for k-medoids clustering. *Expert Systems with Applications*, 36(2):3336–3341, 2009.

[9] Sariel Har-Peled and Bardia Sadri. How fast is the k-means method? *Algorithmica*, 41(3):185–202, 2005.

[10] David Arthur and Sergei Vassilvitskii. k-means++: The advantages of careful seeding. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 1027–1035. Society for Industrial and Applied Mathematics, 2007.

[11] Bahman Bahmani, Benjamin Moseley, Andrea Vattani, Ravi Kumar, and Sergei Vassilvitskii. Scalable k-means++. *Proceedings of the VLDB Endowment*, 5(7):622–633, 2012.

[12] Ragesh Jaiswal and Nitin Garg. Analysis of k-means++ for separable data. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 591–602. Springer, 2012.

[13] Manu Agarwal, Ragesh Jaiswal, and Arindam Pal. k-means++ under approximation stability. In *Theory and Applications of Models of Computation*, pages 84–95. Springer, 2013.

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

[15] Asa Ben-Hur, David Horn, Hava T Siegelmann, and Vladimir Vapnik. Support vector clustering. *The Journal of Machine Learning Research*, 2:125–137, 2002.

[16] Chris Ding and Xiaofeng He. K-means clustering via principal component analysis. In *Proceedings of the twenty-first international conference on Machine learning*, page 29. ACM, 2004.

[17] Francesco Camastra and Alessandro Verri. A novel kernel method for clustering. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 27(5):801–805, 2005.

[18] Wen-Yen Chen, Yangqiu Song, Hongjie Bai, Chih-Jen Lin, and Edward Y Chang. Parallel spectral clustering in distributed systems. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 33(3):568–586, 2011.

[19] Ulrike Von Luxburg. A tutorial on spectral clustering. *Statistics and computing*, 17(4):395–416, 2007.

[20] Maurizio Filippone, Francesco Camastra, Francesco Masulli, and Stefano Rovetta. A survey of kernel and spectral methods for clustering. *Pattern recognition*, 41(1):176–190, 2008.

[21] Maria-Florina Balcan, Avrim Blum, and Anupam Gupta. Clustering under approximation stability. *Journal of the ACM (JACM)*, 60(2):8, 2013.

[22] Ulrike Von Luxburg and Shai Ben-David. Towards a statistical theory of clustering. In *Pascal workshop on statistics and optimization of clustering*. Citeseer, 2005.

[23] Nicholas Jardine and Robin Sibson. Mathematical taxonomy. *London etc.: John Wiley*, 1971.

[24] William E Wright. A formalization of cluster analysis. *Pattern Recognition*, 5(3):273–282, 1973.

[25] Jan Puzicha, Thomas Hofmann, and Joachim M Buhmann. A theory of proximity based clustering: Structure detection by optimization. *Pattern Recognition*, 33(4):617–634, 2000.

[26] Jon Kleinberg. An impossibility theorem for clustering. *Advances in neural information processing systems*, pages 463–470, 2003.

[27] Reza Bosagh Zadeh and Shai Ben-David. A uniqueness theorem for clustering. In *Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence*, pages 639–646. AUAI Press, 2009.

[28] Gunnar Carlsson, Facundo Mémoli, Alejandro Ribeiro, and Santiago Segarra. Hierarchical quasi-clustering methods for asymmetric networks. *arXiv preprint arXiv:1404.4655*, 2014.

[29] Shai Ben-David and Margareta Ackerman. Measures of clustering quality: A working set of axioms for clustering. In *Advances in neural information processing systems*, pages 121–128, 2009.

[30] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Kdd*, volume 96, pages 226–231, 1996.

[31] Antonio Cuevas, Manuel Febrero, and Ricardo Fraiman. Cluster analysis: a further approach based on density estimation. *Computational Statistics & Data Analysis*, 36(4):441–459, 2001.

[32] Maria Halkidi and Michalis Vazirgiannis. A density-based cluster validity approach using multi-representatives. *Pattern Recognition Letters*, 29(6):773–786, 2008.

[33] David Liben-Nowell and Jon Kleinberg. The link prediction problem for social networks. In *Proceedings of the twelfth international conference on Information and knowledge management*, pages 556–559. ACM, 2003.

[34] Cheng-Shang Chang, Chih-Jung Chang, Wen-Ting Hsieh, Duan-Shin Lee, Li-Heng Liou, and Wanjiun Liao. Relative centrality and local community detection. *Network Science*, FirstView:1–35, 8 2015.

[35] Ulrik Brandes, Daniel Delling, Marco Gaertler, Robert Gorke, Martin Hoefer, Zoran Nikoloski, and Dorothea Wagner. On modularity clustering. *Knowledge and Data Engineering, IEEE Transactions on*, 20(2):172–188, 2008.

[36] Mark EJ Newman. Fast algorithm for detecting community structure in networks. *Physical review E*, 69(6):066133, 2004.

[37] Mark EJ Newman and Michelle Girvan. Finding and evaluating community structure in networks. *Physical review E*, 69(2):026113, 2004.

[38] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10):P10008, 2008.

[39] Renaud Lambiotte. Multi-scale modularity in complex networks. In *Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks (WiOpt), 2010 Proceedings of the 8th International Symposium on*, pages 546–553. IEEE, 2010.

[40] J-C Delvenne, Sophia N Yaliraki, and Mauricio Barahona. Stability of graph communities across time scales. *Proceedings of the National Academy of Sciences*, 107(29):12755–12760, 2010.

[41] Cheng-Shang Chang, Chin-Yi Hsu, Jay Cheng, and Duan-Shin Lee. A general probabilistic framework for detecting community structure in networks. In *IEEE INFOCOM '11*, April 2011.

[42] Wayne W Zachary. An information flow model for conflict and fission in small groups. *Journal of anthropological research*, pages 452–473, 1977.

[43] Stella X Yu and Jianbo Shi. Multiclass spectral clustering. In *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, pages 313–319. IEEE, 2003.

[44] Andrew Y Ng, Michael I Jordan, Yair Weiss, et al. On spectral clustering: Analysis and an algorithm. *Advances in neural information processing systems*, 2:849–856, 2002.

[45] Ronald R Coifman, Stephane Lafon, Ann B Lee, Mauro Maggioni, Boaz Nadler, Frederick Warner, and Steven W Zucker. Geometric diffusions as a tool for harmonic analysis and structure definition of data: Diffusion maps. *Proceedings of the National Academy of Sciences of the United States of America*, 102(21):7426–7431, 2005.

[46] P. Erdös and A. Rényi. On random graphs. *Publ. Math. Debrecen*, 6:290–297, 1959.

[47] Alaa Saade, Florent Krzakala, and Lenka Zdeborová. Spectral clustering of graphs with the bethe hessian. In *Advances in Neural Information Processing Systems*, pages 406–414, 2014.

[48] Lenka Zdeborova Aurelien Decelle, Florent Krzakala and Pan Zhang. Mode-net: Modules detection in networks, 2012.

[49] Laurent Massoulié. Community detection thresholds and the weak ramanujan property. In *Proceedings of the 46th Annual ACM Symposium on Theory of Computing*, pages 694–703. ACM, 2014.

[50] Elchanan Mossel, Joe Neeman, and Allan Sly. A proof of the block model threshold conjecture. *arXiv preprint arXiv:1311.4115*, 2013.

[51] Andrea Lancichinetti, Filippo Radicchi, José J Ramasco, Santo Fortunato, et al. Finding statistically significant communities in networks. *PloS one*, 6(4):e18961, 2011.

[52] M. Rosvall and C. T. Bergstrom. Maps of information flow reveal community structure in complex networks. Technical report, Citeseer, 2007.

[53] Martin Rosvall, Daniel Axelsson, and Carl T Bergstrom. The map equation. *The European Physical Journal Special Topics*, 178(1):13–23, 2010.

[54] Stefan Van Der Walt, S Chris Colbert, and Gael Varoquaux. The numpy array: a structure for efficient numerical computation. *Computing in Science & Engineering*, 13(2):22–30, 2011.

[55] Ravindra B Bapat, Ivan Gutmana, and Wenjun Xiao. A simple method for computing resistance distance. *Zeitschrift für Naturforschung A*, 58(9-10):494–498, 2003.

[56] Gabor Csardi and Tamas Nepusz. The igraph software package for complex network research. *InterJournal*, Complex Systems:1695, 2006.

[57] Florent Krzakala, Cristopher Moore, Elchanan Mossel, Joe Neeman, Allan Sly, Lenka Zdeborová, and Pan Zhang. Spectral redemption in clustering sparse networks. *Proceedings of the National Academy of Sciences*, 110(52):20935–20940, 2013.

[58] Yann LeCun and Corinna Cortes. MNIST handwritten digit database. http://yann.lecun.com/exdb/mnist/, 2010.

[59] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[60] Kyle A Beauchamp, Gregory R Bowman, Thomas J Lane, Lutz Maibaum, Imran S Haque, and Vijay S Pande. Msmbuilder2: Modeling conformational dynamics on the picosecond to millisecond scale. *Journal of chemical theory and computation*, 7(10):3412–3419, 2011.

[61] Inderjit Dhillon, Yuqiang Guan, and Brian Kulis. *A unified view of kernel k-means, spectral clustering and graph cuts*. Computer Science Department, University of Texas at Austin, 2004.

[62] Inderjit S Dhillon, Yuqiang Guan, and Brian Kulis. Weighted graph cuts without eigenvectors a multilevel approach. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 29(11):1944–1957, 2007.

[63] Romain Campigotto, Patricia Conde Céspedes, and Jean-Loup Guillaume. A generalized and adaptive method for community detection. *arXiv preprint arXiv:1406.2518*, 2014.