A MinMaxCut Spectral Method for Data Clustering and Graph Partitioning

Chris Ding[a], Xiaofeng He[a], Hongyuan Zha[b], Ming Gu[c], Horst D. Simon[a]
[a]Lawrence Berkeley National Laboratory
University of California, Berkeley, CA 94720
[b] Department of Computer Science and Engineering
Pennsylvania State University, University Park, PA 16802
[c] Department of Mathematics
University of California, Berkeley, CA 94720
{chqding,xhe,hdsimon}@lbl.gov, zha@cse.psu.edu, mgu@math.berkeley.edu

LBNL Tech Report #54111. December 1, 2003

Abstract

The goal of data clustering can be formally stated as a min-max clustering principle: data points are grouped into clusters such that (a) between-cluster similarities are minimized while (b) within-cluster similarities are maximized. Existing methods typically focus on one of the above requirements. Here we propose a new method that emphasizes both requirements simultaneously. The `MinMaxCut` method uses pairwise similarities between all data objects. Viewing them as the adjacency matrix of a weighted graph, `MinMaxCut` is a graph clustering method. An important feature is that the continuous solution of the cluster membership indicator vector is the eigenvector of the generalized graph Laplacian matrix; this simplifies the optimization of the min-max clustering objective function. We provide a comprehensive analysis of this principled clustering method; a systematic evaluation on clustering experiments on Internet newsgroups indicates that `MinMaxCut` outperforms other current popular partitioning/clustering methods.

1 Introduction

Data clustering, also called cluster analysis, is an active research area with a long history (see [21, 25, 14, 23]), from the K-means methods [27, 28, 22] and hierarchical clustering algorithms in 1960s-1970s to the present-day more complex methods such as those based on Gaussian mixtures [30] and other model-based methods[1], graph partitioning methods [33, 19, 34, 31], and methods developed by the database community (see the summary in [20]).

The essential task of data clustering is partitioning data points into disjoint clusters so that (P1) objects in the same cluster are similar, (P2) objects in different clusters are dissimilar. When data objects distribute as compact clumps which are well separated, clusters are well defined and we refer to those well-defined clusters as natural clusters, When the clumps are not compact or when clumps overlap with each other, clusters are not well defined; a clear and meaningful definition of clusters then becomes crucial.

Existing clustering methods typically attempt to satisfy one of the two requirements above. K-means algorithm, for example, attempts to ensure that data point in the same cluster are similar which is (P1), while the graph partitioning methods `RatioCut` and `NormalizedCut` attempt to ensure that objects in different cluster are maximally different, which is (P2).

In this paper we introduce `MinMaxCut` , a graph partitioning based clustering algorithm which incorporate (P1) and (P2) simultaneously. We formally state them as the following min-max clustering principle: data should be grouped into clusters such that the similarity or association across clusters is minimized, while the similarity or association within each cluster is maximized (see [14, 37] for recent studies of clustering objective functions).

Clustering algorithms such as K-means and those based on Gaussian mixtures require the coordinates/attributes of each object explicitly. Graph partitioning algorithm requires only the pairwise similarities between objects. Given the pairwise similarity $S = (s_{ij})$, where s_{ij} indicates the similarity between objects i and j, we may consider S as the adjacency matrix of a weighted graph G, hence the data clustering problem becomes a graph partition problem. (Splitting a dataset into two is rephrased as *cutting* a graph into two subgraphs; cutting a graph into two very unbalanced subgraphs is referred to as *skewed cut*; the boundary between two subgraphs is sometimes called *cut*.)

We mention that graph partitioning has broad range of applications. Popular software packages for this partitioning task are developed [26, 24]. Graph partitioning differs from graph clustering in that graph partitioning leads to equal-sized partitions while graph clustering quite often leads to clusters of different sizes: Natural clusters do not always have same sizes.

Cluster analysis is applied to large amount of data with a variety of distributions/shapes for the clusters. Using the similarity metric, more complex shaped distributions can be accommodated. For example K-means favors spherically shaped clusters, while hierarchical agglomerative clustering can produce elongate-shaped clusters by using single-linkage. Using similarity-based graph partitioning, the connectivity between objects becomes most important, instead of their shape in a Euclidean space which is very hard to model.

The min-max clustering principle favors the objective function optimization approach, i.e, clusters are obtained by optimizing an appropriate objective function. This is mathematically more principled approach, in contrast to procedure-oriented clustering methods, such as the hierarchical algorithms.

In the following we briefly summarize the results obtained in this paper, which also serve as the outline of the paper. In §2, we discuss the `MinMaxCut` for $K = 2$ case. We first show that the continuous solution of the cluster membership indicator vector is the eigenvector of the generalized Laplacian matrix of the similarity matrix. Related work on spectral graph clustering, the `RatioCut` [19] and the `NormalizedCut` [34] are discussed in §2.1. Using random graph model, we show that `MinMaxCut` tends to produce balanced clusters while earlier methods do not (see §2.2). The cluster balancing power of `MinMaxCut` can be softened or hardened by a slight generalization of the clustering objective function (see §2.3). In §2.4, we define the cohesion of a dataset/graph as the optimal value of `MinMaxCut` objective function when the dataset is split into two. We prove important lower and upper bounds for the cohesion value. Experiments on clustering internet newsgroups are presented in §2.5. which show the advantage of `MinMaxCut` compared with existing methods. In §2.6 we derive the conditions for possible skewed clustering for `MinMaxCut` and `NormalizedCut` which shows the balancing power of `MinMaxCut` . In §2.7, we show the `MinMaxCut` linkage is useful for further refinement of clusters obtained from `MinMaxCut` . The linkage differential ordering can further improve the clustering results (see §2.8). In §2.9, we discuss the clustering of a contingency table which can be viewed as a weighted bipartite graph. The simultaneous clustering of rows and columns of the contingency table can be done in much the same way as in 2-way clustering of §2.

In §3, we discuss `MinMaxCut` for $K > 2$ cases. We show K-way `MinMaxCut` leads to the more refined or subtle cluster balance, the similarity-weighted size balance in §3.1. In §3.2, the importance of K eigenvectors are noted with the generalized lower and upper bounds of optimal value of the objective function. The K-way clustering requires two stages, the initial clustering and refinement. In§3.3, three methods of initial clustering are briefly explained, the eigenspace K-means, the divisive and agglomerative clusterings. The cluster refinement algorithms based on `MinMaxCut` objective functions are outline in §3.4.

In §4, the divisive `MinMaxCut` as a K-way clustering method is explained in detail. We first prove the monotonicity of `MinMaxCut` and K-means objective function w.r.t. to cluster merging and spliting in §4.1. In §4.2, we outline the cluster selection methods: those based on the size-priority, average similarity, cohesion and temporary objectives. Stopping criteria are outlined in §4.3. In §4.4, we discuss objective function saturation, a subtle issue in objective function optimization based approaches. In §4.5, results of comprehensive experiments on newsgroup are presented which show the average similarity as a better cluster selection method. Our results also show the importance of `MinMaxCut` based refinement after the initial clusters are obtained in the divisive clustering. This indicates the appropriateness of the `MinMaxCut` objective function. In §5, summary and discussions results are given. Some preliminary results[10, 8] of this paper were previously represented in conferences.

2 Two-way `MinMaxCut`

Given pairwise similarities or associations for a set of n data objects specified in $S = (s_{ij})$, we wish to cluster the data objects into two clusters A, B based on the following *min-max clustering principle*: data points are grouped into clusters such that between-cluster associations are minimized while within-cluster associations are maximized. The association between A, B is the sum of pairwise associations between the two clusters, $s(A, B) = \sum_{i\in A, j\in B} s_{ij}$. The association within cluster A is $s(A, A) = \sum_{i\in A, j\in A} s_{ij}$; $s(B, B)$ is analogously defined. The min-max clustering principle requires

$$\min s(A, B), \quad \max s(A, A), \quad \max s(B, B). \tag{1}$$

These requirements are simultaneously satisfied by minimizing the objective function [10],

$$J_{\text{MMC}} = \frac{s(A, B)}{s(A, A)} + \frac{s(A, B)}{s(B, B)}. \tag{2}$$

Note that there are many objective functions that satisfy Eq.(1). However, for J_{MMC}, a continuous solutions can be computed efficiently.

Clustering solution can be represented by an indicator vector $\mathbf{q}$,

$$q_i = \begin{cases} a & \text{if } \ i \in A \\ -b & \text{if } \ i \in B \end{cases}, \tag{3}$$

where $a = \sqrt{d_B/d_A d}$, $b = \sqrt{d_A/d_B d}$,

$$d_A = \sum_{i\in A} d_i, \quad d_B = \sum_{i\in B} d_i, \quad d = d_A + d_B. \tag{4}$$

and $d_i = \sum_j s_{ij}$ is the degree of node i. Thus

$$\mathbf{q}^T D\mathbf{e} = 0, \quad \mathbf{q}^T D\mathbf{q} = 1, \tag{5}$$

where $D = \text{diag}(d_1, \cdots, d_n)$. We first prove that

$$\min_{\mathbf{q}} J_{\text{MMC}}(A, B) \implies \max_{\mathbf{q}} J_{\text{m}}(\mathbf{q}), \ J_{\text{m}}(\mathbf{q}) = \frac{\mathbf{q}^T S\mathbf{q}}{\mathbf{q}^T D\mathbf{q}}. \tag{6}$$

Define the indicator vector $\mathbf{x}$, where

$$x_i = (q_i - \frac{a-b}{2})\frac{2}{a+b} = \begin{cases} 1 & \text{if} \quad q_i = a \\ -1 & \text{if} \quad q_i = -b \end{cases}$$

Now

$$s(A,B) = \frac{1}{2}\sum_{ij} \frac{(x_i - x_j)^2}{4} s_{ij} = \frac{1}{2}\sum_{ij} \frac{(q_i - q_j)^2}{(a+b)^2} s_{ij} = \frac{\mathbf{q}^T(D-S)\mathbf{q}}{(a+b)^2}. \tag{7}$$

By definition of $\mathbf{q}$ in Eq.(3), we obtain

$$\mathbf{q}^T S\mathbf{q} = a^2 s(A,A) + b^2 s(B,B) - 2ab s(A,B). \tag{8}$$

The orthogonality condition of Eq.(5) becomes

$$as(A,A) - bs(B,B) + (a-b)s(A,B) = 0. \tag{9}$$

With these relations, after some algebraic manipulation, we obtain

$$J_{\text{MMC}} = \frac{1+a/b}{J_{\text{m}} + a/b} + \frac{1+b/a}{J_{\text{m}} + b/a} - 2. \tag{10}$$

Since $b/a > 0$ is fixed, one can easily see that

$$\frac{dJ_{\text{MMC}}}{dJ_{\text{m}}} = -\frac{1+a/b}{(J_{\text{m}} + a/b)^2} - \frac{1+b/a}{(J_{\text{m}} + b/a)^2} < 0.$$

Hence J_{MMC} is a monotonically decreasing function of J_{m}. This proves Eq.(6). □

Optimization of $J_{\text{m}}(\mathbf{q})$ with the constraint that q_i takes discrete values $\{a, -b\}$ is a hard problem. Following §2.1, we let q_i take arbitrary continuous values in the interval $[-1, 1]$. The optimal solution for the Rayleigh quotient of J_{m} in Eq.(6) is the eigenvector $\mathbf{q}$ associated with the largest eigenvalue of the system

$$S\mathbf{q} = \lambda D\mathbf{q}. \tag{11}$$

Let $\mathbf{q} = D^{-1/2}\mathbf{z}$ and multiply both sides by $D^{-1/2}$; this equation becomes a standard eigenvalue problem:

$$D^{-1/2}WD^{-1/2}\mathbf{z}_k = \lambda_k \mathbf{z}_k,\ \lambda_k = 1 - \zeta_k, \tag{12}$$

The eigenvector with largest eigenvalue $\lambda_1 = 1$ is a constant vector $\mathbf{q}_1 = \mathbf{e}$. Since $\mathbf{q}_k$ satisfies the orthogonality relation $\mathbf{z}_k^T\mathbf{z}_p = \mathbf{q}_k^T D\mathbf{q}_p = 0$ if $k \neq q$, constraints in Eq.(5) are automatically satisfied. The desired solution is $\mathbf{q}_2$. We summarize these results as

Theorem 2.1. Clustering a dataset by optimizing the objective function Eq.(2), the optimal cluster indicators are given by the eigenvector $\mathbf{q}_2$.

From Eq.(3), we can recover cluster membership by sign, i.e., $A = \{i \mid \mathbf{q}_2(i) \leq 0\}$, $B = \{i \mid \mathbf{q}_2(i) \geq 0\}$. In general, the optimal dividing point could shift away from 0; we search the dividing point $i_{cut} = 1, \cdots, n-1$, setting

$$A = \{i \mid \mathbf{q}_2(i) \leq \mathbf{q}_2(i_{cut})\}, \quad B = \{i \mid \mathbf{q}_2(i) > \mathbf{q}_2(i_{cut})\}, \tag{13}$$

such that $J_{\text{MMC}}(A,B)$ is minimized. The corresponding A and B are the final clusters.

The computation of the eigenvectors can be done quickly via the Lanczos method [32]. A software package for this calculation, LANSO, is available online (http://www.nersc.gov/~kewu/planso.html). Overall the computational complexity is $O(n^2)$.

2.1 Related work on spectral graph partition

Spectral graph partitioning is based on the properties of eigenvectors of the Laplacian matrix $L = D - W$, first developed by Donath and Hoffman [11] and Fiedler [16, 17]. The method becomes widely known in high performance computing area by the work of Pothen, Simon and Liu [33]. The objective of the partitioning is to minimize the `MinCut` objective, i.e, the cutsize (the between-cluster similarity) $J_{\rm cut}(A,B) = s(A,B)$ with the requirement that two subgraphs have the same number of nodes: $|A| = |B|$. Using indicator variable x_u, $x_u = \{1, -1\}$ depending on $u \in A$ or B, the cutsize is

$$s(A,B) = \sum_{e_{uv} \in E} \frac{(x_u - x_v)^2}{4} s_{uv} = \frac{\mathbf{x}^T (D-S)\mathbf{x}}{2}. \tag{14}$$

Relax x_u from $\{1,-1\}$ to continuous value in $[-1,1]$, minimizing $s(A,B)$ is equivalent to solve the eigensystem

$$(D-S)\mathbf{x} = \zeta \mathbf{x}. \tag{15}$$

Since the trivial $\mathbf{x}_1 = \mathbf{e}$ is associated with $\lambda_1 = 0$, the second eigenvector $\mathbf{x}_2$, also called Fiedler vector, is the solution. Hagen and Kahng[19] remove the requirement $|A| = |B|$ and show that the $\mathbf{x}_2$ provides the continuous solution of the cluster indicator vector for the `RatioCut` objective function[5]

$$J_{\rm rcut} = \frac{s(A,B)}{|A|} + \frac{s(A,B)}{|B|}. \tag{16}$$

The generalized eigensystem of Eq.(15) is

$$(D-S)\mathbf{x} = \zeta D \mathbf{x}, \tag{17}$$

which is identical to Eq.(11) with $\lambda = 1 - \zeta$. The use of this equation has been studied by a number of authors [12, 6, 34]. Chung [6] emphasizes the advantage of using normalized Laplacian matrix which leads to Eq.(17). Shi and Malik [34] propose the `NormalizedCut` ,

$$J_{\rm ncut} = \frac{s(A,B)}{d_A} + \frac{s(A,B)}{d_B} \tag{18}$$

where d_A, d_B defined in Eq.(4) are also called the *volumes* [6] of subgraphs A, B, in contrast to the sizes of A, B. A key observation is that $J_{\rm ncut}$ can be written as

$$J_{\rm ncut} = \frac{s(A,B)}{s(A,A) + s(A,B)} + \frac{s(A,B)}{s(B,B) + s(A,B)}, \tag{19}$$

since

$$d_A \equiv \sum_{i \in A} d_i = \sum_{i \in A,\ j \in G} s_{ij} = \sum_{i \in A} \left(\sum_{j \in A} + \sum_{j \in B} \right) s_{ij} = s(A,A) + s(A,B).$$

and $d_B = s(B,B) + s(A,B)$. The presence of $s(A,B)$ in the denominators of $J_{\rm ncut}$ indicate it is not conformal to the min-max clustering principle. In practical applications, `NormalizedCut` sometimes leads to unbalanced clusters (see §2.5). `MinMaxCut` is developed by being conformal to the min-max clustering principle. In extensive experiments (see §2.5), `MinMaxCut` consistently outperforms `NormalizedCut` and `RatioCut` .

`RatioCut` , `NormalizedCut` and `MinMaxCut` objective functions are first *prescribed* by proper motivating considerations and then $\mathbf{q}_2$ is shown to be the continuous solution of the cluster indicator vectors. It should be noted that in a perturbation analysis on the case where clusters are well separated (thus clearly defined) [9], the same three objective functions can be automatically recovered as the second principal eigenvalue of the corresponding (normalized) Laplacian matrix and indicator vector of Eq.3 are recovered. This further strengthens the connection between clustering objective functions and the Laplacian matrix of a graph.

Besides Laplacian matrix based spectral partitioning methods, other recent partitioning methods use singular value decompositions[2, 13].

2.2 Cluster balance: random graph model analysis

One important feature of the `MinMaxCut` method is that it tends to produce balanced clusters, i.e., the resulting subgraphs have similar sizes. Here we use the random graph model [5, 3] to illustrate this point. Suppose we have a uniformly distributed random graph with n nodes. For this random graph, any two nodes are connected with probability $p, 0 \le p \le 1$. We consider the four objective functions, the `MinCut` , `RatioCut` , `NormalizedCut` and `MinMaxCut` (see §2.1). We have the following

> **Theorem 2.2**. For random graphs, `MinCut` favors highly skewed cuts. `MinMaxCut` favors balanced cut, i.e., both subgraphs have the same size. `RatioCut` and `NormalizedCut` show no size preferences, i.e., each subgraph could have arbitrary size.

Proof. We compute the object functions for the partition of G into A and B. Note that the number of edges between A and B are $p|A||B|$ on average. For `MinCut` , we have

$$J_{\text{mincut}}(A, B) = p|A||B|$$

For `RatioCut` , we have

$$J_{\text{rcut}}(A, B) = \frac{p|A||B|}{|A|} + \frac{p|A||B|}{|B|} = p(|A| + |B|) = np.$$

For `NormalizedCut` , since all nodes have the same degree $(n-1)p$,

$$J_{\text{ncut}}(A, B) = \frac{p|A||B|}{p|A|(n-1)} + \frac{p|A||B|}{p|B|(n-1)} = \frac{n}{n-1}.$$

For `MinMaxCut` , we have

$$J_{\text{MMC}}(A, B) = \frac{|B|}{|A|-1} + \frac{|A|}{|B|-1}.$$

We now minimize these objectives. Clearly, `MinCut` favors either $|A| = n-1$,$|B| = 1$ or $|B| = n-1$,$|A| = 1$, both are skewed cuts. Minimizing $J_{\text{MMC}}(A, B)$, we obtain a balanced cut: $|A| = |B| = n/2$:

$$\min_{A,B} J_{\text{MMC}}(A, B) = \frac{2}{1 - 2/n}. \tag{20}$$

Both Rcut and Ncut objectives have no size dependency and no size preference. □

2.3 Soft and hard MinMaxCut

Clearly `MinMaxCut` has a strong tendency to produce balanced clusters. Although balanced clusters are desirable, sometimes naturally occurring clusters are not necessarily balanced. Here we introduce a generalized `MinMaxCut` that has varying degree of cluster balancing. We define the generalized clustering objective function

$$J^{(\alpha)}_{\text{MMC}} = \left[\frac{s(A, B)}{s(A, A)}\right]^{\alpha} + \left[\frac{s(A, B)}{s(B, B)}\right]^{\alpha} \tag{21}$$

for any fixed parameter $\alpha > 0$.

The important property of $J^{(\alpha)}_{\text{MMC}}$ is that the procedure for computing the clusters remains identical to $\alpha = 1$ case in §2.1, because minimization of $J^{(\alpha)}_{\text{MMC}}$ leads to the same problem of $\max J_{\text{m}}(\mathbf{q})$, i.e.,

$$\min_{\mathbf{q}} J^{(\alpha)}_{\text{MMC}}(A,B) \implies \max_{\mathbf{q}} J_{\text{m}}(\mathbf{q}),$$

for any $\alpha > 0$; this can be proved by repeating the proof of Eq.(6).

The generalized `MinMaxCut` for any $\alpha > 0$ still retains the cluster balancing property as one can easily show that Theorem 2.2 regarding cluster balancing on random graphs remains valid. However, the level of balancing depends on α.

If $\alpha > 1$, $J^{(\alpha)}_{\text{MMC}}$ will have stronger cluster balancing than $J^{(\alpha=1)}_{\text{MMC}}$, because the larger of the two terms

$$\frac{s(A,B)}{s(A,A)}, \quad \frac{s(A,B)}{s(B,B)} \tag{22}$$

will dominate $J^{(\alpha)}_{\text{MMC}}$ more, and thus $\min J^{(\alpha>1)}_{\text{MMC}}$ will more strongly force the two terms to be equal. We call this case the *hard* `MinMaxCut` . In particular, for $\alpha \gg 1$, we have

$$J^{(\alpha\gg1)}_{\text{MMC}} \simeq \left[\max\left(\frac{s(A,B)}{s(A,A)}, \frac{s(A,B)}{s(B,B)}\right)\right]^{\alpha}$$

thus

$$\min_{\mathbf{q}} J^{(\alpha\gg1)}_{\text{MMC}} \Longrightarrow \min_{\mathbf{q}} \max\left(\frac{s(A,B)}{s(A,A)}, \frac{s(A,B)}{s(B,B)}\right). \tag{23}$$

We call this case the "minimax cut". Minimax-cut ignores the details of the smaller term and therefore is less sensitive than $J^{(\alpha=1)}_{\text{MMC}}$.

If $\alpha < 1$, $J^{(\alpha)}_{\text{MMC}}$ will have weeker cluster balancing. This case is more applicable for datasets where natural clusters are of different sizes. Here $1/2 \le \alpha < 1$ are good choices. We call this case the *soft* `MinMaxCut` .

2.4 Cluster cohesion and bounds

Given a dataset of n objects and their pairwise similarity $S = (s_{ij})$, we may partition them into two subsets in many different ways with different values of J_{MMC}. However, the optimal J_{MMC} value

$$h(S) \equiv J^{\text{opt}}_{\text{MMC}}(S) = \min_{\mathbf{q}} J_{\text{MMC}}(\mathbf{q}; S)$$

is a well-defined quantity, although its exact value may not be easily computed.

Definition. Cluster cohesion of a dataset is the smallest value of the MinMaxCut objective function when the dataset is split into two clusters.

Cluster cohesion is a good characterization of a dataset against splitting it into two clusters. Suppose we apply `MinMaxCut` to split a dataset into two clusters. If $J^{\text{opt}}_{\text{MMC}}$ thus obtained is large, this indicates the overlap between the two resulting clusters is large in comparison to the within-cluster similarity, and thus the dataset is likely a single natural cluster and should not be split.

On the other hand, if $J^{\text{opt}}_{\text{MMC}}(S)$ is small, the overlap between the two resulting clusters is small, i.e., two clusters are well-separated, which indicates that the dataset should be splited. Thus $J^{\text{opt}}_{\text{MMC}}$ is a good indicator of cohesion of the dataset with respect to clustering. For this reason, $J^{\text{opt}}_{\text{MMC}}(S)$ is called *cluster cohesion* and is denoted as $h(S)$.

Note that h is similar to Cheeger constant [6] h_1 in graph theory, which is defined as

$$h_1 = \min_{\mathbf{q}} \frac{s(A,B)}{\min(d_A, d_B)} = \min_{\mathbf{q}} \max\left(\frac{s(A,B)}{d_A}, \frac{s(A,B)}{d_B}\right)$$

From Eq.(19), one can see that `NormalizedCut` is a generalization of Cheeger constant, i.e., both terms are retained in the optimization of `NormalizedCut` . Using the analogy of the minimax version of `MinMaxCut` via Eq.(23), we may also say that h_1 is the minimax version of `NormalizedCut` . Since S can be viewed as the adjacency matrix of a graph G, we call h the cohesion of graph G.

For all possible graphs one may expect the cohesion value to have a large range and thus difficult to gauge. Surprisingly, cohesion for an arbitrarily weighted graph is restricted to a narrow range, as we can prove the following :

Theorem 2.4. (a) The largest cohesion value of all possible graphs (similarity matrices S) is

$$\max_S h(S) = \frac{2}{1-2/n} \tag{24}$$

(b) the cohesion of a graph has the bound

$$\frac{4}{1+\lambda_2} - 2 \le h \le \frac{2}{1-2/n} \tag{25}$$

where λ_2 is from Eq.(12).

Proof. Part (a) can be proved by the following two lemmas regarding graphs. Lemma (L1): The unweighted complete graph (clique) has the cohesion of Eq.(24), same as the random graph with $p = 1$ (see Eq.(20)). Lemma (L2): All graphs, both weighted and unweighted, have cohesion smaller than that of the complete graph. L2 is very intuitive and can be proved rigorously by starting with a clique and removing edges. Details are skipped here. Part (b) is proved by considering $J_{\text{MMC}}(J_{\text{m}}, a/b)$ as a function of a/b and J_{m}. It can be shown that

$$J_{\text{MMC}}(J_{\text{m}}, \frac{a}{b}) \ge \min_{a/b} J_{\text{MMC}}(J_{\text{m}}, \frac{a}{b}) = \frac{4}{1+J_{\text{m}}} - 2 \ge \frac{4}{1+\lambda_2} - 2 \tag{26}$$

The last inequality follows from

$$J_{\text{m}}(\mathbf{q}) = \frac{\mathbf{q}^T S \mathbf{q}}{\mathbf{q}^T D \mathbf{q}} \le \max_{\mathbf{q}} \frac{\mathbf{q}^T S \mathbf{q}}{\mathbf{q}^T D \mathbf{q}} = \lambda_2.$$

□

Theorem 2.4 establishes cluster cohesion $J_{\text{MMC}}^{\text{opt}}$ as a useful quantity to characterize a dataset with the chosen similarity metric. The upper bound is useful for checking whether a partition of the dataset is within the right range.

2.5 Internet newsgroups clustering experiments

Document clustering has been popular in analyzing text information. Here we perform experiments on newsgroup articles in 20 newsgroups (dataset available online[29]). We focus on three datasets, each has two newsgroups:

NG1/NG2	NG10/NG11	NG18/NG19
NG1: alt.atheism	NG10: rec.sport.baseball	NG18: talk.politics.mideast
NG2: comp.graphics	NG11: rec.sport.hockey	NG19: talk.politics.misc

Word-document matrix $X = (x_1, \cdots, x_n)$ is first constructed. 2000 words are selected according to the mutual information between words and documents

$$I(w) = \sum_x p(w,x) \log_2[p(w,x)/p(w)p(x)]$$

where w represents a word and x represents a document. Words are stemmed using [29]. Standard **tf.idf** scheme for term weighting is used and standard cosine similarity between two documents x_1, x_2: $\mathrm{sim}(x_1, x_2) = x_1 \cdot x_2 / |x_1||x_2|$ is used. When each document, column of X, is normalized to 1 using L_2 norm, document-document similarities are calculated as $W = X^T X$. W is interpreted as the weight/affinity matrix of the undirected graph. From this similarity matrix, we perform the clustering as explained above.

For comparison purpose, we also consider three other clustering methods: the `RatioCut` the `NormalizedCut` and the principle direction divisive partitioning (`PDDP`) [2]. `PDDP` is based on the idea of principle component analysis (PCA) applied to the vector-space model on X. First X is centered, i.e., the average of each row (a word) is subtracted. Then the first principle direction is computed. The loadings of the documents (the projection of each document on the principle axis) form a 1-dim linear search order. This provides a heuristic very similar to the linear search order provided by the Fiedler vector. Instead of searching through to find a minimum based on some objective function, `PDDP` partitions data into two parts at the center of mass.

We perform these two-cluster experiments in a way similar to cross-validation. We divide one newsgroup A randomly into K_1 subgroups and the other newsgroup B randomly into K_2 subgroups. Then one of the K_1 subgroups of A is mixed with one of the K_2 subgroups of B to produce a dataset G. The graph partition methods are run on this dataset G to produce two clusters. Since the true label of each newsgroup article is known, we use accuracy, percentage of newsgroup articles correctly clustered, as a measure of success. This is repeated for all $K_1 K_2$ pairs between A and B, and the accuracy is averaged. In this way, every newsgroup articles is used the same number of times. The mean and standard deviation of accuracy are listed.

In Table 1, the clustering results are listed for balanced cluster cases, i.e., both subgroups have about 200 newsgroup articles. MinMaxCut performs about the same as Ncut for newsgroups NG1/NG2, where the cluster overlap is small. MinMaxCut performs substantially better than Ncut for newsgroups NG10/NG11 and newsgroups NG18/NG19, where the cluster overlaps are large. MinMaxCut performs slightly better than `PDDP` . Rcut always performs the worst among the 4 methods and will not be studied further.

In Table 2, the clustering results are listed for unbalanced cases, i.e., one subgroup has 300 newsgroup articles and the other subgroup has 200. This is generally a harder problem due to the unbalanced prior distributions. In this case, both MinMaxCut and Ncut perform reasonably well, no clear deterioration is seen, while the performance of `PDDP` clearly deteriorated. This indicates the strength of `MinMaxCut` method using graph model. `MinMaxCut` consistently performs better than `NormalizedCut` for cases where the cluster overlaps are large.

Dataset	`MinMaxCut`	`NormalizedCut`	`RatioCut`	`PDDP`
NG1/NG2	97.2±1.1	97.2±0.8	63.2±16.2	96.4±1.2
NG10/NG11	79.5±11.0	74.4±20.4	54.9±2.5	89.1±4.7
NG18/NG19	83.6±2.5	57.5±0.9	53.6±3.1	71.9±5.4

Table 1: Accuracy (%) of clustering experiments using `MinMaxCut` , `RatioCut` , `NormalizedCut` and `PDDP` . Each test set G is a mixture of 400 news articles, 200 from each newsgroup.

Dataset	MinMaxCut	NormalizedCut	PDDP
NG1/NG2	97.6 ± 0.8 %	97.2± 0.8 %	90.6 ± 2.1%
NG10/NG11	85.7 ± 8.3 %	73.8± 16.6 %	87.4 ± 2.6%
NG18/NG19	78.8 ± 4.5 %	65.7± 0.5 %	59.6 ± 2.4%

Table 2: Accuracy of clustering experiments using `MinMaxCut` , `NormalizedCut` and `PDDP` . Each test set G is a mixture of 300 news articles from one newsgroup and 200 news articles from the other newsgroup.

2.6 Cluster balance: skewed cut analysis

We further study the reasons that `MinMaxCut` consistently outperforms `NormalizedCut` in large overlap cases. `NormalizedCut` sometimes cuts out a small subgraph, because the presence of $s(A, B)$ in the denominators helps to produce a smaller J_{ncut} value for the skewed cut than for the balanced cut.

We examine several cases and one specific case is shown in Figure 1. The cut points and relevant quantities for `MinMaxCut` and `NormalizedCut` are listed in Table 3. `NormalizedCut` has two pronounced valleys, and produces a skewed cut, while `MinMaxCut` has a single valley and gives a balanced cut. Further examination shows that in both cases, the cutsizes $s(A, B)$ obtained in `NormalizedCut` are equal or bigger than the within-cluster similarity of the smaller cluster as listed in Table 3. In these cases, clearly the `NormalizedCut` objective [see Eq.(18)] is not appropriate. In the `MinMaxCut` objective, the cutsize is absent in the denominators; this provides a balanced cut.

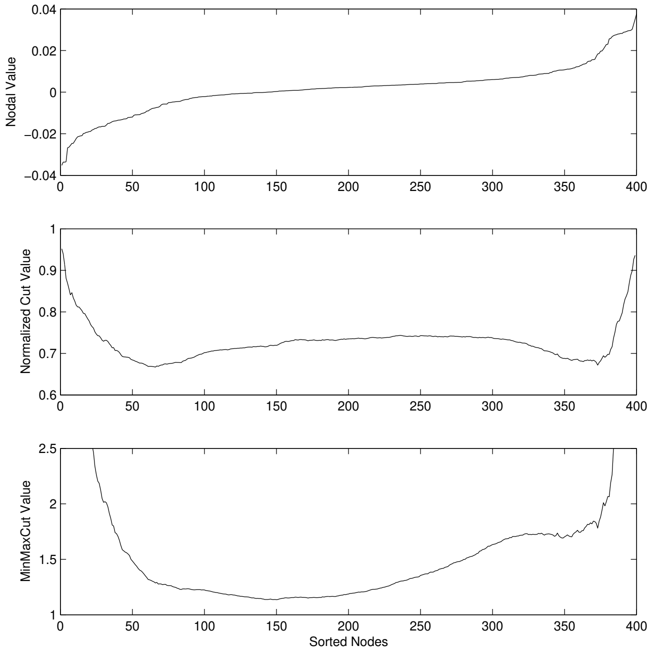

Figure 1: Top: Nodal values of $\mathbf{q}_2$. Middle: `NormalizedCut` values as the cutpoint moves from $i_{cut} = 1, \cdots, n-1$. Bottom: `MinMaxCut` values. Dataset from NG18/NG19. The skewed cut of `NormalizedCut` is obvious.

These case studies provide some insights into those graph partition methods. Prompted by these studies, here we provide further analysis and derive general conditions under which a skewed cut will occur. Consider the balanced cases where $s(A, A) \simeq s(B, B)$. Let

$$s(A, B) = f \cdot \langle s \rangle, \quad \langle s \rangle = \frac{1}{2}(s(A, A) + s(B, B)),$$

Method	i_{cut}	s(A,B)	s(A,A)	s(B,B)
$J_{\rm ncut}$	66	869.6	771.2	5467
$J_{\rm MMC}$	150	1418	2136	3006

Table 3: Cutpoint, between-cluster and within-cluster similarities for the dataset in Figure 1.

where $f > 0$ is the average fraction of cut relative to within cluster associations.

In the case when the partition is optimal, A and B are exactly the partitioning result. The corresponding **NormalizedCut** value is

$$J_{\rm ncut}(A,B) = \frac{s(A,B)}{s(A,A)+s(A,B)} + \frac{s(A,B)}{s(B,B)+s(A,B)} \simeq \frac{2f}{1+f} \tag{27}$$

For a skewed partition A_1, B_1, we have $s(A_1,A_1) \ll s(B_1,B_1)$, and therefore $s(A_1,B_1) \ll s(B_1,B_1)$. The corresponding $J_{\rm ncut}$ value is

$$J_{\rm ncut}(A_1,B_1) \simeq \frac{s(A_1,B_1)}{s(A_1,A_1)+s(A_1,B_1)} \tag{28}$$

Using **NormalizedCut** , a skewed or incorrect cut will happen if $J_{\rm ncut}(A_1,B_1) < J_{\rm ncut}(A,B)$ Using Eqs.(27, 28), this condition is satisfied if

$$\texttt{NormalizedCut}: \quad s(A_1,A_1) \geq (\frac{1}{2f} - \frac{1}{2}) s(A_1,B_1)$$

We repeat the same analysis using **MinMaxCut** and calculating $J_{\rm MMC}(A,B)$ and $J_{\rm MMC}(A_1,B_1)$. The condition for a skewed cut using MinMaxCut is $\text{MinMaxCut}_1 < \text{MinMaxCut}_0$, which is

$$\texttt{MinMaxCut}: \quad s(A_1,A_1) \geq \frac{1}{2f}\, s(A_1,B_1).$$

For large overlap case, say, $f = 1/2$, the conditions for possible skewed cut are:

$$\begin{aligned} \texttt{NormalizedCut}: \quad & s(A_1,A_1) = s(B_1,B_1) & \leq & \quad s(A_1,B_1)/2, \\ \texttt{MinMaxCut}: \quad & s(A_1,A_1) = s(B_1,B_1) & \leq & \quad s(A_1,B_1). \end{aligned} \tag{29}$$

The relevant quantity is listed in Table 4. For datasets newsgroups 10-11, and newsgroups 18-19, the condition for skewed **NormalizedCut** is satisfied most of the time, leading to many skewed cuts and therefore lower clustering accuracy in Tables 1 and 2. For the same datasets, condition for skewed MinMaxCut is not satisfied most of time, leading to more correct cuts and therefore higher clustering accuracy. Eq.(29) is the main results of this analysis.

Dataset	s(A,B)	s(A,A)	s(B,B)	f
NG1/NG2	549.4	1766.4	1412.5	0.346
NG10/NG11	772.8	1372.8	1581.0	0.523
NG18/NG19	1049.5	2093.9	1665.5	0.558

Table 4: Average values of s(A,B), s(A,A), s(B,B) and the fraction in three datasets using **MinMaxCut** .

2.7 Improved `MinMaxCut`: Linkage-based refinements

So far we has discussed **MinMaxCut** using the eigenvectors of Eq.(11) as the continuous solution of the objective function as provided by Theorem 2.1. This is a good solution to the **MinMaxCut** problem, as the experiment results shown above. But this is still an approximate solution. Given a current clustering solution, we can refine it to improve the `MinMaxCut` objective function. There are many ways to refine a given clustering solution. In this and next subsections, we discuss two refinement straitagies and show the corresponding experimental results.

Searching for optimal i_{cut} in Theorem 2.1 is equivalent to a linear search based on the order defined by sorting the elements of $\mathbf{q}_2$, which we call $\mathbf{q}_2$-order. Let $\pi = (\pi_1, \cdots, \pi_n)$ represent a permutation of $(1, \cdots, n)$. The $\mathbf{q}_2$-order is the permutation π induced when sorting $\mathbf{q}_2(i)$ in increasing order, i.e., $\mathbf{q}_2(\pi_i) \leq \mathbf{q}_2(\pi_{i+1})$ for all i. The linear search algorithm based on π is to search for minimal $J_{\text{MMC}}(A, B)$ as $j = 1, 2, \cdots, n-1$, while setting clusters C_1, C_2 as

$$A = \{i \mid \mathbf{q}_2(\pi_i) \leq \mathbf{q}_2(\pi_j)\}, \quad B = \{i \mid \mathbf{q}_2(\pi_i) > \mathbf{q}_2(\pi_j)\}. \tag{30}$$

The linear search implies that nodes on one side of the cut point must belong to one cluster: if $\mathbf{q}_2(i) \geq \mathbf{q}_2(j) \geq \mathbf{q}_2(k)$ where i, j, k are nodes, then the linear search will not allow the situation that i, k belong to one cluster and j belongs to the other cluster. Such a strict order is not necessary. In fact, in large overlap cases, we expect some nodes could be moved to the other side of the cut, lowering the overall objective function.

How to identify those nodes near the boundary of between the two clusters? For this purpose, we define linkage ℓ as a closeness or similarity measure between two clusters (subgraphs):

$$\ell(A, B) = s(A, B)/s(A, A)s(B, B) \tag{31}$$

(This is motivated by the *average linkage* $\ell(A, B) = s(A, B)/|A||B|$ in hierarchical agglomerative clustering. Following the spirit of `MinMaxCut` , we replaced $|A|, |B|$ by $s(A, A)$, $s(B, B)$). For a single node u, its linkage to subgraph A is $\ell(A, u) = s(A, u)/s(A, A)$. Now we can identify the nodes near the cut. If a node u is well inside a cluster, u will have a large linkage with the cluster, and a small linkage with the other cluster. If u is near the partition boundary, its linkages with both clusters should be close. Therefore, we define the linkage difference

$$\Delta\ell(u) = \ell(u, A) - \ell(u, B). \tag{32}$$

A node with small $\Delta\ell$ should be near the cut and is a possible candidate to be moved to the other cluster.

In Figure 2, we show linkage difference $\Delta\ell$ for all nodes. The vertical line is the cut point. It is interesting to observe that not only many nodes have small $\Delta\ell$, but quite a number of nodes whose $\Delta\ell$ have the wrong signs (e.g., $\Delta\ell(u) < 0$ if $u \in A$, or, $\Delta\ell(v) > 0$ if $v \in B$). For example, node #62 has a relatively large negative $\Delta\ell$. This implies node #62 has a larger linkage to cluster B even though it is currently located in cluster A (left of the cutpoint). Indeed, if we move node #62 to cluster B, the objective function is reduced. Therefore we find a better solution.

After moving node #62 to cluster B, we try to move another node with negative $\Delta\ell$ from cluster A to cluster B depending on whether the objective function is lowered. In fact, we move all nodes in cluster A with negative $\Delta\ell$ to cluster B if the objective function is lowered. Similarly we move all nodes in cluster B with positive $\Delta\ell$ to cluster A. This procedure of swapping nodes is called the "linkage-based swap". It is implemented by sorting the array $s(u)\Delta\ell(u)$ [$s(u) = -1$ if $u \in A$ and $s(u) = 1$ if $u \in B$] in decreasing order to provide a priority list and then moving the nodes, one by one. The greedy move starts from the top of the list to the last node u where $s(u)\Delta\ell(u) \geq 0$. This swap reduces the objective function and increases the partitioning quality. In Table 5, the effects on clustering accuracy due to the swap are listed. In all cases, the accuracy increases. Note that in the large overlap cases, NG9/NG10, NG18/NG19, the accuracy increase about 10% over the MinMaxCut without refinement.

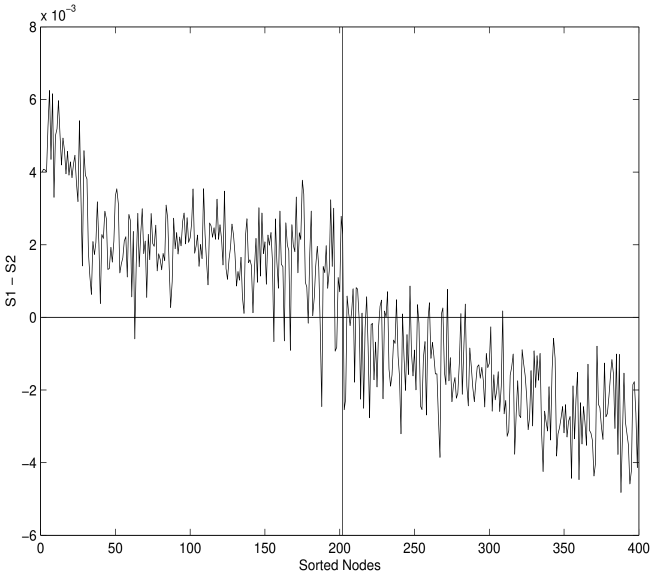

Figure 2: Linkage difference of all nodes. The vertical line indicates the cutpoint using MinMaxCut. Nodes on the left forms cluster A and nodes on the right forms cluster B.

If $s(u)\Delta\ell(u) < 0$ but close to 0, node u is in the correct cluster, although it is close to the cut. Thus we select the smallest 5% of the nodes with $s(u)\Delta\ell(u) < 0$ as the candidates, and move those which reduce MinMaxCut objective to the other cluster. This is done in both cluster A and B. We call this procedure "linkage-based move". Again, these moves reduce MinMaxCut objective and therefore improve the solution. In Table 5, their effects on improving clustering accuracy are shown. Putting together, the linkage bassed refinements improve the accuracy by 20%. Note the final MinMaxCut results are about 30-50% better than `NormalizedCut` and about 6-25% better than `PDDP` (see Tables 5 and 1).

Dataset	`MinMaxCut`	+Swap	+Swap+Move
NG1/NG2	97.2 ± 1.1%	97.5 ± 0.8 %	97.8 ±0.7%
NG10/NG11	79.5 ±11.0%	85.0 ± 8.9 %	94.1 ±2.2%
NG18/NG19	83.6 ± 2.5%	87.8 ± 2.0 %	90.0 ±1.4%

Table 5: Improvements of clustering accuracy due to linkage-based refinements for `MinMaxCut` alone, `MinMaxCut` plus swap, and `MinMaxCut` plus swap and move over 5% smallest $\Delta\ell$ on both sides of the cutpoint.

2.8 Improved `MinMaxCut`: Linkage differential order

Given a current clustering solution A, B, we can always compute the linkage difference Eq.(32) for every nodes. Now by sorting linkage differences we obtain an ordering which we call linkage differential ordering (LD-order).

The motivation of the LD-order is from observing linkage differences as shown in Figure 2. We see that many nodes far away from the cut point have wrong $\Delta\ell$ signs, that is, they should belong to the other subgraph. This suggests that the $\mathbf{q}_2$-order is not the perfect linear search order.

This prompt us to apply the linear search algorithm of Eq.(30) to the LD-order to search for optimal `MinMaxCut` . The results are given in Table 6. We see that the MinMaxCut values obtained on LD-order are lower than that based on the $\mathbf{q}_2$-order. The clustering accuracy also increases substantially. Note that the LD order can be recursively applied to the clustering results for further improvements.

Dataset	Acc($\mathbf{q}_2$)	Acc(LD)	$J^{\rm opt}_{\rm MMC}(\mathbf{q}_2)$	$J^{\rm opt}_{\rm MMC}$(LD)
NG1/NG2	97.2 ± 1.1%	97.6±0.8%	0.698	0.694
NG10/NG11	79.5 ±11.0%	87.2±8.0%	1.186	1.087
NG18/NG19	83.6 ± 2.5%	89.2±1.8%	1.126	1.057

Table 6: Improvements on accuracy (2nd and 3rd columns) due to the linkage differential order over $\mathbf{q}_2$-order. Improvements on $J^{\rm opt}_{\rm MMC}$ values are also shown. (4th and 5th columns).

2.9 Bi-clustering: simultaneous clustering of rows and columns of a contingency table

In many applications we look for inter-dependence among different aspects (attributes) of the same data objects. For example in text processing, a collection of documents is represented by a rectangular word-document association matrix, where each column represents a document and each row represents a word. The mutual interdependence reflect the fact that the content of a document is determined by the word occurrences, while the meaning of words can be inferred through their occurrences across different documents. The association data matrix $P = (p_{ij})$ typically has non-negative data entries. It can be studied as contingency table and viewed as a bipartite graph with P as its adjacency matrix as shown in Figure 3. A row is represented by an r-node and a column by a c-node. Co-occurrence counts (probability) between row r_i and column c_j is represented by a weighted edge between r_i and c_j.

For a contingency table with m rows and n columns, we wish to partition the rows R into two clusters R_1, R_2 and simultaneously partition the columns C into two clusters C_1, C_2. Let $s(R_p, C_q) \equiv \sum_{r_i \in R_p} \sum_{c_j \in C_q} p_{ij}$. Clusterings are done such that between-cluster associations $s(R_1, C_2), s(R_2, C_1)$ are minimized while within-cluster associations $s(R_1, C_1), s(R_2, C_2)$ are maximized (see Figure 3). These min-max clustering requirements lead to the following objective

$$J_{\rm MMC}(C_1, C_2; R_1, R_2) = \frac{s(R_1, C_2) + s(R_2, C_1)}{2s(R_1, C_1)} + \frac{s(R_1, C_2) + s(R_2, C_1)}{2s(R_2, C_2)} \tag{33}$$

If $n = m$ and $p_{ij} = p_{ji}$, Eq.(2.9) is reduced to Eq.(2). Let indicator vector $\mathbf{f}$ determine how to split R into R_1, R_2 and indicator vector $\mathbf{g}$ splits C into C_1, C_2:

$$f_i = \begin{cases} a & \text{if} \quad r_i \in R_1 \\ -b & \text{if} \quad r_i \in R_2 \end{cases}, \quad g_i = \begin{cases} a & \text{if} \quad c_i \in C_1 \\ -b & \text{if} \quad c_i \in C_2 \end{cases} \tag{34}$$

Let $d^r_i = \sum_{j=1}^n p_{ij}$ be row sums and $d^c_j = \sum_{i=1}^m p_{ij}$ be column sums. Form diagonal matrices $D_r = \text{diag}(d^r_1, \cdots, d^r_m)$, $D_c = \text{diag}(d^c_1, \cdots, d^c_n)$. Define the scaled association matrix,

$$\widehat{P} = D_r^{-1/2} P D_c^{-1/2} = \sum_{k=1}^{\min(n,m)} \mathbf{u}_k \lambda_k \mathbf{v}_k^T, \tag{35}$$

with the singular value expansion explicitly written. We have the following:

Theorem 2.9. Simultaneous clustering of rows and columns based on the objective function Eq.(2.9), the continuous solution of the optimal clustering indicators are given by $\mathbf{f}_2 = D_r^{-1/2}\mathbf{u}_2$, and $\mathbf{g}_2 = D_c^{-1/2}\mathbf{v}_2$.

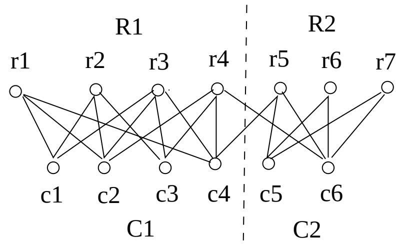

Figure 3: A bipartite graph with r-nodes and c-nodes. The dashed line indicates a possible clustering. Between-cluster associations $s(R_1, C_2)$, $s(R_2, C_1)$ go across the dashed line. Within-cluster associations $s(R_1, C_1)$, $s(R_2, C_2)$ stay on the same side.

The proof is an extension of Theorem 2.1 by treating the bipartite graph P as a standard graph[36] $S = \begin{pmatrix} 0 & P \\ P^T & 0 \end{pmatrix}$. Details are skipped due to space limit. The use of SVD is also noted in [7].

3 K-way MinMaxCut

So far we have focused on 2-way clustering. Now we extend to K-way cluster, $K \geq 3$. We define the objective function as the sum of all possible pairs of 2-way J_{MMC}:

$$J_{\text{MMC}}(C_1, \cdots, C_K) = \sum_{1 \leq p < q \leq K} J_{\text{MMC}}(C_p, C_q) = \sum_{k=1}^{K} \frac{s(C_k, \bar{C}_k)}{s(C_k, C_k)} \tag{36}$$

where $\bar{C}_k = \sum_{p \neq k} C_p$ is the complement of C_k. For comparison, `RatioCut` is extended to K-way clustering as [4]

$$J_{\text{rcut}}(C_1, \cdots, C_K) = \sum_{k=1}^{K} \frac{s(C_k, \bar{C}_k)}{|C_k|}, \tag{37}$$

and `NormalizedCut` is extended to K-way clustering as

$$J_{\text{ncut}}(C_1, \cdots, C_K) = \sum_{k=1}^{K} \frac{s(C_k, \bar{C}_k)}{d_{C_k}} = \sum_{k=1}^{K} \frac{s(C_k, \bar{C}_k)}{s(C_k, C_k) + s(C_k, \bar{C}_k)}. \tag{38}$$

Note that for large K, $s(C_k, \bar{C}_k) = \sum_{p \neq k} s(C_p, C_k)$ is likely to be larger than the within-cluster similarity $s(C_k, C_k)$, i.e., `MinMaxCut` differs from `NormalizedCut` much more than in the $K = 2$ case. From the analysis in section §2.6, `NormalizedCut` is more likely to produce skewed cuts. Hence `MinMaxCut` is essential in $K-$way clustering.

The analysis of `MinMaxCut` , `RatioCut` , and `NormalizedCut` on random graph model as in section §2.2 can be easily extended to $K \geq 3$ case, with identical conclusions. i.e., `RatioCut` and `NormalizedCut` show no size preferences, while on random graph model as in `MinMaxCut` favors balanced cut.

3.1 Cluster balance: size vs. similarity

In the above on cluster balance, we are primarily concerned with cluster size, i.e., we desire the final clusters obtained have approximately same sizes,

$$|C_1| \simeq |C_2| \simeq \cdots \simeq |C_K|. \tag{39}$$

There is another form of cluster balance, as we discuss below. First of all, when minimizing $J_{\text{MMC}}(C_1, \cdots, C_K)$, there are K terms, all of which are positive. For J_{MMC} to be minimized, all terms should be of approximately same value: minimization does not favor the situation that one term is much larger than the rest. Thus we have

$$\frac{s(C_1, \bar{C}_1)}{s(C_1, C_1)} \simeq \frac{s(C_2, \bar{C}_2)}{s(C_2, C_2)} \simeq \cdots \simeq \frac{s(C_K, \bar{C}_K)}{s(C_K, C_K)} \tag{40}$$

Now define the average between-cluster similarity $\bar{s}_{k\bar{k}}$ and the average within-cluster similarity $\bar{s}_{kk}$,

$$\bar{s}_{k\bar{k}} = \frac{s(C_k, \bar{C}_k)}{|C_k|(n - |C_k|)}, \quad s_{kk} = \frac{s(C_k, C_k)}{|C_k|^2}.$$

we have

$$\frac{\bar{s}_{1\bar{1}}|(n - |C_1|)}{\bar{s}_{11}|C_1|} \simeq \frac{\bar{s}_{2\bar{2}}|(n - |C_2|)}{\bar{s}_{22}|C_2|} \simeq \cdots \simeq \frac{\bar{s}_{K\bar{K}}|(n - |C_K|)}{\bar{s}_{KK}|C_K|}$$

Assume further that $\bar{s}_{1\bar{1}} \approx \cdots \approx \bar{s}_{k\bar{k}}$ and also $|C_k| \ll n$, we obtain

$$\bar{s}_{11}|C_1| \simeq \bar{s}_{22}|C_2| \simeq \cdots \simeq \bar{s}_{KK}|C_K|. \tag{41}$$

We call this the similarity-weighted size balance. `MinMaxCut` is studied in a recent study on clustering objective functions [37], where for a dataset of articles about *sports*, for $K = 10$ clustering, `MinMaxCut` produces clusters where the cluster sizes vary about a factor of 3.3 while the the similarity-weighted cluster size vary only a factor of 1.5 (example in Table 9 of [37]).

3.2 Bounds of K-way `MinMaxCut`

The lower and upper bounds of J_{MMC} for $K = 2$ (see section §2.4) can be extended to $K > 2$ case:

Theorem 3.2. For K-way `MinMaxCut` , we have the following bounds:

$$\frac{K^2}{1 + \zeta_2 + \cdots + \zeta_K} - K \leq J^{\text{opt}}_{\text{MMC}}(C_1, \cdots, C_K) \leq \frac{K^2 - K}{1 - K/n} \tag{42}$$

where $\zeta_2, \cdots, \zeta_K$ are the largest eigenvalues of Eq.(12).

Proof. The proof of the lower-bound relating to the first K eigenvectors is given in [18] (which differ from those for $K = 2$ in §2.1 and §2.4). The upper-bound is a simple extension from the $K = 2$ case. □

3.3 Initial K-way `MinMaxCut` clustering

K-way `MinMaxCut` is more complicated because there are multiple eigenvectors involved as explained by Theorem 3.2. Our approach is to first obtain approximate K initial clusters and then refine them. We discuss three methods for initial clusterings here.

Eigenspace K-means As provided by Theorem 3.2. cluster membership indicators of the K-way `MinMaxCut` are closely related to the first K eigenvectors. Thus we may use the projection in the K-dimensional eigenspace formed by the K eigenvectors and perform a K-means clustering. K-means cluster is a popular and efficient method. It minimizes the following clustering objective function

$$J_{\text{Kmeans}}(K) = \sum_{k=1}^{K} \sum_{i \in C_k} (\mathbf{x}_i - \mathbf{c}_k)^2$$

where $\mathbf{x}_i$ is projected feature vector in the eigenspace and $\mathbf{c}_k = \sum_{i \in C_k} \mathbf{x}_i / |C_k|$. This approach has been used in [4, 34, 35].

Divisive `MinMaxCut`. We start from the top, treating the whole dataset as a cluster. We repeatedly partition a current cluster into two via the 2-way `MinMaxCut` (a leaf node in a binary tree) until the number of clusters reaches a predefined value K, or some other stopping criteria are met. The crucial issue here is how to select the next candidate cluster to split. Details is explained in section §4.

Agglomerative `MinMaxCut`. Here clusters are built from bottom up like conventional hierarchical agglomerative clustering. During each recursive procedure, we select two *current* clusters C_p and C_q and merge them to form a bigger cluster. The standard cluster selection methods include single linkage, complete linkage and average linkage. For `MinMaxCut` objective function, the MinMax linkage of Eq.(31) seems to be more appropriate. The cluster merging is repeated until a stopping condition is met.

3.4 K-way `MinMaxCut` refinement

Once the initial clustering (i.e., in divisive `MinMaxCut`) is computed, the refinements should be applied to improve the `MinMaxCut` objective function. The cluster refinement for $K = 2$ discussed in §2.7 may be extended to $K > 2$ case by applying the 2-way linkage-based refinement *pairwisely* on all pairs of clusters.

On the other hand a direct K-way linkage-based refinement procedure may be adopted: Assume a node u currently belongs to cluster C_k. The linkage difference $\Delta\ell_{pq}(u) = \ell(u, C_p) - \ell(u, C_q)$ for all other $K - 1$ clusters are computed. The smallest $\Delta\ell_{pq}(u)$ and the corresponding cluster indices are stored as an entry in a priority list. This is repeated for all nodes so every entry of the list is filled. The list is then sorted according to $\Delta\ell_{pq}(u)$ to obtain the final priority list. Following the list, nodes are then moved one after another to the appropriate clusters if the overall `MinMaxCut` objective is reduced. This completes one pass. Several passes may be necessary.

4 Divisive `MinMaxCut`

Divisive `MinMaxCut` is one practical algorithm for implementing K-way `MinMaxCut` via the hierarchical approach. It amounts to recursively select and split a cluster into two smaller ones in a top-down fashion until terminated. One advance of our divisive `MinMaxCut` over the traditional hierarchical clustering is that our methods have a clear objective function; Refinements of the clusters obtained from divisive process improve both the objective function and the clustering accuracy, as demonstrated in the experiments (§4.5). Divisive clustering depends crucially on the criterion of selecting the cluster to split.

4.1 Monotonicity of cluster objective functions

It is instructive to see how clustering objective functions change with respect to the change of K, the number of clusters. Given the dataset and similarity measure (Euclidean distance in K-means and similarity graph weight in `MinMaxCut`), the global optimal value of the objective function is a function of K. An important property of these clustering objective functions is the monotonicity: as K increases $K = 2, 3, \cdots$, the `MinMaxCut` objective *increases* monotonically, while the K-means objective *decreases* monotonically. Thus there is a fundamental difference between the graph-based `MinMaxCut` and the Euclidean distance based K-means :

Theorem 4.1. Given the dataset and the similarity metric, as K increases, (a) the optimal value of the K-means objective function decreases monotonically:

$$J^{\text{opt}}_{\text{Kmeans}}(C_1, \cdots, C_K) > J^{\text{opt}}_{\text{Kmeans}}(C_1, \cdots, C_K, C_{K+1})$$

and (b) the optimal value of the MinMax Cut objective function increases monotonically:

$$J^{\text{opt}}_{\text{MMC}}(C_1, \cdots, C_K) < J^{\text{opt}}_{\text{MMC}}(C_1, \cdots, C_K, C_{K+1})$$

Proof. (a) is previously known. To prove (b), we assume A, B_1, B_2 are the optimal clusters for $K = 3$ for a given dataset. and merge B_1, B_2, into a cluster. We compute the current $J_{\text{MMC}}(A, B)$ and obtain

$$J^{\text{B-merge}}_{\text{MMC}}(A, B) - J^{\text{opt}}_{\text{MMC}}(A, B_1, B_2) = \frac{s(A, B)}{s(B, B)} - \frac{s(B_1, A) + s(B_1, B_2)}{s(B_1, B_1)} - \frac{s(B_2, A) + s(B_2, B_1)}{s(B_2, B_2)} < 0,$$

noting $s(A, B) = s(A, B_1) + s(A, B_2)$, $s(B_1, B_1) < s(B, B)$ and $s(B_2, B_2) < s(B, B)$. The global minimum for $K = 2$ must be lower than or equal to the particular instance of $J_{\text{MMC}}(A, B)$. Thus we have

$$J^{\text{opt}}_{\text{MMC}}(A, B) \leq J^{\text{B-merge}}_{\text{MMC}}(A, B) < J^{\text{opt}}_{\text{MMC}}(A, B_1, B_2).$$

□

Theorem 4.1 shows the difference between `MinMaxCut` objective and K-means objective. If we use the optimal value of the objective function to judge what is the optimal K, then K-means favors large number of clusters while MinMaxCut favors small number of clusters. The monotonic increase or decrease indicate that one cannot determine optimal K from the objective functions alone. Another consequence is that in the top-down divisive clustering, as clusters are split into more clusters, the K-means objective will steadily decrease while the MinMaxCut objective will steadily increase.

4.2 Cluster selection

Suppose the dataset is clustered into m clusters in the divisive clustering. The question is how to select one of these m clusters to split.

(1) Size-priority cluster split. Select the cluster with largest size to split. This approach gives priority to produce size-balanced clusters. However, natural clusters are not restricted to the situation where each cluster has the same size. Thus this approach is not necessarily the optimal approach.

(2) Average similarity. Define *average* within-cluster similarity as $\bar{s}_{kk} = s_{kk}/n_k^2$. We select the cluster with smallest $\bar{s}_{kk}$ to split. A cluster C_k with large $\bar{s}_{kk}$ implies that cluster members are strongly similar to each other, i.e., the cluster is compact. This criterion will increase the compactness of resulting clusters, which is a goal of min-max clustering principle.

(3) Cluster cohesion. We select the cluster p with the smallest cohesion among the current leaf clusters: $p = \arg\min_k h_k$. A cluster C_k with small cohesion h_k implies it can be meaningfully split into two.

(4) Similarity-cohesion. Combination of cohesion with average similarity. We select the cluster p according to

$$p = arg \min_k \ \bar{s}^{\gamma}_{kk} h^{(1-\gamma)}_k. \tag{43}$$

by setting $\gamma = 1/2$. Note that setting $\gamma = 1$, we get similarity criterion; setting $\gamma = 0$, we get cohesion criterion.

(5) Temporary objective. All above cluster choices are based on cluster characteristics and do not involve the clustering objective Eq.(36). Since the goal of clustering is to optimize the objective function, we choose the cluster C_k such that the split of C_k leads to the smallest increase in the overall objective temporarily.

4.3 Stopping criteria

In our experiments below, we terminate the divisive procedure when the number of leaf clusters reaches the pre-defined K. Another criterion is based on cluster cohesion. Theorem 4.1(b) indicates that as the divisive process continues and the number of leaf clusters increase, cluster cohesion of these leaf clusters increases. So a threshold on cohesion is a good stop criterion in applications.

4.4 Objective function saturation

If a dataset has K reasonably distinguishable clusters, these natural clusters could have many different shapes and sizes. But in many datasets, clusters overlap substantially and natural clusters cannot be defined clearly. Therefore, in general, a single (even the "best" if exists) objective function J can not effectively model the vast different types of datasets. For many datasets, as J is optimized, the accuracy (quality) of clustering is usually improved. But this works only up to a point. Beyond that, further optimization of the objective will not improve the quality of clustering because the objective function does not necessarily model the data in fine details. We here formalize this characteristics of clustering objective function as the *saturation* of objective function.

Definition. For a given measure η of quality of clustering (i.g, accuracy), the saturation objective, $J_{\rm sat}$, is defined to be the value when J is further optimized beyond $J_{\rm sat}$, η is no longer improved. We say η reaches its saturation value $\eta_{\rm sat}$.

Saturation accuracy is a useful concept and also a useful measure. Given a dataset with known class labels, there is a unique saturation accuracy for a clustering method. Saturation accuracy gives a good sense on how well the clustering algorithm will do on the given dataset.

In general we have to use the clustering method to do extensive clustering experiments to compute saturation accuracy. Here we propose an effective method to compute an upper bound on saturation accuracy for a clustering method. The method is the following. (a) Initialize with the perfect clusters constructed from the known class labels. At this stage, the accuracy is 100%. (b) Run the refinement algorithm on this clustering until convergence. (c) Compute accuracy and other measures. These values are the upper bounds on saturation values.

4.5 K-way clustering of Internet newsgroups

We apply the divisive `MinMaxCut` algorithm to document clustering. We perform experiments on Internet newsgroup articles in 20 newsgroups. as in §2.5. We focus on two sets of 5-cluster cases. The choice of $K = 5$ is to have enough levels in the cluster tree; we avoid $K = 4, 8$ where the clustering results are less sensitive to cluster selection. The first dataset includes

```
  Dataset M5:                         Dataset L5:

NG2:  comp.graphics                 NG2:  comp.graphics
NG9:  rec.motorcycles               NG3:  comp.os.ms-windows
NG10: rec.sport.baseball            NG8:  rec.autos
NG15: sci.space                     NG13: sci.electronics
NG18: talk.politics.mideast         NG19: talk.politics.misc
```

In M5, clusters overlap at medium level. In L5, overlaps among different clusters are large. From each set of newsgroups, we construct two datasets of different sizes: (A) randomly select 100 articles from each newsgroup. (B) randomly select 200, 140, 120, 100, 60 from the 5 newsgroups, respectively. Dataset (A) has clusters of equal sizes, which is presumably easier to cluster. Dataset (B) has clusters of significantly varying sizes, which is presumably difficult to cluster. Therefore, we have 4 newsgroup - cluster size combination categories

L5B: large overlapping clusters of balanced sizes

L5U: large overlapping clusters of unbalanced sizes

M5B: medium overlapping clusters of balanced sizes

M5U: medium overlapping clusters of unbalanced sizes

For each category, 5 different datasets randomly sampled from the newsgroups dataset; the divisive MinMaxCut algorithm is applied to each of them. The final results are the average of these 5 random datasets in each categories.

	M5B	M5U	L5B	L5U
Saturation	92.5(2.0)	91.7(1.6)	81.4(2.1)	79.0(4.4)
Size-P I	82.8(3.4)	77.1(10.8)	67.2(2.9)	62.9(6.7)
Size-P F	91.8(1.7)	81.7(9.9)	71.8(4.8)	68.4(1.9)
cohesion I	66.1(10.6)	75.6(13.8)	46.3(11.6)	50.9(14.7)
cohesion F	73.0(10.8)	78.8(13.2)	49.6(5.3)	58.1(13.8)
Tmp-obj I	80.3(9.0)	70.9(2.2)	56.9(4.9)	60.1(4.2)
Tmp-obj F	87.0(11.6)	75.0(1.3)	58.7(5.6)	68.8(2.8)
avg-sim I	83.5(2.0)	88.4(1.8)	69.3(2.3)	74.8(4.6)
avg-sim F	91.7(1.1)	91.7(1.3)	72.4(4.1)	74.1(2.5)
sim-coh I	83.5(2.0)	88.4(1.8)	63.5(5.4)	71.0(2.3)
sim-coh F	91.8(1.2)	91.0(1.0)	67.1(8.0)	72.6(2.3)

Table 7: Accuracy (in percentage) of divisive `MinMaxCut` clustering. Errors in parenthesis.

The results of clustering on the four datasets are listed in Table 7. The upper bounds of saturation values are computed as described in §4.4. Clustering results for each cluster selection method, size-priority (Size-P), average similarity (avg-sim), cohesion and similarity-cohesion (sim-coh) (see Eq.43) and temporary objective (tmp-obj) are given in 2 rows: "I" (initial) are the results immediately after divisive cluster; "F" (final) are the results after two rounds of greedy refinements.

A number of observations can be made from these extensive clustering experiments. (1) The best results are obtained by average similarity cluster selection. This is consistent for all 4 datasets. (2) The similarity-cohesion cluster selection gives very good results, statistically no different from average similarity selection method. (3) Cluster cohesion alone as the selection method gives consistently poorest results. The temporary objective choice performs slightly better than cohesion criterion, but still substantially below avg-sim and sim-coh choices. These results are somehow unexpected. We checked the details of several divisive processes. The temporary objective and cohesion often lead to unbalanced clusters because of the greedy nature and unboundedness of these choices[1]. (4) Size-priority selection method gives good results for datasets with balanced sizes, but not as good results for datasets with unbalanced cluster sizes. These are as expected. (5) The refinement based on `MinMaxCut` objective almost *always* improves the accuracy for all cluster selection methods on all datasets. This indicates the importance of refinements in hierarchical clustering. (6) Accuracies of the final clustering with avg-sim and sim-coh choices are very close to the saturation values, indicating the obtained clusters are as good as the `MinMaxCut` objective function could provide. (7) Dataset M5B has been studied in [35] using K-means methods. The standard K-means method achievers an accuracy of 66%, while two improved K-means methods achieve 76-80% accuracy.

[1]A current cluster C_k is usually split into balanced clusters C_{k1}, C_{k2} by the `MinMaxCut` . However, C_{k1} and C_{k2} may be quite smaller than other current clusters, because no mechanism exists in the divisive process to enforce balance across all current clusters. After several divisive steps, they could become substantially out of balance. In contrast, avg-similarity and size-priority choices prevent large unbalance to occur.

In comparison, the divisive `MinMaxCut` achieves 92% accuracy.

5 Summary and discussions

In this paper, we provide a comprehensive analysis on `MinMaxCut` spectral data clustering method. Comparing to earlier clustering methods, `MinMaxCut` has a strong cluster balancing feature (§2.2, §2.6, §3.1). The 2-way clustering can be computed easily while the K-way clustering requires a divisive clustering (§4).

In divisive `MinMaxCut` , cluster selections based on average similarity and cluster cohesion leads to balanced clusters in final stage and thus better clustering quality. Experiments on agglomerative `MinMaxCut` (as discussed in §3.3) indicate[8] that agglomerative `MinMaxCut` is as good as the divisive `MinMaxCut` , both in clustering quality and in computational efficiency.

Our extensive experiments, on medium and large overlapping clusters with balanced and unbalanced cluster sizes, show that refinements of the clusters obtained in divisive and agglomerative `MinMaxCut` *always* improve clustering quality, strongly indicating the min-max clustering objective function captures the essential features of clusters in a wide range of situations. This supports our emphasis on the objective function optimization based approach.

Since the cluster refinement is an essential part of objective function based approach, efficient refinement algorithms are needed. The refinement methods discussed in §2.7, §2.8, §3.4 are of order $O(n^2)$ complexity. An efficient refinement algorithm like Fiduccia-Mattheyses linear time heuristic[15] is highly desirable.

A counter point to the objective function optimization approach is the objective function saturation, i.e., objective optimization is useful only up to a certain point (see §4.4). Therefore finding a universal clustering objective function is another important direction of research. On the order hand, the saturation values of accuracy or objective functions can be used as a good assessment of the effectiveness of the clustering method as shown in Table 7. However, this point does not favor the procedure oriented clustering approach, where the lack of objective function makes the self-consistent assessment impossible; justifications of the method are empirical.

Acknowledgments. This work is supported by U.S. Department of Energy, Office of Science (MICS Office and LDRD) under contract DE-AC03-76SF00098 and an NSF grant CCR-0305879.

References

[1] J.D. Banfield and A. E. Raftery. Model-based gaussian and non-gaussian clustering. *Biometrics*, 49:803–821, 1993.

[2] D. Boley. Principal direction divisive partitioning. *Data mining and knowledge discovery*, 2:325–344, 1998.

[3] B. Bollobas. *Random Graphs*. Academic Press, 1985.

[4] P.K. Chan, M.Schlag, and J.Y. Zien. Spectral k-way ratio-cut partitioning and clustering. *IEEE Trans. CAD-Integrated Circuits and Systems*, 13:1088–1096, 1994.

[5] C.-K. Cheng and Y.A. Wei. An improved two-way partitioning algorithm with stable performance. *IEEE. Trans. on Computed Aided Desgin*, 10:1502–1511, 1991.

[6] F.R.K. Chung. *Spectral Graph Theory*. Amer. Math. Society, 1997.

[7] I. S. Dhillon. Co-clustering documents and words using bipartite spectral graph partitioning. *Proc. ACM Int'l Conf Knowledge Disc. Data Mining (KDD 2001)*, 2001.

[8] C. Ding and X. He. Cluster merge and split in hierarchical clustering. *Proc. IEEE Int'l Conf. Data Mining*, pages 139–146, 2002.

[9] C. Ding, X. He, and H. Zha. A spectral method to separate disconnected and nearly-disconnected web graph components. In *Proc. ACM Int'l Conf Knowledge Disc. Data Mining (KDD)*, pages 275–280, 2001.

[10] C. Ding, X. He, H. Zha, M. Gu, and H. Simon. A min-max cut algorithm for graph partitioning and data clustering. *Proc. IEEE Int'l Conf. Data Mining*, 2001.

[11] W.E. Donath and A. J. Hoffman. Lower bounds for partitioning of graphs. *IBM J. Res. Develop.*, 17:420–425, 1973.

[12] R. V. Driessche and D. Roose. An improved spectral bisection algorithm and its application to dynamic load balancing. *Parallel Computing*, 21:29–48, 1995.

[13] P. Drineas, A. Frieze, R. Kannan, S. Vempala, and V. Vinay. Clustering in large graphs and matrices. In *Proc. 19th ACM-SIAM Symposium on Discrete Algorithms*, 1999.

[14] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification, 2nd ed.* Wiley, 2000.

[15] C.M. Fiduccia and R.M. Mattheyses. A linear time heuristic for improving network partitions. *Proc. 19th IEEE Design Automation Conference*, pages 175–181, 1982.

[16] M. Fiedler. Algebraic connectivity of graphs. *Czech. Math. J.*, 23:298–305, 1973.

[17] M. Fiedler. A property of eigenvectors of non-negative symmetric matrices and its application to graph theory. *Czech. Math. J.*, 25:619–633, 1975.

[18] M. Gu, H. Zha, C. Ding, X. He, and H. Simon. Spectral relaxation models and structure analysis for k-way graph clustering and bi-clustering. *Penn State Univ Tech Report CSE-01-007*, 2001.

[19] L. Hagen and A.B. Kahng. New spectral methods for ratio cut partitioning and clustering. *IEEE. Trans. on Computed Aided Desgin*, 11:1074–1085, 1992.

[20] J. Han and M. Kamber. *Data Mining: Concepts and Techniques.* Morgan Kaufmann, 2000.

[21] J. Hartigan. *Clustering Algorithms.* John Wiley & Sons, 1975.

[22] J.A. Hartigan and M.A. Wang. A K-means clustering algorithm. *Applied Statistics*, 28:100–108, 1979.

[23] T. Hastie, R. Tibshirani, and J. Friedman. *Elements of Statistical Learning.* Springer Verlag, 2001.

[24] B. Hendrickson and R. Leland. Chaco mesh partitioning software. *http://www.cs.sandia.gov/CRF/chac.html.*

[25] A.K. Jain and R.C. Dubes. *Algorithms for clustering data.* Prentice Hall, 1988.

[26] G. Karypis and V. Kumar. Metis graph partitioning software. *http://www-users.cs.umn.edu/ karypis/metis/.*

[27] S.P. Lloyd. Least squares quantization in pcm. *Bell Telephone Laboratories Paper, Marray Hill*, 1957.

[28] J. MacQueen. Some methods for classification and analysis of multivariate observations. *Proc. 5th Berkeley Symposium*, pages 281–297, 1967.

[29] A. McCallum. Bow: A toolkit for statistical language modeling, text retrieval, classification and clustering. *http://www.cs.cmu.edu/mccallum/bow*, 1996.

[30] G. McLachlan and T. Krishnan. *The EM Algorithm and Extensions.* John Wiley, 1997.

[31] A.Y. Ng, M.I. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. *Proc. Neural Info. Processing Systems (NIPS 2001)*, 2001.

[32] B. N. Parlett. *The Symmetric Eigenvalue Problem.* SIAM Press, 1998.

[33] A. Pothen, H. D. Simon, and K. P. Liou. Partitioning sparse matrices with egenvectors of graph. *SIAM Journal of Matrix Anal. Appl.*, 11:430–452, 1990.

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

[35] H. Zha, C. Ding, M. Gu, X. He, and H.D. Simon. Spectral relaxation for K-means clustering. *Advances in Neural Information Processing Systems 14*, pages 1057–1064, 2002.

[36] H. Zha, X. He, C. Ding, M. Gu, and H.D. Simon. Bipartite graph partitioning and data clustering. *Proc. Int'l Conf. Information and Knowledge Management (CIKM 2001)*, 2001.

[37] Y. Zhao and G. Karypis. Criterion functions for document clustering: Experiments and analysis. *Univ. Minnesota, CS Dept. Tech Report #01-40*, 2001.