Clustering with Semidefinite Programming and Fixed Point Iteration

Pedro Felzenszwalb   PFF@BROWN.EDU
School of Engineering and Department of Computer Science
Brown University
Providence, RI 02912, USA

Caroline Klivans   KLIVANS@BROWN.EDU
Division of Applied Mathematics
Brown University
Providence, RI 02912, USA

Alice Paul   ALICE_PAUL@BROWN.EDU
Department of Biostatistics
Brown University
Providence, RI 02912, USA

Editor: Silvia Villa

Abstract

We introduce a novel method for clustering using a semidefinite programming (SDP) relaxation of the Max k-Cut problem. The approach is based on a new methodology for rounding the solution of an SDP relaxation using iterated linear optimization. We show the vertices of the Max k-Cut relaxation correspond to partitions of the data into at most k sets. We also show the vertices are attractive fixed points of iterated linear optimization. Each step of this iterative process solves a relaxation of the closest vertex problem and leads to a new clustering problem where the underlying clusters are more clearly defined. Our experiments show that using fixed point iteration for rounding the Max k-Cut SDP relaxation leads to significantly better results when compared to randomized rounding.

Keywords: Clustering, Semidefinite programming, Optimization.

1. Introduction

Semidefinite programming (SDP) relaxations have led to significant advances in the development of combinatorial optimizaton algorithms. This includes a variety of problems with applications to machine learning. Many challenging optimization problems can be approximately solved by a combination of an SDP relaxation and a rounding step. One of the best examples of this paradigm is the celebrated Max Cut approximation algorithm of Goemans and Williamson (1995).

From a theoretical point of view algorithms based on SDP relaxations can lead to strong approximation guarantees. However, such approximation guarantees do not always translate to practical solutions. Many algorithms with good theoretical guarantees rely on randomized rounding methods that can produce solutions that have undesirable artifacts despite having high objective value. This motivates the development of effective deterministic methods for

rounding the solutions of SDP relaxations. Recent advances based on the sum-of-squares hierarchy have also motivated the development of new general methods for rounding the solutions of SDP relaxations (see, e.g., Barak et al. (2014)).

In this paper we introduce a novel method for clustering using the Max k-Cut SDP relaxation described in Frieze and Jerrum (1997). The approach is based on a new methodology for rounding the solution of an SDP relaxation introduced by the current authors in Felzenszwalb et al. (2021). Our rounding method involves fixed point iteration with a map that optimizes a linear function over a convex body. Figure 1 shows a clustering example, comparing the result of our fixed point iteration method for rounding the solution of the Max k-Cut relaxation to the result obtained using the randomized rounding method in Frieze and Jerrum (1997).

The SDP relaxation for Max k-Cut involves linear optimization over a convex body that we call the k-way elliptope. In Section 4 we show that the vertices of the k-way elliptope correspond to partitions (clusterings) of the data into at most k sets, generalizing the result from Laurent and Poljak (1995) for the elliptope (the $k = 2$ case).

As pointed out already by the authors of Frieze and Jerrum (1997), the randomized rounding method for the Max k-Cut SDP relaxation has some significant shortcomings. The approximation factor of the randomized algorithm appears good on the surface, but is not much better than the approximation factor one gets by simply randomly partitioning the data. Randomized rounding often generates a partition with fewer than k sets. The result in Figure 1(b) is a partition with 7 non-empty clusters despite the fact that $k = 8$. We also see in Figure 1(b) that the resulting clusters are not compact. Instead different clusters have significant overlap. In Section 6 we compare our fixed point iteration method to the randomized rounding procedure in several examples, showing that the fixed point approach can produce much better clusterings in practice.

The work in Mahajan and Ramesh (1999) gives a general method for derandomizing approximation algorithms based on SDP relaxations, including the Max k-Cut relaxation we use for clustering. Although the method in Mahajan and Ramesh (1999) is interesting from a theoretical point of view, the approach is not practical for problems of non-trivial size. More recent methods for derandomizing the Max Cut approximation algorithm include Engebretsen et al. (2002) and Bhargava and Kosaraju (2005). These methods are all based on the randomized rounding method of Goemans and Williamson (1995) but replace the randomization with a search over a limited number of discretized choices. In contrast, our fixed point iteration method is not based on derandomization techniques and is instead based on a novel approach for rounding the solution of an SDP relaxation.

Intuitively the problem of rounding a solution of the SDP relaxation for Max k-Cut can be interpreted as a new clustering problem. This motivates the iterative nature of our algorithm. Our rounding procedure solves a sequence of SDP problems where the underlying clusters become more clearly defined in each iteration. In Felzenszwalb et al. (2021) we showed that iterated linear optimization in a convex region always converges to a fixed point. In the case of the k-way elliptope the integer solutions to the Max k-Cut problem are attractive fixed points. Additionally, when rounding the solution of the SDP relaxation, we ideally round to the closest vertex (k-way partition). We show in Section 5 that each iteration of our rounding procedure corresponds to a relaxation of this objective.
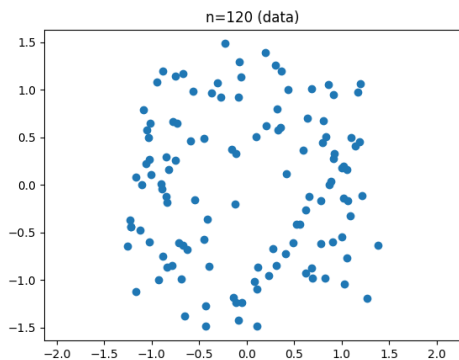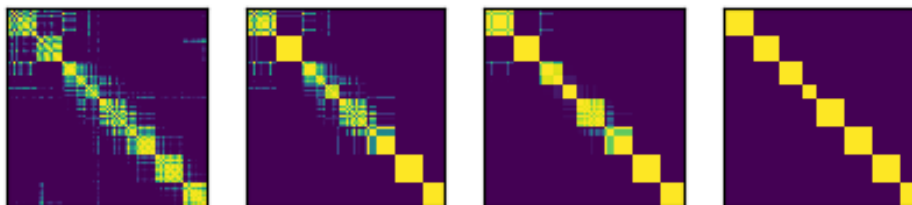
(a) Input data
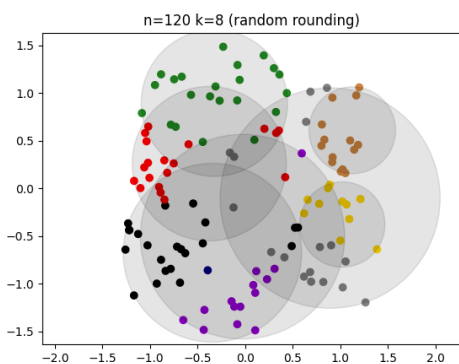
(b) Randomized rounding (best of 50 trials)
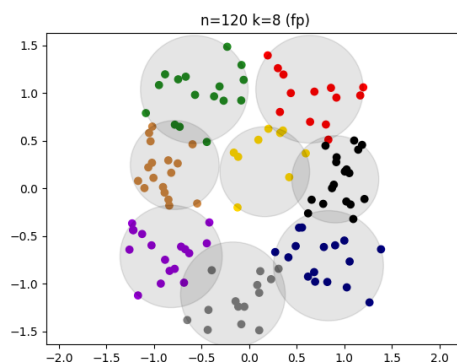
(c) Fixed point iteration

(d) Sequence of solutions generated by fixed point iteration

Figure 1: Clustering 120 points into 8 clusters using the Max k-Cut SDP relaxation: (a) input data, (b) clustering using randomized rounding, and (c) clustering using fixed point iteration. Each cluster is shown with a different color and a minimal enclosing circle. A solution to the Max k-Cut SDP relaxation is a matrix in the k-way elliptope. (d) illustrates the sequence of matrices obtained by fixed point iteration. The final matrix is an integer solution defining a clustering.

Mixon et al. (2017) also solve an SDP relaxation to cluster subgaussian mixtures. However, rather than round this SDP solution directly, they revert back to employing Lloyd's algorithm to partition the columns of the solution. Our method, instead, continues to use the strength of semidefinite programming to recover a partition.

For some applications convex relaxations have been shown to recover the "true" hidden structure in the data (see, e.g., Candès and Tao (2010)). For clustering applications it was shown in Awasthi et al. (2015) and Mixon et al. (2017) that convex relaxations can recover a ground truth clustering if the data is sufficiently well-separated. However, in practice the data is rarely well-separated (and there is often no ground truth clustering). Nonetheless, good clusterings might exist that can be extremely useful for data processing, coding or analysis. The data in Figure 1 illustrates an example of this situation. The data was generated by sampling from several Gaussian distributions with significant overlap. In this case there is no way to recover the ground truth clustering, but the result of our fixed point iteration method still provides a good solution that can be used for subsequent analysis.

In Section 2 we discuss how Max k-Cut can be used to formulate the clustering problem. In Section 3 we review the Max k-Cut SDP relaxation and the randomized rounding method from Frieze and Jerrum (1997). In Section 4 we study the convex body that arises from the SDP relaxation and show the vertices of the feasible region correspond to partitions. In Section 5 we describe how iterated linear optimization leads to a deterministic method for rounding the solution of the SDP relaxation. The approach solves a sequence of relaxations to the closest vertex problem, and leads to clusters that are more clearly defined in each iteration. Finally, in Section 6 we illustrate experimental results of our new rounding method and compare them to the randomized rounding approach from Frieze and Jerrum (1997). We also compare the result of our method to the k-means algorithm for clustering images in the MNIST handwritten digit dataset.

2. Clustering with Max k-Cut

Clustering problems are often formulated using pairwise measures of similarity or dissimilarity between objects. Intuitively we would like to partition the data so that pairs of objects within a cluster are similar to each other while pairs of objects in different clusters are dissimilar. This natural idea leads to a variety of formulations of clustering as graph partition problems. One benefit of graph-based approaches to clustering is the ability to define pairwise measures that incorporate both categorical and numerical variables (see, e.g. Bertsimas et al. (2021)).

Let $G = (V, E)$ be a weighted graph. Let $[n] = \{1, \ldots, n\}$. To simplify notation we assume throughout that $V = [n]$ and that the graph is complete. A k-partition is a partition of V into k disjoint sets $(A_1, \ldots, A_k)$, some of which may be empty. Let M be a symmetric matrix of pairwise non-negative weights. The weight of a partition P is the sum of the weights of the pairs $\{i, j\} \subseteq [n]$ that are split (or cut) by P,

$$w(P) = \sum_{1 \leq r < s \leq k} \sum_{i \in A_r} \sum_{j \in A_s} M_{i,j}.$$

The Max k-Cut problem is to find a k-partition maximizing $w(P)$.

As an example, let $D = \{x_1, \ldots, x_n\}$ be n points in $\mathbb{R}^d$. One of the most commonly used formulations for clustering data in Euclidean space involves optimizing the k-means objective. For $A \subseteq [n]$ let $m(A)$ be the mean of the points indexed by A,

$$m(A) = \frac{1}{|A|} \sum_{i \in A} x_i.$$

The k-means objective (1) is to partition the data into clusters to minimize the sum of squared distances from each point to the center of its cluster,

$$\operatorname*{argmin}_{(A_1,\ldots,A_k)} \sum_{r=1}^{k} \sum_{i \in A_r} ||x_i - m(A_i)||^2. \tag{1}$$

This objective encourages partitions of the data into "compact" clusters, and is widely used both for clustering and for vector quantization in coding applications (see, e.g., Lloyd (1982); MacKay (2003); Jain and Dubes (1988); Arthur and Vassilvitskii (2007)).

Let $M_{i,j} = ||x_i - x_j||^2$ and consider the Max k-Cut objective with these weights. Maximizing the weight of the pairs $\{i,j\}$ that are split by a partition is the same as minimizing the weights of the pairs $\{i,j\}$ that are not split. Moreover, the sum of squared distances between points within a set A_i can be expressed in terms of the sum of squared distances between each point in A_i and the mean of the set:

$$\operatorname{argmax} w(A_1, \ldots, A_k) = \operatorname*{argmin}_{(A_1,\ldots,A_k)} \sum_{r=1}^{k} \sum_{i,j \in A_r} ||x_i - x_j||^2, \tag{2}$$

$$= \operatorname*{argmin}_{(A_1,\ldots,A_k)} \sum_{r=1}^{k} |A_r| \sum_{i \in A_r} ||x_i - m(A_r)||^2. \tag{3}$$

The objective function (3) is similar to the k-means objective (1), except that in the case of Max k-Cut there is a preference towards balanced partitions. In Section 6 we illustrate the results of clustering experiments with Max k-Cut using this formulation.

The work in Wang and Sha (2011) also considered clustering with Max k-Cut and used an SDP relaxation to solve the resulting problem. That work focused on an information theoretical formulation of the clustering problem that could also be used within our framework. SDP relaxations of the k-means objective for clustering have also been considered in Peng and Wei (2007), Awasthi et al. (2015), and Mixon et al. (2017).

Most of the previous work on clustering using graph-based methods has focused on formulations based on minimum cuts and spectral algorithms (Shi and Malik (2000); Meila and Shi (2001); Ng et al. (2002); Weiss (1999); Kannan et al. (2004)). In this case the weight of an edge represents similarity (or affinity) between elements instead of dissimilarity. Minimum cut formulations often include some form of normalization (see, e.g., Shi and Malik (2000); Kannan et al. (2004)) or a balance requirement to avoid trivial partitions (otherwise the cut is minimized when one partition is very small). In contrast to minimum cut formulations, clustering using a maximum cut formulation naturally encourages balanced partitions, as they maximize the total number of edges that are split.

3. SDP Relaxation for Max k-Cut

The Goemans and Williamson (1995) approximation algorithm for Max Cut (clustering into *two* clusters) is based on an SDP relaxation and a randomized rounding method. The relaxation involves the optimization of a linear function over a convex body $\mathcal{L}_n$ known as the *elliptope*. Laurent and Poljak (1995) showed that the vertices of $\mathcal{L}_n$ correspond to bipartitions of $[n]$. The fact that the vertices of $\mathcal{L}_n$ are bipartitions gives an explanation as to why in some cases the SDP relaxation can lead directly to an integer solution that is an *optimal* solution to the Max Cut problem (see also Cifuentes et al. (2020)).

Frieze and Jerrum (1997) generalized the approximation algorithm of Goemans and Williamson to a randomized approximation method for Max k-Cut. In this case the algorithm involves linear optimization over a convex body $\mathcal{L}_{n,k}$ that we call the *k-way elliptope*.

Below we briefly review the SDP relaxation and randomized rounding method for Max k-Cut introduced by Frieze and Jerrum (1997). In Section 4 we show the vertices of the k-way elliptope correspond to k-partitions, generalizing the result from Laurent and Poljak (1995) for the elliptope. In Section 5 we describe a determinisic method for rounding the solution of the SDP relaxation for Max k-Cut based on iterated linear optimzation.

The Max k-Cut SDP relaxation is based on a reformulation of the combinatorial problem in terms of Gram matrices. Let $a_1, \ldots, a_k$ be the vertices of an equilateral simplex Σ_k in $\mathbb{R}^{k-1}$ centered around the origin and scaled such that $||a_i|| = 1$.

A k-partition $P = (A_1, \ldots, A_k)$ can be encoded by n vectors $(y_1, \ldots, y_n)$ with $y_i = a_j$ if $i \in A_j$. Define the k-partition matrix X^P to be the Gram matrix of $(y_1, \ldots, y_n)$. For $i \neq j$ we have $a_i \cdot a_j = -1/(k-1)$. Therefore,

$$X_{i,j}^P = \begin{cases} 1 & \text{if } \{i,j\} \text{ are together in } P \\ -1/(k-1) & \text{if } \{i,j\} \text{ are split by } P \end{cases} \tag{4}$$

$$1 - X_{i,j}^P = \begin{cases} 0 & \text{if } \{i,j\} \text{ are together in } P \\ k/(k-1) & \text{if } \{i,j\} \text{ are split by } P \end{cases} \tag{5}$$

For two n by n matrices X and Y let

$$X \cdot Y = \sum_{i=1}^{n} \sum_{j=1}^{n} X_{i,j} Y_{i,j}.$$

Now the weight of a partition, $w(P)$, can be written as,

$$w(P) = \frac{k-1}{2k}(1 - X^P) \cdot M,$$

where M is the symmetric matrix of pairwise weights.

Definition 1 *The set of k-partition matrices $\mathcal{Q}_{n,k}$ is the set of Gram matrices of n vectors $(y_1, \ldots, y_n)$ with $y_i \in \{a_1, \ldots, a_k\}$ for $1 \leq i \leq n$.*

We can reformulate the Max k-Cut problem as an optimization over k-partiton matrices,

$$\operatorname*{argmax}_{X \in \mathcal{Q}_{n,k}} \frac{k-1}{2k}(1 - X) \cdot M.$$

6

The SDP relaxation of Max k-Cut is based on relaxing the requirement that $y_i \in \{a_1, \ldots, a_n\}$ in the definition of $\mathcal{Q}_{n,k}$ to allow y_i to be any unit vector in $\mathbb{R}^n$, with the additional constraint that $y_i \cdot y_j \geq -1/(k-1)$.

Let $\mathcal{S}(n) \subset \mathbb{R}^{n \times n}$ be the set of $n \times n$ symmetric matrices.

Definition 2 *The* elliptope $\mathcal{L}_n$ *is the subset of matrices in* $\mathcal{S}(n)$ *that are positive semidefinite and have all 1's on the diagonal:*

$$\mathcal{L}_n = \{X \in \mathcal{S}(n) \mid X \succeq 0, X_{i,i} = 1\}.$$

The matrices in $\mathcal{L}_n$ exactly correspond to Gram matrices of n unit vectors $(y_1, \ldots, y_n)$ in $\mathbb{R}^n$ (Goemans and Williamson (1995); Laurent and Poljak (1995)).

Definition 3 *The* k-way elliptope $\mathcal{L}_{n,k}$ *is the subset of matrices in* $\mathcal{L}_n$ *where every entry is at least* $-1/(k-1)$:

$$\mathcal{L}_{n,k} = \{X \in \mathcal{L}_n \mid X_{i,j} \geq -1/(k-1)\}.$$

The matrices in $\mathcal{L}_{n,k}$ correspond to Gram matrices of n unit vectors $(y_1, \ldots, y_n)$ in $\mathbb{R}^n$ with $y_i \cdot y_j \geq -1/(k-1)$.

The randomized algorithm in Frieze and Jerrum (1997) involves the SDP relaxation,

$$\underset{X \in \mathcal{L}_{n,k}}{\operatorname{argmax}} \frac{k-1}{2k}(1 - X) \cdot M.$$

Let X be an optimal solution to the SDP relaxation. We can interpret X as the Gram matrix of n unit vectors $(y_1, \ldots, y_n)$ in $\mathbb{R}^n$, obtained using a Cholesky decomposition $X = VV^T$. To generate a k-partition the rounding method in Frieze and Jerrum (1997) selects k unit vectors $(u_1, \ldots, u_k)$ independently from a uniform distribution and assigns y_i to the closest vector u_j.

When $k > 2$ rounding a k-partition matrix X^P using the randomized procedure above can generate a partition that is different from P because different sets in P can be merged. More generally randomized rounding often generates a partition with fewer than k sets because some vector u_j is not the closest vector to any v_i. In Section 6 we show experimental results that illustrate several problems that arise in practice when using the randomized rounding procedure. Even when the cut value of the random partition is relatively high, the resulting clustering can have undesirable artifacts.

4. The k-way elliptope

The main result of this section is that the vertices of $\mathcal{L}_{n,k}$ are the matrices in $\mathcal{Q}_{n,k}$ and correspond to k-partitions of $[n]$. Note that a k-partition may have some empty sets, so the vertices of $\mathcal{L}_{n,k}$ correspond to partitions of $[n]$ into at most k sets.

For a matrix $X \in \mathcal{L}_{3,k}$ we have

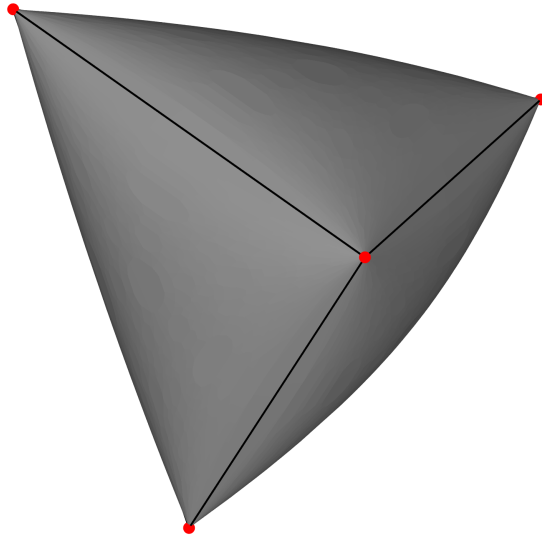$$X = \begin{pmatrix} 1 & x & y \\ x & 1 & z \\ y & z & 1 \end{pmatrix}.$$

Figure 2: The 2-way elliptope $\mathcal{L}_{3,2}$ has 4 vertices (red points), corresponding to partitions of 3 distinguished elements into at most 2 sets.
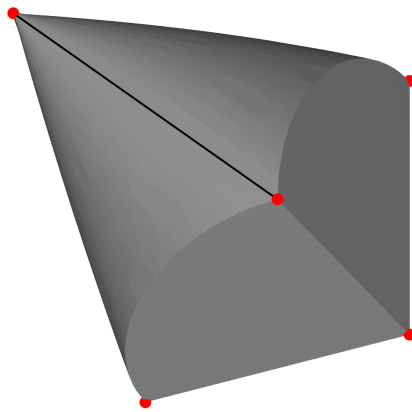

Figure 3: The 3-way elliptope $\mathcal{L}_{3,3}$ has 5 vertices (red points), corresponding to partitions of 3 distinguished elements into at most 3 sets.

Therefore we can visualize X as a point $(x, y, z) \in \mathbb{R}^3$.

Figure 2 illustrates $\mathcal{L}_{3,2}$. This convex body has 4 vertices, with one vertex for each partition of 3 distinguished elements into at most 2 sets. The partitions are listed below.

$P_1 = (\{1, 2, 3\}, \emptyset)$
$P_2 = (\{1, 2\}, \{3\})$
$P_3 = (\{1, 3\}, \{2\})$
$P_4 = (\{2, 3\}, \{1\})$

Figure 3 illustrates $\mathcal{L}_{3,3}$. This convex body has 5 vertices, with one vertex for each partition of 3 distinguished elements into at most 3 sets. The partitions are listed below.

$P_1 = (\{1, 2, 3\}, \emptyset, \emptyset)$
$P_2 = (\{1, 2\}, \{3\}, \emptyset)$
$P_3 = (\{1, 3\}, \{2\}, \emptyset)$
$P_4 = (\{2, 3\}, \{1\}, \emptyset)$
$P_5 = (\{1\}, \{2\}, \{3\})$

Note that $\mathcal{L}_{n,k}$ is simply the elliptope intersected with an orthant. This characterization is useful to understand the geometric and combinatorial structure of $\mathcal{L}_{n,k}$. The difference between $\mathcal{L}_{n,r}$ and $\mathcal{L}_{n,s}$ is the amount by which the intersecting orthant is translated.

If we translate the orthant continuously from 0 to $-1/(n-1)$ the vertex that represents the grouping of all elements into a single set remains fixed. This vertex is the matrix of all 1's. On the other hand, the vertex that represents the partition of all elements into different sets only appears when the orthant reaches $-1/(n-1)$. This vertex is the n-partition matrix with 1's on the diagonal and $-1/(n-1)$ in the off diagonal entries.

Let Δ be a convex subset of $\mathbb{R}^n$. For $x \in \Delta$, the *normal cone of Δ at x* is the set

$$N(\Delta, x) = \{y \in \mathbb{R}^n \mid y \cdot x \geq y \cdot z \ \ \forall z \in \Delta\}.$$

A vertex of Δ is (by definition) a point with a full-dimensional normal cone.

The following result shows that the normal cone of $\mathcal{L}_{n,k}$ at a k-partition matrix is full-dimensional.

Proposition 4 *If $X \in \mathcal{Q}_{n,k}$ and $||Y - X|| < 1/(k-1)$ then $Y \in N(\mathcal{L}_{n,k}, X)$.*

Proof If $||Y - X|| < 1/(k-1)$ then $|(Y - X)_{i,j}| < 1/(k-1)$ for all $\{i, j\} \subseteq [n]$. Since $X_{i,j}$ is either 1 or $-1/(k-1)$ we can see that Y has the same sign pattern as X. If $Z \in \mathcal{L}_{n,k}$ all of the entries in Z are between $-1/(k-1)$ and 1. Therefore $Y \cdot X \geq Y \cdot Z$. ∎

The next proposition shows that k-partition matrices are the only matrices in $\mathcal{L}_{n,k}$ where every entry is either $-1/(k-1)$ or 1.

Proposition 5 *If $X \in \mathcal{L}_{n,k}$ and $X_{i,j} \in \{-1/(k-1), 1\}$ for all $\{i, j\} \subseteq [n]$ then $X \in \mathcal{Q}_{n,k}$.*

Proof Suppose $X_{i,j} \in \{-1/(k-1), 1\}$ for all $\{i, j\} \subseteq [n]$. Since X is a Gram matrix we know that $X_{i,j} = 1$ defines an equivalence relation on $[n]$. Let $(A_1, \ldots, A_l)$ be the equivalence classes of this relationship. If $i \in A_r$ and $j \in A_s$ with $r \neq s$ then $X_{i,j} = -1/(k-1)$. This means we have l unit vectors with the dot product between each pair equal to $-1/(k-1)$. Lemma 4 in Frieze and Jerrum (1997) implies that $l \leq k$. We conclude $X \in \mathcal{Q}_{n,k}$. ∎

9

Let
$$T(M) = \operatorname*{argmax}_{X \in \mathcal{L}_n} M \cdot X.$$

Note that the argmax in the definition of T may not be unique. In this case $T(M)$ is set valued. When we write $X = T(M)$ we allow X to be *any* element of $T(M)$.

We will use the following lemma from Felzenszwalb et al. (2021).

Lemma 6 *Let $M \in \mathcal{S}(n)$ and $X = T(M)$.*

Suppose X is the Gram matrix of n unit vectors $(v_1, \ldots, v_n)$. Then,

(a) *There exists real values α_i such that*

$$\sum_{j \neq i} M_{i,j} v_j = \alpha_i v_i.$$

(b) *The vectors $(v_1, \ldots, v_n)$ are linearly dependent and $\operatorname{rank}(X) < n$.*

(c) *There exists a diagonal matrix D such that,*

$$MX = DX.$$

Now we are ready to prove the main result of this section, showing that the vertices of $\mathcal{L}_{n,k}$ are in fact the integer solutions to the Max k-Cut problem.

Theorem 7 *The vertices of $\mathcal{L}_{n,k}$ are the k-partition matrices.*

Proof Suppose $X \in \mathcal{Q}_{n,k}$. Proposition 4 implies $N(\mathcal{L}_{n,k}, X)$ is full-dimensional.

Now suppose $X \notin \mathcal{Q}_{n,k}$. Proposition 5 implies that X must have at least one entry $X_{i,j} \notin \{-1/(k-1), 1\}$ Since $\mathcal{L}_{n,k} \subseteq \mathcal{L}_n$ we know $X \in \mathcal{L}_n$. Note that $X_{i,j} \notin \{-1, 1\}$. Let J be a matrix with $J_{i,j} = J_{j,i} = 1$ and $J_{k,l} = 0$ for $\{k, l\} \neq \{i, j\}$.

Let $\mathcal{O}_{n,k} = \{X \mid X_{i,j} \geq -1/(k-1)\}$. Since $\mathcal{L}_{n,k} = \mathcal{L}_n \cap \mathcal{O}_{n,k}$ we have

$$N(\mathcal{L}_{n,k}, X) = \{v + u, \mid v \in N(\mathcal{L}_n, X), u \in N(\mathcal{O}_{n,k}, X)\}.$$

We consider $N(\mathcal{L}_n, X)$ and $N(\mathcal{O}_{n,k}, X)$ separately. We show J is not in the linear span of $N(\mathcal{L}_n, X)$ and J is orthogonal to $N(\mathcal{O}_{n,k}, X)$. This implies J is not in the linear span of $N(\mathcal{L}_{n,k}, X)$ and therefore $N(\mathcal{L}_{n,k}, X)$ is not full-dimensional.

For the sake of contradiction suppose J is in the linear span of $N(\mathcal{L}_n, X)$. Then $J = R - M$ with R and M both in $N(\mathcal{L}_n, X)$. This implies $M + J = R$ and $M + J \in N(\mathcal{L}_n, X)$. Since $M \in N(\mathcal{L}_n, X)$ we have $X = T(M)$ and by the lemma above X is the Gram matrix of $(v_1, \ldots, v_n)$ where,

$$v_i \propto \sum_{k \neq i} M_{i,k} v_k.$$

Since $M + J \in N(\mathcal{L}_n, X)$ we have $X = T(M + J)$ and the lemma also implies,

$$v_i \propto \sum_{k \neq i} (M_{i,k} + J_{i,k}) v_k \Rightarrow v_i \propto v_j + \sum_{k \neq i} M_{i,k} v_k.$$

10

Therefore

$$v_i \propto v_j.$$

Since $X_{i,j} \notin \{-1, 1\}$ the unit vectors v_i and v_j can not be proportional and we have a contradiction. Therefore J is not in the linear span of $N(\mathcal{L}_n, X)$.

Now note

$$N(\mathcal{O}_{n,k}, X) = \text{cone}(\{e^{r,s} \mid X_{r,s} = -1/(k-1)\}),$$

where $e^{r,s}$ is the matrix that is 0 everywhere except in the (r, s) position which has value 1. The dot product $J \cdot e^{r,s}$ equals 0 for all (r, s) with $X_{r,s} = -1/(k-1)$. Therefore $N(\mathcal{O}_{n,k}, X)$ is orthogonal to J. ∎

5. Iterated linear optimization and rounding

A key step in solving a combinatorial optimization problem via a convex relaxation involves *rounding* a solution of the convex relaxation to a solution of the original combinatorial optimization problem. In the case of Max k-Cut this involves mapping a solution $X \in \mathcal{L}_{n,k}$ to a k-partition matrix $Y \in \mathcal{Q}_{n,k}$.

Recall that $X \in \mathcal{L}_{n,k}$ is the Gram matrix of n unit vectors $(v_1, \ldots, v_n)$. The problem of rounding X can be seen as a new clustering problem, where we would like to partition $(v_1, \ldots, v_n)$ into k sets. To solve this clustering problem we look for $Y \in \mathcal{Q}_{n,k}$ that is close to X. Relaxing this problem to $\mathcal{L}_{n,k}$ we obtain a new SDP with solution $X' \in \mathcal{L}_{n,k}$. Our rounding method involves repeating this process multiple times. The process is guaranteed to converge and the underlying unit vectors become more clearly clustered with each step.

5.1 Iterated linear optimization

Let $\Delta \subset \mathbb{R}^n$ be a compact convex subset containing the origin. Let $T(x)$ be the map defined by linear optimization over Δ,

$$T(x) = \underset{y \in \Delta}{\text{argmax}}\, x \cdot y.$$

In Felzenszwalb et al. (2021) it was shown that fixed point iteration with T always converges to a fixed point of T. Furthermore, when Δ is the elliptope, $T(X)$ solves a relaxation to the closest vertex problem. Here we derive a similar result for the k-way elliptope, and show that iterated linear optimization in $\mathcal{L}_{n,k}$ can be used to round a solution to the SDP relaxation for Max k-Cut.

5.2 Deterministic rounding in $\mathcal{L}_{n,k}$

Let $X \in \mathcal{L}_{n,k}$ be a solution to the SDP relaxation of a Max k-Cut problem. If $X \in \mathcal{Q}_{n,k}$ then X defines k-partition. Otherwise we look for $Y \in \mathcal{Q}_{n,k}$ that is closest to X. By relaxing this problem we obtain a new SDP. Solving the new SDP leads to a new solution $Y \in \mathcal{L}_{n,k}$. If $Y \in \mathcal{Q}_{n,k}$ then Y is the closest k-partition matrix to X. Otherwise we recursively look for a matrix $Z \in \mathcal{Q}_{n,k}$ that is closest to Y. The approach leads to a fixed point iteration process with a map T' that optimizes a linear function over $\mathcal{L}_{n,k}$.

Let $a = (1 - k/2)/(k - 1)$ and A be the matrix where every entry equals a. If $Y \in \mathcal{Q}_{n,k}$ then all entries in Y are in $\{-1/(k-1), 1\}$ and all entries in $Y + A$ are in

$$\left\{ -\frac{k/2}{k-1}, \frac{k/2}{k-1} \right\}.$$

Therefore $(Y + A) \cdot (Y + A)$ is constant.

Let $Y \in \mathcal{Q}_{n,k}$ and consider the following expansion,

$$\begin{aligned}
||X - Y||^2 &= ||(X + A) - (Y + A)||^2 \\
&= (X + A) \cdot (X + A) + (Y + A) \cdot (Y + A) - 2(X + A) \cdot (Y + A).
\end{aligned}$$

Note that $(X + A) \cdot (X + A)$ does not depend on Y and $(Y + A) \cdot (Y + A)$ is constant. Therefore the closest k-partition matrix to X is,

$$\begin{aligned}
Y = \underset{Y \in \mathcal{Q}_{n,k}}{\operatorname{argmin}} ||X - Y||^2 &= \underset{Y \in \mathcal{Q}_{n,k}}{\operatorname{argmax}} (X + A) \cdot (Y + A), \\
&= \underset{Y \in \mathcal{Q}_{n,k}}{\operatorname{argmax}} (X + A) \cdot Y.
\end{aligned}$$

Relaxing this problem to $\mathcal{L}_{n,k}$ we obtain $Y = T(X + A)$ where T is defined over $\Delta = \mathcal{L}_{n,k}$.

Let $T'(X) = T(X + A)$. That is,

$$T'(X) = \underset{Y \in \mathcal{L}_{n,k}}{\operatorname{argmax}} (X + A) \cdot Y.$$

Fixed point iteration Our rounding method involves fixed point iteration with T'. That is, we generate a sequence $\{X_t\}$ where $X_0 = X$ and,

$$X_{t+1} = T'(X_t).$$

Note that iteration with T' is equivalent to iteration with T in $\Delta = \mathcal{L}_{n,k} + A$.

Figure 4 shows the result of the fixed point iteration process in $\mathcal{L}_{n,k}$ for different values of k. In each case we start from the result of the SDP relaxation of Max k-Cut for a graph with $n = 50$ vertices and random weights. In each example fixed point iteration with T' converges to a k-partition matrix after a small number of iterations.

Figure 1 in Section 1 shows an example of the sequence of solutions generated by T' for a geometric clustering problem.

We say that a fixed point x of a map $f : \Delta \to \Delta$ is *attractive* if $\exists \epsilon > 0$ such that $||x - x_0|| < \epsilon$ implies that iteration with f starting at x_0 converges to x.

In Felzenszwalb et al. (2021) we characterized all of the fixed points of T' when $k = 2$ (in this case $T' = T$) and showed the attractive fixed points are exactly the k-partition matrices. Here we consider the case when $k \geq 2$.

Proposition 8 *The k-partition matrices are attractive fixed points of T'.*

Proof Let $X \in \mathcal{Q}_{n,k}$ be a k-partition matrix.

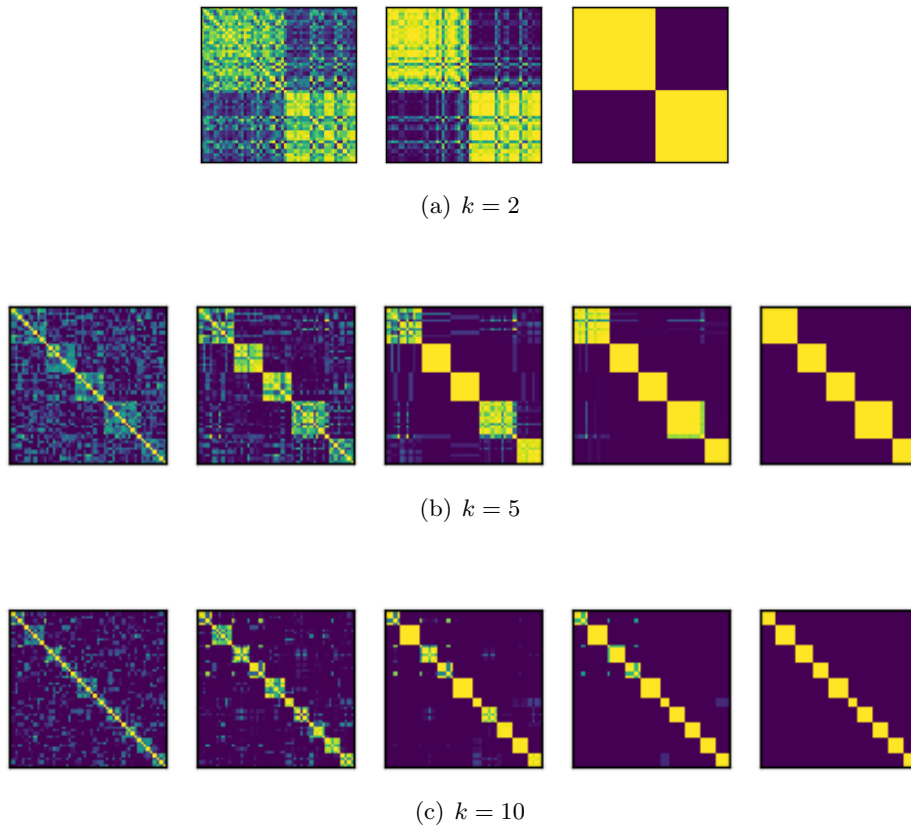(a) $k = 2$

(b) $k = 5$

(c) $k = 10$

Figure 4: Fixed point iteration with T', starting from the solution of the SDP relaxation of Max k-Cut for a graph with 50 vertices and random weights. To visualize a matrix in $\mathcal{L}_{n,k}$ we show an $n \times n$ picture with a pixel for each entry in the matrix. Bright yellow pixels indicate entries with high value, and dark blue pixels indicate entries with low value. In each case we obtain a sequence of solutions that converge to a k-partition matrix. The rows and columns in each example have been permuted so the final matrix is block diagonal to facilitate visualization.

Let $Y \in \mathcal{L}_{n,k}$ with $\|Y - X\| < (k/2)/(k-1)$. Then $|(Y + A)_{i,j} - (X + A)_{i,j}| = |(Y - X)_{i,j}| < (k/2)/(k-1)$. Since every entry in $X + A$ is either $-(k/2)/(k-1)$ or $(k/2)/(k-1)$ we see that $Y + A$ has the same sign pattern of $X + A$.

For $Z \in \mathcal{L}_{n,k}$ we have $|(Z + A)_{i,j}| \leq |(X + A)_{i,j}|$. Therefore $(Y + A) \cdot (Z + A) \leq (Y + A) \cdot (X + A)$. Moreover if $Z \neq X$ we have $(Y + A) \cdot (Z + A) < (Y + A) \cdot (X + A)$. This implies $X = T'(Y)$ and X is an attractive fixed point of T'. ∎

Define $f : \mathcal{L}_{n,k} \to \mathbb{R}$ as,

$$f(X) = (X + A) \cdot (X + A) = \sum_{i=1}^{n} \sum_{j=1}^{n} (X_{i,j} + a)^2.$$

Consider the representation of $X \in \mathcal{L}_{n,k}$ as the Gram matrix of n unit vectors $(v_1, \ldots, v_n)$ and recall that $X_{i,j} = v_i \cdot v_j \geq -1/(k-1)$.

If $v_i \cdot v_j = 1$ (the vectors are as close as possible) then $(X_{i,j} + a) = (k/2)/(k-1)$. If $v_i \cdot v_j = -1/(k-1)$ (the vectors are as far as possible) then $(X_{i,j} + a) = -(k/2)/(k-1)$. For any other choice $-(k/2)/(k-1) < (X_{i,j} + a) < (k/2)/(k-1)$.

Proposition 5 shows that k-partition matrices are exactly the matrices in $\mathcal{L}_{n,k}$ where every entry is in $\{-1/(k-1), 1\}$. Therefore the matrices that maximize $f(X)$ are the k-partition matrices. We can see $f(X)$ as a measure of how close X is to a k-partition matrix. We can also interpret $f(X)$ as a measure of how well clustered the vectors $(v_1, \ldots, v_n)$ are.

The convergence results from Felzenszwalb et al. (2021) imply the sequence $\{X_t\}$ converges to a fixed point of T'. Additionally, the results from Felzenszwalb et al. (2021) imply that $f(X)$ increases in each iteration. Therefore the vectors defining X_t become more clearly clustered in each iteration.

We have shown that matrices in $\mathcal{Q}_{n,k}$ are *attractive* fixed points of T'. Although the map T' has other fixed points, our numerical experiments suggest that these are the only attractive fixed points, and that fixed point iteration with T' starting from a generic point in $\mathcal{L}_{n,k}$ always converges to a k-partition matrix.

We have done several numerical experiments iterating T' to round a solution of the Max k-Cut relaxation. In one setting we generated 50×50 random weight matrices M with independent weights sampled from a Gaussian distribution with mean 0 and standard deviation 1 (note that in this case the matrix M may have negative weights). In another setting we generated 50×50 random weight matrices M by sampling 50 points in $\mathbb{R}^{10}$ independently from the uniform distribution over $[0,1]^{10}$. We then set $M_{ij} = ||x_i - x_j||^2$. We ran 100 different trials with $k = 5$ in each setting. Fixed point iteration with T' starting from the solution of the Max k-Cut relaxation always converged to a k-partition matrix. The number of iterations until convergence in the first setting was between 3 and 10, with an average of 7.02. The number of iterations until convergence in the second setting was between 3 and 4, with an average of 3.01.

6. Clustering Experiments

In this section we illustrate the results of our clustering method on several datasets. The algorithms were implemented in Python and run on a computer with an Intel i7 CPU @ 2.6 Ghz with 8GB of RAM. We use the cvxpy package for convex optimization together with the SCS (splitting conic solver) package to solve SDPs. The fixed point iteration process we use for rounding involves solving a sequence of SDPs identical to the Max k-Cut SDP but with a different objective. For the examples below solving each SDP took 1 to 3 minutes. The fixed point iteration method converged after 1 to 5 iterations in each case.

In each clustering experiment we have a dataset $D = \{x_1, \ldots, x_n\}$ with $x_i \in \mathbb{R}^d$. As discussed in Section 2 we cluster the data by defining a Max k-Cut problem with weights $M_{i,j} = ||x_i - x_j||^2$. With this choice of weights the maximum weight partition should lead to compact and balanced clusters, where points within a single cluster are close together while points in different clusters are far from each other. Section 6.1 illustrates clustering results on synthetic data, while Section 6.2 evaluates the results of clustering a subset of the MNIST handwritten digits (LeCun et al.).

6.1 Synthetic data

We performed several experiments using synthetic data in $\mathbb{R}^2$ to facilitate the visualization of the results. Figures 5, 6, and 7 illustrate the results of clustering a dataset of 200 points into 5, 10 and 20 clusters respectively.[1] In each case we compare the result obtained using fixed point iteration for rounding the solution of the Max k-Cut relaxation to the result of randomized rounding. For the randomized rounding method we repeat the rounding procedure 50 times and select the partition with highest weight generated over all trials.

In all of the experiments we see that the weight of the partition generated by the fixed point iteration method is higher than the weight of the best partition generated by randomized rounding. When $k = 5$ (Figure 5) the results of the two methods are similar, but the weight of the partition generated by fixed point iteration is slightly better. When $k = 10$ and $k = 20$ (Figures 6 and 7) the results of randomized rounding are significantly degraded while the results of the fixed point iteration method remain very good.

Figure 8 illustrates the partitions obtained in different trials of randomized rounding for the case when $k = 5$. We can see there is a lot of variance in the results and that the random rounding method often leads to poor clusterings even when k is relatively small. The result of our fixed point iteration method in the same data is shown in Figure 5(c).

Figure 9 shows an example where the input data has 160 points generated by sampling 20 points from 8 different Gaussian distributions. The Gaussian distributions have standard deviation $\sigma = 0.2$ and means arranged around a circle of radius 1. In this case there is a ground truth clustering where points are grouped according to the Gaussian used to generate them. The overlap between the distributions is sufficiently high that it is impossible to recover the ground truth clustering perfectly, but the result of the fixed point iteration method is closely aligned with the ground truth. The results of randomized rounding are not as good even when we select the best of 50 trials of the randomized procedure.

We repeated the last experiment 10 times to quantify the difference between the two rounding methods. Each repetition was done with a new dataset generated from the mixture of Gaussians. Table 1(a) summarizes the minimum, maximum, and mean value of the ratio $w(C)/w(D)$, where C is the partition produced by the fixed point iteration method and D is the result of the best of 50 trials of randomized rounding. In these experiments the ratio $w(C)/W(D)$ was always above one, showing that our fixed point iteration method always returns a partition with higher weight. We also compare the resulting clusterings to the ground truth using the Rand index (Rand (1971)).

The Rand index is a number between 0 and 1 measuring the agreement between two clusterings. Let A and B be clusterings. Let a be the number of pairs of elements that are in the same cluster in both A and B while b is the number of pairs of elements that are in different clusters in both A and B. The Rand index is,
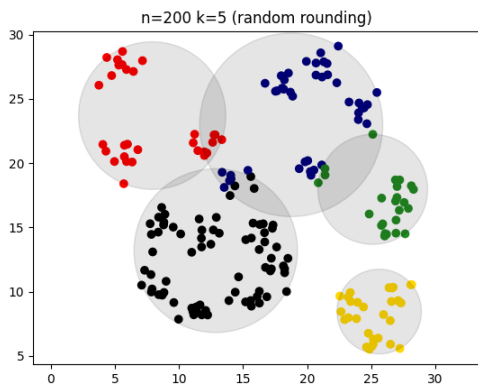
$$R(A, B) = \frac{a + b}{\binom{n}{2}}.$$

We see in Table 1(b) that our fixed point rounding method consistently produces a clustering that is more similar to the ground truth than the randomized rounding method.
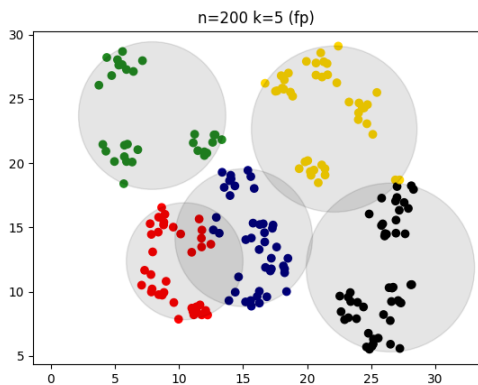
---

1. We used a subset of the D-31 dataset from Veenman et al. (2002) with 10 points from each of 20 clusters.
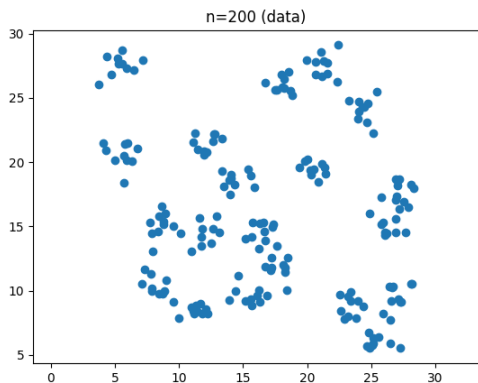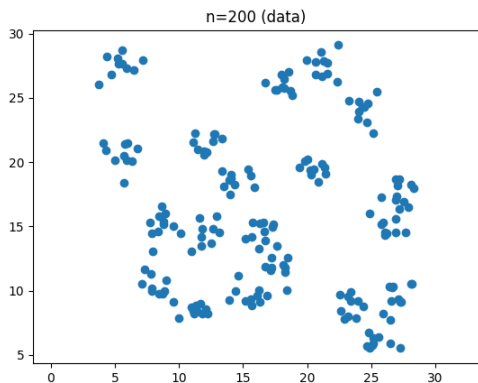
(a) Input data

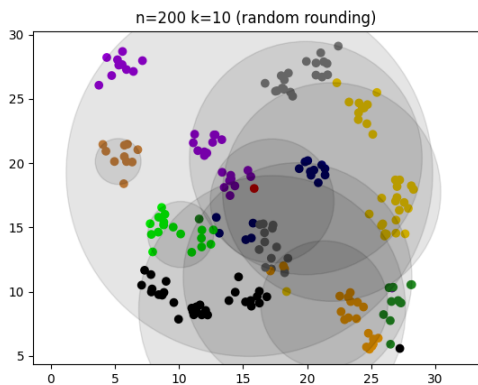(b) Randomized (best of 50) $w(C) = 3543294$

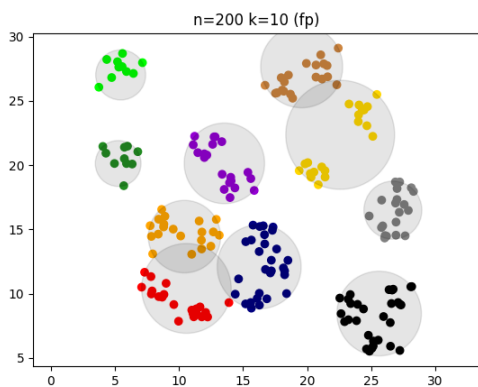

(c) Fixed point iteration $w(C) = 3589259$

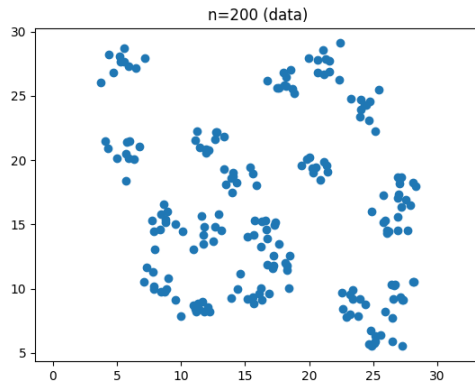Figure 5: Clustering 200 points with $k = 5$.

(a) Input data

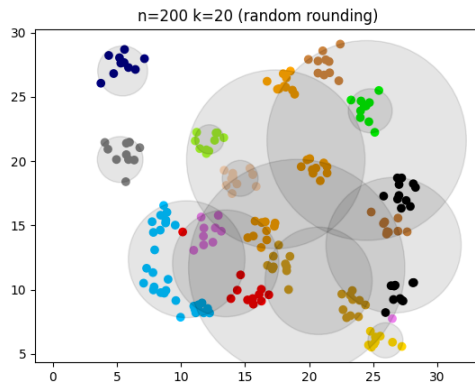(b) Randomized (best of 50) $w(C) = 3587153$

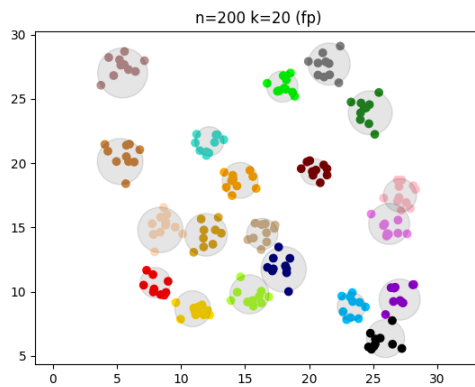(c) Fixed point iteration $w(C) = 3701677$

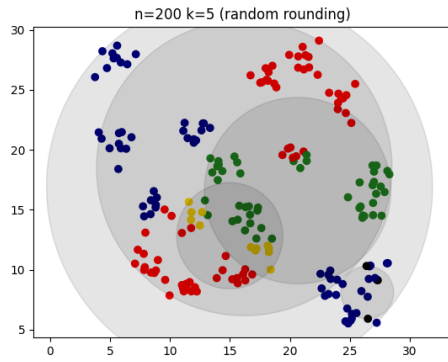Figure 6: Clustering 200 points with $k = 10$.

(a) Input data

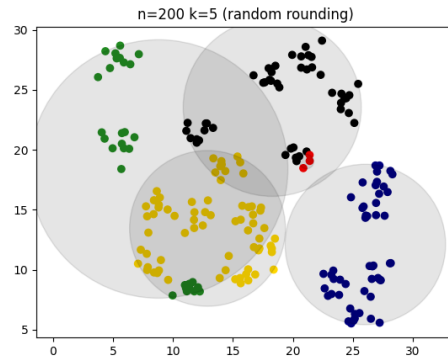(b) Randomized (best of 50) $w(C) = 3658976$
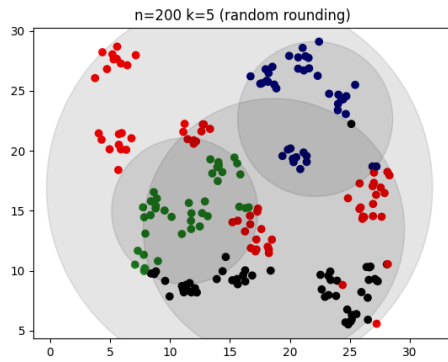

(c) Fixed point iteration $w(C) = 3722073$
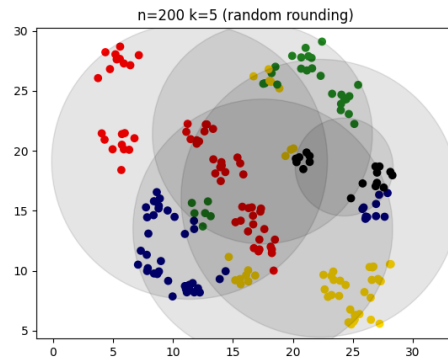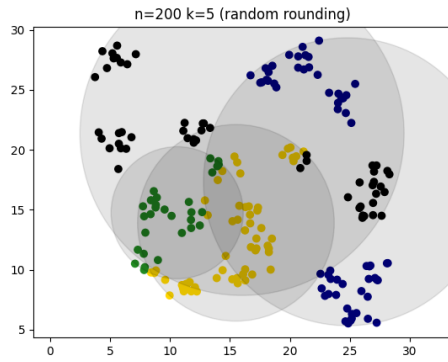
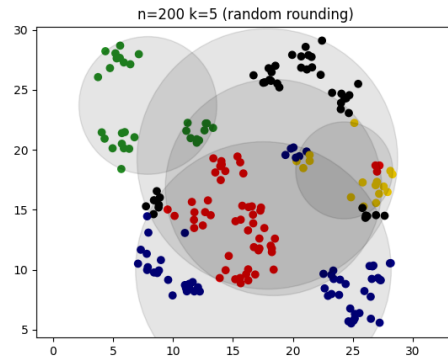Figure 7: Clustering 200 points with $k = 20$.

(a) Trial 1

(b) Trial 2
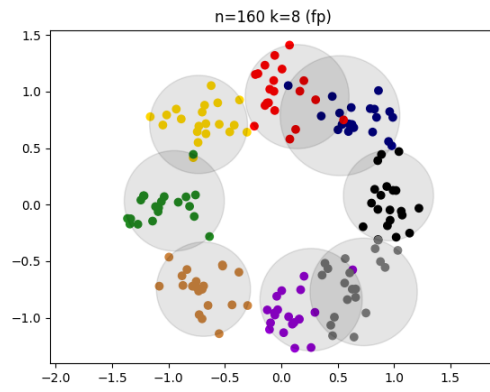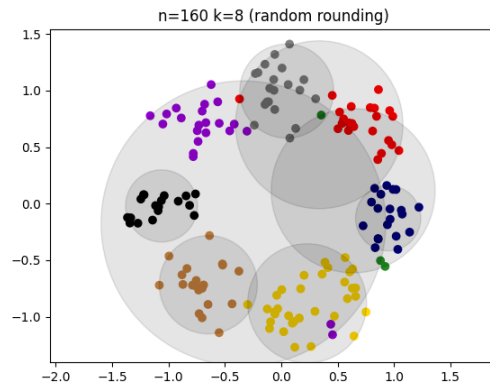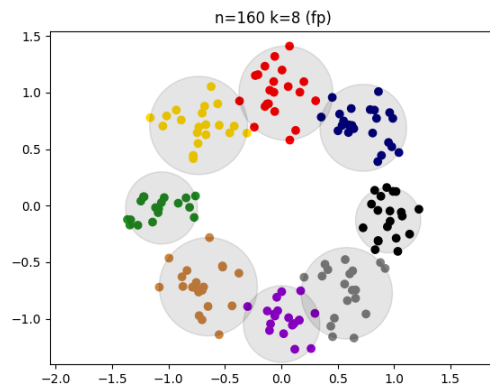
(c) Trial 3

(d) Trial 4

(e) Trial 5

(f) Trial 6

Figure 8: Clustering 200 points with $k = 5$ and randomized rounding. The result of the fixed point iteration method in the same data is shown in Figure 5(c).

(a) Input data and ground truth clustering

(b) Randomized (best of 50) $w(C) = 26837$

(c) Fixed point iteration $w(C) = 27158$

Figure 9: Clustering 160 points sampled from 8 Gaussians with $k = 8$.

	(a) Partition weights		(b) Clustering accuracy	
	$w(C)/w(D)$			Rand index
minimum	1.005		fixed point	0.972 ± 0.006
maximum	1.026		randomized rounding	0.935 ± 0.018
mean	1.014			

Table 1: Experiments with 10 different datasets generated by a mixture of Gaussians (see Figure 9). (a) Minimum, maximum and mean value of $w(C)/w(D)$, where C is the result of the fixed point iteration method and D is the result of the best of 50 trials of randomized rounding. (b) Mean and standard deviation of the Rand index for both methods.
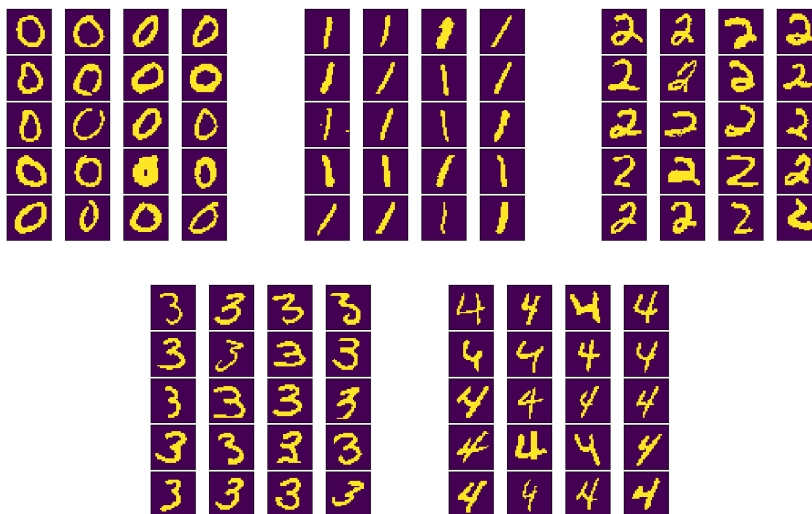

Figure 10: Subset of MNIST handwritten digits used in one of the clustering trials.

6.2 MNIST digits

In this section we evaluate the performance of our clustering method on a subset of the MNIST handwritten digits dataset (LeCun et al.).

We performed multiple clustering trials with different subsets of the MNIST training data. In each trial we selected 20 random examples for each of 5 digits (0, 1, 2, 3 and 4). Each example was represented by a 784 dimensional 0/1 vector encoding a 28x28 binary image. Figure 10 shows the data from one of the trials.

In Table 2 we report the accuracy of the clustering results obtained using both our fixed point iteration method and the randomized method for rounding the Max k-Cut relaxation. For comparison we also evaluate the results of clustering the data using the k-means algorithm. We see that our fixed point iteration method for rounding the Max

Method	Rand index
Max k-Cut SDP with fixed point iteration	0.907 ± 0.026
Max k-Cut SDP with randomized rounding	0.874 ± 0.037
k-means	0.864 ± 0.034

Table 2: Clustering MNIST hardwritten digits. We report both the mean and standard deviation of the Rand index over 20 trials with random subsets of the data.

k-Cut relaxation gives the best accuracy when compared both to the use of randomized rounding and the k-means algorithm.[2]

References

David Arthur and Sergei Vassilvitskii. k-means++: The advantages of careful seeding. In *ACM-SIAM Symposium on Discrete Algorithms*, pages 1027–1035, 2007.

Pranjal Awasthi, Afonso S. Bandeira, Moses Charikar, Ravishankar Krishnaswamy, Soledad Villar, and Rachel Ward. Relax, no need to round: Integrality of clustering formulations. In *Conference on Innovations in Theoretical Computer Science*, 2015.

Boaz Barak, Jonathan A Kelner, and David Steurer. Rounding sum-of-squares relaxations. In *ACM Symposium on Theory of Computing*, pages 31–40, 2014.

Dimitris Bertsimas, Agni Orfanoudaki, and Holly Wiberg. Interpretable clustering: an optimization approach. *Machine Learning*, 110(1):89–138, 2021.

Ankur Bhargava and S Rao Kosaraju. Derandomization of dimensionality reduction and SDP based algorithms. In *Workshop on Algorithms and Data Structures*, pages 396–408, 2005.

Emmanuel J Candès and Terence Tao. The power of convex relaxation: Near-optimal matrix completion. *IEEE Transactions on Information Theory*, 56(5):2053–2080, 2010.

Diego Cifuentes, Corey Harris, and Bernd Sturmfels. The geometry of SDP-exactness in quadratic optimization. *Mathematical Programming*, 182:399–428, 2020.

Lars Engebretsen, Piotr Indyk, and Ryan O'Donnell. Derandomized dimensionality reduction with applications. In *ACM-SIAM Symposium on Discrete Algorithms*, page 705–712, 2002.

Pedro Felzenszwalb, Caroline Klivans, and Alice Paul. Iterated linear optimization. *Quarterly of Applied Mathematics*, 79:601–615, 2021.

Alan Frieze and Mark Jerrum. Improved approximation algorithms for max k-cut and max bisection. *Algorithmica*, 18(1):67–81, 1997.

---

2. We used the scipy library implementation of k-means. Each trial involved 10 different random initializations of the initial cluster centers using the k-means++ method.

Michel Goemans and David Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the ACM*, 42 (6):1115–1145, 1995.

Anil K Jain and Richard C Dubes. *Algorithms for clustering data*. Prentice-Hall, 1988.

Ravi Kannan, Santosh Vempala, and Adrian Vetta. On clusterings: Good, bad and spectral. *Journal of the ACM*, 51(3):497–515, 2004.

Monique Laurent and Svatopluk Poljak. On a positive semidefinite relaxation of the cut polytope. *Linear Algebra and its Applications*, 223/224:439–461, 1995.

Yann LeCun, Corinna Cortes, and Christopher Burges. MNIST database of handwritten digits. http://yann.lecun.com/exdb/mnist/.

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

David MacKay. *Information theory, inference and learning algorithms*. Cambridge University press, 2003.

Sanjeev Mahajan and Hariharan Ramesh. Derandomizing approximation algorithms based on semidefinite programming. *SIAM Journal on Computing*, 28(5):1641–1663, 1999.

Marina Meila and Jianbo Shi. A random walks view of spectral segmentation. In *AISTATS*, 2001.

Dustin G Mixon, Soledad Villar, and Rachel Ward. Clustering subgaussian mixtures by semidefinite programming. *Information and Inference: A Journal of the IMA*, 6(4):389–415, 2017.

Andrew Y Ng, Michael I Jordan, and Yair Weiss. On spectral clustering: Analysis and an algorithm. In *Advances in Neural Information Processing Systems*, pages 849–856, 2002.

Jiming Peng and Yu Wei. Approximating k-means-type clustering via semidefinite programming. *SIAM journal on optimization*, 18(1):186–205, 2007.

William M Rand. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical association*, 66(336):846–850, 1971.

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

Cor J. Veenman, Marcel J. T. Reinders, and Eric Backer. A maximum variance cluster algorithm. *IEEE Transactions on pattern analysis and machine intelligence*, 24(9):1273–1280, 2002.

Meihong Wang and Fei Sha. Information theoretical clustering via semidefinite programming. In *AISTATS*, 2011.

Yair Weiss. Segmentation using eigenvectors: a unifying view. In *IEEE International Conference on Computer Vision*, 1999.