Theoretical foundations for efficient clustering

by

Shrinu Kushagra

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Doctor of Philosophy
in
Computer Science

Waterloo, Ontario, Canada, 2019

Examining Committee Membership

The following served on the Examining Committee for this thesis. The decision of the Examining Committee is by majority vote.

External Examiner:	Sanjoy Dasgupta Professor, Computer Science and Engineering University of California, San Diego
Supervisor(s):	Shai Ben-David Professor, Dept. of Computer Science, University of Waterloo
Internal Member:	Yaoliang Yu Assistant Professor, Dept. of Computer Science University of Waterloo
Internal-External Member:	Chaitanya Swamy Professor, Dept. of Combinatorics & Optimization University of Waterloo
Other Member(s):	Eric Blais Assistant Professor, Dept. of Computer Science University of Waterloo

Author's Declaration

This thesis consists of material all of which I authored or co-authored: see Statement of Contributions included in the thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Statement of Contributions

This thesis consists of material all of which I authored or co-authored. Chapter 2 is based on a joint work with Hassan Ashtiani and Shai Ben-David [Ashtiani et al., 2016]. The first part of Chapter 3 (the framework of promise correlation clustering) is based on a joint work with Ihab Ilyas and Shai Ben-David [Kushagra et al., 2019a]. The experimental and hashing based algorithm in this chapter is based on the joint work with Hemant Saxena, Ihab Ilyas and Shai Ben-David [Kushagra et al., 2019b]. Chapter 4 is based on a joint work with Samira Samadi and Shai Ben-David [Kushagra et al., 2016]. Chapter 5 is based on a joint work with Yaoliang Yu and Shai Ben-David [Kushagra et al., 2017].

I understand that my thesis may be made electronically available to the public.

Abstract

Clustering aims to group together data instances which are similar while simultaneously separating the dissimilar instances. The task of clustering is challenging due to many factors. The most well-studied is the high computational cost. The clustering task can be viewed as an optimization problem where the goal is to minimize a certain cost function (like k-means cost or k-median cost). Not only are the minimization problems NP-hard but often also NP-hard to approximate (within a constant factor). There are two other major issues in clustering, namely *under-specificity* and *noise-robustness*. The focus of this thesis is tackling these two issues while simultaneously ensuring low computational cost.

Clustering is an under-specified task. The same dataset may need to be clustered in different ways depending upon the intended application. Different solution requirements need different approaches. In such situations, domain knowledge is needed to better define the clustering problem. We incorporate this by allowing the clustering algorithm to interact with an oracle by asking whether two points belong to the same or different cluster. In a preliminary work, we show that access to a small number of same-cluster queries makes an otherwise NP-hard k-means clustering problem computationally tractable. Next, we consider the problem of clustering for data de-duplication; detecting records which correspond to the same physical entity in a database. We propose a correlation clustering like framework to model such record de-duplication problems. We show that access to a small number of same-cluster queries can help us solve the 'restricted' version of correlation clustering. Rather surprisingly, more relaxed versions of correlation clustering [1] are intractable even when allowed to make a 'large' number of same-cluster queries.

Next, we explore the issue of noise-robustness of clustering algorithms. Many real-world datasets, have on top of cohesive subsets, a significant amount of points which are 'unstructured'. The addition of these *noisy* points makes it difficult to detect the structure of the remaining points. In the first line of work, we define noise as not having significantly large dense subsets. We provide computationally efficient clustering algorithms that capture all meaningful clusterings of the dataset; where the clusters are cohesive (defined formally by notions of clusterability) and where the noise satisfies the gray background assumption. We complement our results by showing that when either the notions of structure or the noise requirements are relaxed, no such results are possible. In the second line of work, we develop a generic procedure that can transform objective-based clustering algorithms into one that is robust to outliers (as long the number of such points is not 'too large'). In particular, we develop efficient noise-robust versions of two common clustering algorithms and prove robustness guarantees for them.

[1] We refer to it as promise correlation clustering.

Acknowledgements

I would like to take this opportunity and express my deepest thanks to my advisor Prof. Shai Ben-David. I have been working with Shai for the last six years, the first two as his masters and the remaining as his phd advisee. His willingness to let the student explore new problems and uncharted territories (while hand-holding at appropriate but not all the time) helped me develop the ability to think independently and clearly about any given problem. His ability to formulate interesting problems and present his ideas confidently are two qualities I greatly admire and helped me immensely during my phd. If at the end of these six years, I acquired atleast 10% of both these qualities, I would consider myself lucky. The last six years were truly an intellectually challenging and simulating journey thanks to Shai.

Next, I would like to take the opportunity to thank my thesis committee members starting with Prof. Yaoliang Yu. Besides Shai, he is probably the faculty member who I had the most interactions with. Prof. Yao was always friendly and approachable whenever I needed some advice on research and otherwise. Next, I would like to express my gratitude for the other members of my committee. Firstly, for serving on the committee and their insightful comments. Special mention for Prof. Sanjoy Dasgupta who agreed to serve as an external examiner. Next, I would also like to thank Hassan Ashtiani who is also a co-author on one chapter of this dissertation.

I was lucky to be a part of the university Varsity Squash team. I would like to thank my squash coach, Mr. Clive Porter for giving me the opportunity to be part of an amazing squash community here in waterloo and ontario. Special thanks to my squash captain and good friends Cameron Seth and Tori Grootjen. Travelling with the team for various tournaments, the thrill of competition, the agony of loss and the joy of winning. The squash team has given me memories that I am going to carry with me for the rest of my life. Regularly playing squash was one of the reasons that I was able to maintain my mental composure during my phd.

This journey would have been incomplete without the love and support of my friends who have supported me through the ups and downs of life, both personal and professional. First and foremost, Shivam Dembla who at this point is more a brother than a friend. He is one person who has always had my back during the many challenging times I faced during the past few years. Next, I would like to thank my very good friend Rupinder Ghotra for his constant support and encouragement over the years. Finally, I would like to thank Samira Samadi, a friend who is also a co-author on part of this dissertation. Her constant support and encouragement (during a period where I was not able to make a lot

of progress towards my research) really helped to regain my focus. Had it not been for her during those times, I might have given up.

I would like to take thank my parents for constant support and encouragement over the years. For giving me the freedom to pursue whatever career option I wanted and being a pillar of strength. My parents' work ethic and hard-work has been a source of inspiration for me which I try, rather unsuccessfully, to incorporate in my life.

Finally, I would like to thank the two most important people in my life, my grand-parents. Whatever semblance of success I have had has been due to their unconditional love and good wishes and sacrifices. I have been fortunate to have two sets of parents, a blessing which only very few people can claim to have. The thesis is a culmination of their blessings and prayers for me.

Dedication

In the loving memory of Shreya.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Clustering is a term used to describe a wide variety of unsupervised learning algorithms. One popular definition of clustering is that it attempts to partition a given dataset into subsets (or *clusters*) such that similar points share the same cluster and dissimilar points are separated into different clusters. Clustering is a very challenging task and in this thesis we examine three such challenges.

The first challenge that we address is computational complexity. This is one of the most well-studied challenges of clustering. Clustering is commonly posed as an optimization problem where the goal is to minimize a certain cost function. For many common cost functions such as k-means or k-median or k-center it is known that the optimization problem is NP-hard [Dasgupta, 2008], [Megiddo and Supowit, 1984].

The second challenge that we examine is the issue of *under-specificity*. The basic definition of clustering requires an algorithm to partition the dataset into coherent subsets which are 'well-separated'. On a closer look, we see that this definition is problematic. Consider a set of (say n) points on a straight line where each pair of adjacent points have a small distance (say α) between them. If we impose the requirement that each pair of similar points should share a cluster then all the points would end up in a single cluster. This would violate the second requirement as dissimilar points would also share the same cluster. Similarly, if we require that all the dissimilar instances be separated then we would end up separating similar pairs of points as well. This example is shown in Fig. 1.1

Hence, different algorithms 'focus more' on different aspects of the clustering definition. For example, single-linkage tries to put similar points in the same cluster. However, it may end up having dissimilar points in the same cluster as well. On the other hand, the Lloyd's algorithm tries to separate dis-similar points but may end up separating similar points as

Figure 1.1: An example of n points on the real line separated by a small distance α. This figure highlights that it is not always possible to satisfy the two requirements from clustering algorithms.

well. The basic definition of clustering does not have enough information to resolve this conflict. Thus, we say that the clustering problem is under-specified. The fundamental question that we ask here is that 'how do we prefer one clustering algorithm over another?'

We address the problem of under-specificity by incorporating domain knowledge into the clustering problem. The clustering algorithm is allowed to interact with an oracle (or an expert) by asking whether two points should belong to the same or different clusters. The oracle replies either 'yes' or 'no' to the *same-cluster* query depending upon whether the two points belong to the same or different clusters. In this case, the goal of the clustering algorithm is to recover the clustering which the oracle has in its 'mind'.

In the first part of the thesis we simultaneously address the challenges of under-specificity and computational cost. We study the *computational* and *query* complexity of various clustering problems in this framework. Consider the following simple observation. Given any clustering instance, if the algorithm is allowed to make n^2 (where n is the size of the dataset) queries to the same-cluster oracle then recovering the true or target clustering is trivial. In this dissertation, one of the important questions that we examine is the following. Is it possible to efficiently solve (in polyniomial time) an otherwise intractable clustering problem while making a 'small' number of same-cluster queries to the oracle?

Now, we discuss the third challenge; namely the issue of *noise-robustness* of clustering algorithms. The basic definition of clustering says that the goal is to partition a dataset into clusters such that similar points share a cluster while dissimlar points are separated into different clusters. This definition makes sense when the given dataset has a cohesive structure. That is, the dataset can be partitioned into groups or clusters which have some inter-cluster separation. However, real-world datasets, on top of this structure, have a significant subset of points which are unstructured. The addition of these *noisy* points makes it difficult for the clustering algorithm to detect the structure of remaining points. The precise definition of 'unstructuredness' or noisy points varies depending upon the structure that the clustering algorithm is trying to detect. In this dissertation, we consider two definitions of noise and develop efficient noise-robust clustering algorithms in each case.

We address all the challenges of clustering under a formal framework. The goal is to have a framework which can be mathematically analyzed and is also relevant to practitioners. Obviously, the exact framework varies depending upon the clustering problem we are considering. To analyze these issues in a formal framework, we will be mostly relying on mathematical theorems and proofs. But in some cases (where applicable), we have also complemented our results with experiments and simulations.

1.1 Our Contributions

As we alluded to before, our contributions can be divided into two categories. One is dealing with under-specificity and the second related to noise robustness of clustering algorithms. In this next sub-sections, we go into more details of each of these and outline our objectives and contributions made.

1.1.1 Under-specificity

The first main contribution is to develop a formal framework to incorporate domain knowledge into the clustering problem. We introduce a semi-supervised clustering framework. The learner is allowed to interact with a domain expert (or an oracle) by asking whether two data instances belong to the same or different cluster (same-cluster query). The oracle has a target clustering in its mind and responds by answering either 'yes' or 'no' (depending upon whether the two points belong to the same or different clusters according to the target clustering). We assume that the oracle is perfect and has complete knowledge of the ground truth clustering. Hence, given any pair of points it always gives the correct response. We consider two clustering problems under this framework.

Clustering with advice (k-means)

We consider a setting where the oracle conforms to a center-based clustering with a notion of margin. That is, the target clustering has the following property. Every cluster-center is 'more' closer (γ-times closer) to points in its own cluster than to points in a different cluster. Larger values of γ imply greater separation between different clusters.

Under this framework, we study the query and computational complexity of recovering the target clustering. We provide an algorithm which runs in $O\big(kn \log n\big)$ time and makes $O\big(k \log n + k^2 \log k\big)$ same-cluster queries to the oracle and succeeds in recovering the

oracle's clustering with high probability. Here n is the size of the dataset and k is the number of clusters.

We also consider the case when the oracle conforms to the optimal k-means clustering under γ-margin. Then, our query-based algorithm can find the optimal solution in polynomial time. Interestingly, we prove that even when the optimal k-means clustering satisfies margin conditions, without queries, finding that solution is NP-hard. Thus having access to relatively few oracle queries can allow efficient solutions to otherwise intractable problems.

Correlation clustering with advice

We consider the problem of correlation clustering under the semi-supervised clustering framework. Correlation clustering is very useful for modelling the record de-duplication problem; the task of detecting multiple records that correspond to the same real-world entity in a database. Here the goal is to put records corresponding to the same physical entity in the same cluster and putting records corresponding to different physical entities into different clusters.

Formally, given a complete graph G with the edges labelled 0 and 1, the goal of correlation clustering is to find a clustering that minimizes the number of 0 edges within a cluster plus the number of 1 edges across different clusters. In other words, the goal is to find a clustering which minimizes the *correlation loss* w.r.t the graph G.

Promise correlation clustering

The optimal clustering C^* can also be viewed as a complete graph G^* (unknown to the clustering algorithm) with edges corresponding to points in the same cluster being labelled 1 and other edges being labelled 0. If it is known that the edge difference between G and G^* is zero, then finding C^* is easy (find the connected components of G). We consider a variant of this problem where it is promised that the edge difference between G and the unknown G^* is "small". The goal is to find the clustering which minimizes the correlation loss w.r.t G.

We now wish to analyze the computational and query complexity of the promise correlation clustering (PCC) problem. We prove that the promise version is still NP-hard. Rather surprisingly, we further prove that even with access to a same-cluster oracle, the promise version is intractable as long as the number queries to the oracle is $o(n)$ (the proof assumes the Exponential Time Hypothesis; n is the number of vertices).

Restricted correlation clustering
Given these negative results, we consider a restricted version of correlation clustering. First observe that in the standard version, the goal is to find a clustering over the class of all possible clusterings of the dataset. Here, we restrict ourselves to a given class $\mathcal{F}$ of clusterings. Another difference is that we want to minimize the correlation loss w.r.t the unknown target clustering C^* rather than a graph G.

We now wish to analyze the query and computational complexity of this problem. We offer an algorithmic approach (using same-cluster queries) and prove that the query complexity is upper bounded by a term that depends only on the VC-Dim($\mathcal{F}$) and is independent of the dataset size. We also provide computationally efficient algorithms for a few common classes of clusterings.

1.1.2 Noise-robustness

We now describe our framework to address the issue of noise in clustering algorithms. We are given a dataset X which is made up of two parts. The first is the structured or clusterable component S. Mathematically, this is captured by introducing notions of 'clusterability of data'. Intuitively, these notions say that the set S is composed of k different clusters and the clusters are 'well-separated' from each other. In this thesis, we consider two such notions, center-proximity and center-separation. Each of them formalize the idea of well-separatedness in a different way. The second component of the dataset is the noisy or unstructured part N. The clustering algorithm receives X as its input. It does not have any knowledge about S (or its substructure) or N. The goal is to partition X into components so that the structure of S is *preserved*. Any algorithm which achieves this goal is said to be noise-robust.

Detecting cluster structure in the presence of sparse noise

We consider the problem of detecting the structure of S when the noisy part N is sparse. That is, the only restriction about the noisy part of the data is that it does not create significantly large clusters. We introduce efficient algorithms that discover and cluster every subset S of the data with the following property. S has a meaningful structure (as captured by a notion of clusterability) and its complement is structureless or sparse. We say that our algorithm is robust to sparse noise. Notably, the success of our algorithms do not depend on any upper bound on the fraction of noisy data. We complement our results by showing that when either the notions of structure or the noise requirements are relaxed, no such results are possible.

Detecting cluster structure in the presence of outliers

We propose a generic regularization-based method that transforms any center-based clustering objective into a noise-robust one. We use our procedure to obtain regularized versions of two common clustering algorithms based on the k-means objective function. We prove that these regularized algorithms are robust to outliers (under clusterability assumptions and mildness properties of the noisy points).

1.2 Reading the thesis

This dissertation is composed of two components. The first addresses the problem of underspecificity in clustering and is covered in Chapters 2 and 3. The second part is about the issue of noise-robustness of clustering algorithms and is covered in Chapters 4 and 5. The two parts are independent of one another and the reader can start with whichever one he/she is more interested in.

We have ensured that each chapter is self-contained. Note that the thesis does not have a dedicated chapter on related work or on notation/preliminaries. Rather these are included in the relevant chapters. Some of the missing proofs can be found in appendices at the end of the corresponding chapters.

1.3 Technical background

Some of the proofs in the thesis use the following classical results from learning theory literature. Readers who are unfamiliar with result can also refer to the standard texts on this subject, for example, [Shalev-Shwartz and Ben-David, 2014].

1.3.1 Learning theory results

Theorem 1.1 (Vapnik and Chervonenkis [Vapnik and Chervonenkis, 2015])**.** *Let X be a domain set and D a probability distribution over X. Let H be a class of subsets of X of finite VC-dimension d. Let $\epsilon, \delta \in (0,1)$. Let $S \subseteq X$ be picked i.i.d according to D of size m. If $m > \frac{c}{\epsilon^2}(d \log \frac{d}{\epsilon} + \log \frac{1}{\delta})$, then with probability $1-\delta$ over the choice of S, we have that $\forall h \in H$*

$$\left| \frac{|h \cap S|}{|S|} - P(h) \right| < \epsilon$$

Theorem 1.2 (Fundamental theorem of learning [Blumer et al., 1989])**.** *Here, we state the theorem as in the book [Shalev-Shwartz and Ben-David, 2014]. Let H be a class of functions $h : \mathcal{X} \to \{0,1\}$ of finite VC-Dimension, that is* VC-Dim$(H) = d < \infty$*. Let D be a probability distribution over X and h^* be some unknown target function. Given $\epsilon, \delta \in (0,1)$. Let err_D be the $\{0,1\}$-loss function $err : H \to [0,1]$. That is $err_D(h) = \mathbf{P}_{x \in D}[h(x) \neq h^*(x)]$. Sample a set $S = \{(x_1, y_1), \ldots, (x_m, y_m)\}$ according to the distribution D. Define $err_S(h) = \sum_{i=1}^{m} \frac{\mathbf{1}_{[h(x_i) \neq h^*(x_i)]}}{m}$. If $m \geq a\frac{d+\log(1/\delta)}{\epsilon^2}$, then with probability at least $1-\delta$ over the choice of S, we have that for all $h \in H$*

$$|err_D(h) - err_S(h)| \leq \epsilon$$

where a is an absolute global constant.

1.3.2 Concentration inequalities

Some of the proofs use the following concentration inequalities.

Theorem 1.3 (Sum of geometric random variables [Brown, 2011])**.** *Let $X = X_1 + \ldots + X_n$ be n geometrically distributed random variables such that $\mathbf{E}[X_i] = \mu$. Then*

$$\mathbf{P}[X > (1+\nu)n\mu] \leq \exp\left(\frac{-\nu^2\mu n}{2(1+\nu)}\right)$$

Theorem 1.4 (Generalized Hoeffding's Inequality (e.g., [Ashtiani and Ghodsi, 2015]))**.** *Let $X_1, \ldots . X_n$ be i.i.d random vectors in some Hilbert space such that for all i, $\|X_i\|_2 \leq R$ and $E[X_i] = \mu$. If $n > c\frac{\log(1/\delta)}{\epsilon^2}$, then with probability at least $1-\delta$, we have that*

$$\left\|\mu - \frac{1}{n}\sum X_i\right\|_2^2 \leq R^2\epsilon$$

Theorem 1.5 (Multiplicative Chernoff bound [Mitzenmacher and Upfal, 2005])**.** *Let $X_1, \ldots, X_n$ be i.i.d random variables in $\{0,1\}$ such that $\mu = E[X_i]$. Let $X = \frac{\sum X_i}{n}$. Then for any $0 < \epsilon < 1$*

$$P\left[\ X > (1+\epsilon)\mu\ \right] \ \leq\ \exp\left(\frac{-\epsilon^2\mu n}{3}\right)$$

Theorem 1.6 (Multiplicative Chernoff bound [Mitzenmacher and Upfal, 2005])**.** *Let $X_1, \ldots, X_n$ be i.i.d random variables in $\{0,1\}$ such that $\mu = E[X_i]$. Let $X = \frac{\sum X_i}{n}$. Then for any $0 < \epsilon < 1$*

$$P\left[\ X < (1-\epsilon)\mu\ \right] \ \leq\ \exp\left(\frac{-\epsilon^2\mu n}{2}\right)$$

Chapter 2

Clustering with same-cluster queries

Clustering is an under-specified task. Given a dataset, there is no unique or 'correct' solution. The solution of choice varies with the intended application. In the absence of knowledge of the desired application, clustering problem is usually not well-defined. We need to incorporate domain expertise to better define the clustering problem. At the same time, performing clustering under many natural models of computation is computationally intractable.

In this chapter, we take a new approach to alleviate these two issues in clustering. We first allow the clustering algorithm to be semi-supervised (through access to answers of natural queries). This makes the clustering problem better defined. We then ask the following question. *Can semi-supervision help relax the computational burden of clustering?*

Aside from trial and error, one principled approach to supervision in clustering is through *link/do-not-link* constraints [Basu et al., 2002, Basu et al., 2004, Kulis et al., 2009]. In this case, besides the usual input, the clustering algorithm also receives a set of pairs which should always share the same cluster and a set of pairs which should always be separated into different clusters. Our approach combines the user-friendliness of link/do-not-link constraints with the advantages of interactiveness. We call it *same-cluster* queries. That is, the algorithm can interact with a domain expert (or an oracle) by asking whether two elements belong to the same cluster or not. The oracle responds by answering 'yes' or 'no' depending upon whether the two points share a cluster or are separated into different clusters.

The general setting in this chapter is the following. Let X be a set of elements that should be clustered and d a distance metric over it. The oracle (e.g., a domain expert) has information about a target clustering C^* in its mind. That target clustering satisfies some

predefined clusterability condition (which we call γ-margin). The notion says that each cluster center is γ times closer to the points in its own cluster than to points in different clusters. The clustering algorithm has access to (X, d) and can also ask same-cluster queries about C^*. The goal of the algorithm is to efficiently recover the target clustering C^* while making a 'small' number of same-cluster queries to the oracle.

We provide a polynomial time algorithm for clustering with queries, that succeeds (with high probability) under the assumption that the input satisfies the γ-margin condition for $\gamma > 1$. This algorithm makes $O\big(k^2 \log k + k \log n\big)$ same-cluster queries to the oracle and runs in $O\big(kn \log n\big)$ time, where k is the number of clusters and $n = |X|$.

We also consider the following special case. The target C^* (in the oracle's mind) is the optimal solution of the k-means clustering problem and also satisfies the clusterability condition. In this case, the query-based algorithm succeeds with high probability as long as $\gamma > 1$. On the other hand, we show that without access to a same-cluster oracle, k-means clustering is NP-hard even when the optimal solution satisfies γ-margin property for $\gamma = \sqrt{3.4} \approx 1.84$ and $k = \Theta(n^\epsilon)$ (for any $\epsilon \in (0, 1)$). This shows that access to a small number of same-cluster queries can provide an efficient algorithm for an otherwise NP-hard problem. This is an interesting trade-off between query complexity and computational complexity in the clustering domain.

2.1 Related Work

This work combines two directions of research. The first is clustering under semi-supervision and the second is computational complexity of clustering.

The most common framework to convey supervision in clustering is through a set of link/do-not-link constraints [Basu et al., 2002, Basu et al., 2004, Kulis et al., 2009]. Here, the clustering algorithm is provided a list of pairs which should share the same cluster and a list of pairs which should be separated into different clusters. Here, the supervision is non-interactive. On the interactive side, Balcan et al. [Balcan and Blum, 2008] introduced the notion of *split/merge* queries. In their framework, the domain expert (or oracle) receives a clustering of the given dataset. Given a clustering, the oracle responds by suggesting either to merge two clusters or to split a cluster. For each query, the oracle should evaluate a clustering of the entire dataset. Another example of supervision is clustering was introduced by Ashtiani and Ben-David [Ashtiani and Ben-David, 2015]. Here, the oracle receives a small subset of the dataset and responds by outputting the 'correct' clustering of that subset. Our approach combines the user-friendliness of link/do-not-link

constraints (as opposed to requiring the oracle to answer queries about the whole data set or to cluster subsets of the data) with the advantages of interactiveness.

The computational complexity of clustering is very well studied problem. Clustering is usually posed as an optimization problem where the goal is to minimize a given objective (or cost) function. k-means and k-median are two popular objective functions. However, many of the results are negative. For example, minimizing the k-means objective is NP-hard. Moreover, it is NP-hard even for $k = 2$ [Dasgupta, 2008], or even in a two-dimensional plane [Vattani, 2009, Mahajan et al., 2009]. Similarly, minimizing the k-median objective is also NP-hard. Despite these negative results, clustering algorithms are routinely and successfully applied in practice. To explain this gap between theory and practice, notions of data niceness under which clustering becomes easy have been considered. The hypothesis is that real-world datasets are not worst-case and often have nice properties which makes clustering tractable. [Ben-David, 2015] has an extensive survey of such notions.

The notion of α-center proximity introduced by Awasthi et al. [Awasthi et al., 2012] is most related to our notion of margin (a comparison between the two can be found in Appendix A.2). In the restricted scenario (i.e., when the centers of clusters are selected from the data set), their algorithm efficiently recovers the target clustering for $\alpha > 3$. Balcan and Liang [Balcan and Liang, 2012] provide a different algorithm and prove that recovery is possible for $\alpha > \sqrt{2} + 1$. Ben-David and Reyzin [Ben-David and Reyzin, 2014] show that this problem is NP-hard for $\alpha < 2$. Variants of these proofs for our γ-margin condition yield the feasibility of k-median clustering (centers of clusters are selected from the input) when the input satisfies the condition with $\gamma > 2$ and NP hardness when $\gamma < 2$.

2.2 Problem Formulation

Let X be a subset of some Euclidean space, $\mathbf{R}^d$. A k-clustering of the set X partitions it into k disjoint subsets $\mathcal{C} = \{C_1, \ldots, C_k\}$. We say $x_1 \overset{C}{\sim} x_2$ if x_1 and x_2 belong to the same cluster according to C. Also, $n := |X|$. We say that a clustering C is *center-based* if there exists centers $\mu = \{\mu_1, \ldots, \mu_k\} \subset \mathbf{R}^d$ such that C corresponds to the Voronoi diagram over these centers. Namely, for every x in X and $i \leq k$, $x \in C_i \Leftrightarrow i = \arg\min_j d(x, \mu_j)$.

Next, we introduce our data niceness property or our notion of clusterability of a data set.

Definition 2.1 (γ-margin)**.** *Let X be set of points in a metric space (M, d). Let $C = \{C_1, \ldots, C_k\}$ be a center-based clustering of X induced by centers $\mu_1, \ldots, \mu_k \in M$. We*

say that C satisfies the γ-margin property if the following holds. For all $i \in [k]$ and every $x \in C_i$ and $y \in X \setminus C_i$,

$$\gamma d(x, \mu_i) < d(y, \mu_i)$$

We now introduce our notion of a same-cluster oracle and same-cluster queries. For a clustering $C^* = \{C_1^*, \dots C_k^*\}$, a C^*-oracle is a function $\mathcal{O} : X \to \{true, false\}$ that answers queries according to C^*. We can think of the oracle as a human or domain-expert that has knowledge about the desired clustering and can answer the clustering algorithm's queries. The clustering algorithm then tries to recover C^* by querying $\mathcal{O}$.

Definition 2.2 (Same-cluster Query)**.** *Given a clustering instance X and a C^*-oracle $\mathcal{O}$. A same-cluster query asks whether two instances x_1 and x_2 belong to the same cluster, i.e.,*

$$\mathcal{O}(x_1, x_2) = \begin{cases} true & \text{if } x_1 \overset{C^*}{\sim} x_2 \\ false & o.w. \end{cases}$$

2.3 Clustering using same-cluster queries

In this section we provide an efficient algorithm for clustering with queries. The setting is the one described in the previous section. In particular, it is assumed that the oracle has a center-based clustering in his mind which satisfies the γ-margin property. The space is Euclidean and the center of each cluster is the center of mass of the instances in that cluster. The algorithm not only makes same-cluster queries, but also another type of query defined as below.

Definition 2.3 (Cluster-assignment Query)**.** *A cluster-assignment query asks the cluster index that an instance x belongs to. In other words $\mathcal{O}_{C^*}(x) = i$ if and only if $x \in C_i^*$.*

Note however that each cluster-assignment query can be replaced with k same-cluster queries (see appendix A.1). Therefore, we can express everything in terms of the more natural notion of same-cluster queries, and the use of cluster-assignment query is just to make the representation of the algorithm simpler.

Our algorithm works in two phases. In the first phase, it approximates the center of one of the clusters. It does this by asking cluster-assignment queries about a set of points sampled uniformly at randomly, until it has a sufficient number of points from at least

one cluster (say C_p). It uses the mean of these points μ'_p as an approximation of the true cluster center μ_p.

In the second phase, the algorithm recovers all of the instances belonging to C_p. In order to do that, it first sorts all of the instances based on their distance to μ'_p. The margin assumption ensures that all of the points in C_p lie inside a sphere centered at μ'_p (which does not include points from any other cluster). It then tries to find the radius of this sphere by doing binary search using same-cluster queries. After that, the elements in C_p will be located and can be removed from the data set. The algorithm repeats this process k times to recover all of the clusters. The details of our approach is in Algorithm 1. Note that β is a small constant[1].

Algorithm 1: Algorithm for $\gamma(>1)$-margin instances with queries

Input: Clustering instance X, a C^*-oracle $\mathcal{O}$, the number of clusters k and a parameter $\delta \in (0,1)$

Output: A clustering C of the set X

$C = \{\}$, $S = X$ and $\eta = \beta \frac{\log k + \log(1/\delta)}{(\gamma-1)^4}$

while $S \neq \phi$ **do**

Phase 1

Let $l = k\eta + 1$

Draw l points from S uniformly at random

Let $S_t = \{x \in S : \mathcal{O}(x) = t\}$. where the oracle answers cluster-assignment queries about the members of S

$p = \arg\max_t |S_t|$ and $\mu'_p := \frac{1}{|S_p|} \sum_{x \in S_p} x$.

Phase 2

Sorts elements of $\{x : x \in S\}$ in increasing order of $d(x, \mu'_p)$.

Binary search over S, to find i such that $b_i \in C_p$ and $b_{i+1} \notin C_p$. (This step involves making same-cluster queries to the oracle $\mathcal{O}$).

$C'_p = \{x \in S : d(x, \mu'_p) \leq d(b_i, \mu'_p)\}$.

Delete C'_p from S and $C = C \cup \{C'_p\}$

end

Theorem 2.6 shows that if $\gamma > 1$ then our algorithm recovers the target clustering with high probability. Next, we give bounds on the time and query complexity of our algorithm.

[1]It corresponds to the constant appeared in generalized Hoeffding inequality bound, discussed in Theorem 1.4 in appendix 1.3.2

Theorem 2.7 shows that our approach needs $O(k \log n + k^2 \log k)$ queries and runs with time complexity $O(kn \log n)$.

Lemma 2.4. *Given a clustering instance $X \subseteq \mathbf{R}^d$. Let $C = \{C_1, \ldots, C_k\}$ be any center-based clustering of X that satisfies the γ-margin property. Let μ_i be the set of centers corresponding to the centers of mass of C_i. Let μ_i' be such that $d(\mu_i, \mu_i') \leq r(C_i)\epsilon$, where $r(C_i) = \max_{x \in C_i} d(x, \mu_i)$. If $\gamma \geq 1 + 2\epsilon$ then*

$$\forall x \in C_i, \forall y \in X \setminus C_i \Rightarrow d(x, \mu_i') < d(y, \mu_i')$$

Proof. Fix any $x \in C_i$ and $y \in C_j$. $d(x, \mu_i') \leq d(x, \mu_i) + d(\mu_i, \mu_i') \leq r(C_i)(1 + \epsilon)$. Similarly, $d(y, \mu_i') \geq d(y, \mu_i) - d(\mu_i, \mu_i') > (\gamma - \epsilon) r(C_i)$. Combining the two, we get that $d(x, \mu_i') < \frac{1+\epsilon}{\gamma-\epsilon} d(y, \mu_i')$. □

Lemma 2.5. *Let the framework be as in Lemma 2.4. Let S_p, C_p, μ_p, μ_p' and η be defined as in Algorithm 1, and $\epsilon = \frac{\gamma-1}{2}$. If $|S_p| > \eta$, then the probability that $d(\mu_p, \mu_p') > r(C_p)\epsilon$ is at most $\frac{\delta}{k}$.*

Proof. Define a uniform distribution U over C_p. Then μ_p and μ_p' are the true and empirical mean of this distribution. Using a standard concentration inequality (Thm. 1.4 from Appendix 1.3.2) shows that the empirical mean is close to the true mean, completing the proof. □

Theorem 2.6. *Given a clustering instance $X \subseteq \mathbf{R}^d$. Let $C^* = \{C_1^*, \ldots, C_k^*\}$ be center-based clustering of X that satisfies the γ-margin property. Given $\delta \in (0, 1)$ and $\gamma > 1$ and a C^*-oracle, with probability at least $1 - \delta$, Algorithm 1 outputs C^*.*

Proof. In the first phase of the algorithm we are making $l > k\eta$ cluster-assignment queries. Therefore, using the pigeonhole principle, we know that there exists cluster index p such that $|S_p| > \eta$. Then Lemma 2.5 implies that the algorithm chooses a center μ_p' such that with probability at least $1 - \frac{\delta}{k}$ we have $d(\mu_p, \mu_p') \leq r(C_p)\epsilon$. By Lemma 2.4, this would mean that $d(x, \mu_p') < d(y, \mu_p')$ for all $x \in C_p$ and $y \notin C_p$. Hence, the radius r_i found in the phase two of Alg. 1 is such that $r_i = \max_{x \in C_p} d(x, \mu_p')$. This implies that C_p' (found in phase two) equals to C_p. Hence, with probability at least $1 - \frac{\delta}{k}$ one iteration of the algorithm successfully finds all the points in a cluster C_p. Using union bound, we get that with probability at least $1 - k\frac{\delta}{k} = 1 - \delta$, the algorithm recovers the target clustering. □

Theorem 2.7. *Let the framework be as in theorem 2.6. Then Algorithm 1*

- *Makes* $O\big(k \log n + k^2 \frac{\log k + \log(1/\delta)}{(\gamma-1)^4}\big)$ *same-cluster queries to the oracle* $\mathcal{O}$.
- *Runs in* $O\big(kn \log n + k^2 \frac{\log k + \log(1/\delta)}{(\gamma-1)^4}\big)$ *time.*

Proof. In each iteration (i) the first phase of the algorithm takes $O(\eta)$ time and makes $\eta+1$ cluster-assignment queries (ii) the second phase takes $O(n \log n)$ times and makes $O(\log n)$ same-cluster queries. Each cluster-assignment query can be replaced with k same-cluster queries; therefore, each iteration runs in $O(k\eta+n \log n)$ and uses $O(k\eta+\log n)$ same-cluster queries. By replacing $\eta = \beta \frac{\log k + \log(1/\delta)}{(\gamma-1)^4}$ and noting that there are k iterations completes the proof. □

2.4 Hardness of k-means with Margin

Finding k-means solution without the help of an oracle is generally computationally hard. In this section, we will show that solving Euclidean k-means remains hard even if we know that the optimal solution satisfies the γ-margin property for $\gamma = \sqrt{3.4}$. In particular, we show the hardness for the case of $k = \Theta(n^\epsilon)$ for any $\epsilon \in (0, 1)$.

In Section 2.3, we proposed a polynomial-time algorithm that could recover the target clustering using $O(k^2 \log k + k \log n)$ queries, assuming that the clustering satisfies the γ-margin property for $\gamma > 1$. Now assume that the oracle conforms to the optimal k-means clustering solution. In this case, for $1 < \gamma \leq \sqrt{3.4} \approx 1.84$, solving k-means clustering would be NP-hard without queries, while it becomes efficiently solvable with the help of an oracle [2].

Given a set of instances $\mathcal{X} \subset \mathbf{R}^d$, the k-means clustering problem is to find a clustering $\mathcal{C} = \{C_1, \ldots, C_k\}$ which minimizes $f(\mathcal{C}) = \sum_{C_i} \min_{\mu_i \in \mathbf{R}^d} \sum_{x \in C_i} \|x - \mu_i\|_2^2$. The decision version of k-means is, given some value L, is there a clustering $\mathcal{C}$ with cost $\leq L$? The following theorem is the main result of this section.

Theorem 2.8. *Finding the optimal solution to Euclidean k-means objective function is NP-hard when $k = \Theta(n^\epsilon)$ for any $\epsilon \in (0, 1)$, even when the optimal solution satisfies the γ-margin property for $\gamma = \sqrt{3.4}$.*

[2] To be precise, note that the algorithm used for clustering with queries is probabilistic, while the lower bound that we provide is for deterministic algorithms. However, this implies a lower bound for randomized algorithms as well unless $BPP \neq P$

This result extends the hardness result of [Ben-David and Reyzin, 2014] to the case of Euclidean metric, rather than arbitrary one, and to the γ-margin condition (instead of the α-center proximity there). In the subsequent sections, we provide the proof of the main result.

2.4.1 Overview of the proof

Our method to prove Thm. 2.8 is based on the approach employed by [Vattani, 2009]. However, the original construction proposed in [Vattani, 2009] does not satisfy the γ-margin property. Therefore, we have to modify the proof by setting up the parameters of the construction more carefully.

To prove the theorem, we will provide a reduction from the problem of Exact Cover by 3-Sets (X3C) which is NP-Complete [Garey and Johnson, 2002], to the decision version of k-means.

Definition 2.9 (X3C)**.** *Given a set U containing exactly $3m$ elements and a collection $\mathcal{S} = \{S_1, \ldots, S_l\}$ of subsets of U such that each S_i contains exactly three elements, does there exist m elements in $\mathcal{S}$ such that their union is U?*

We will show how to translate each instance of X3C, $(U, \mathcal{S})$, to an instance of k-means clustering in the Euclidean plane, X. In particular, X has a grid-like structure consisting of l rows (one for each S_i) and roughly $6m$ columns (corresponding to U) which are embedded in the Euclidean plane. The special geometry of the embedding makes sure that any low-cost k-means clustering of the points (where k is roughly $6ml$) exhibits a certain structure. In particular, any low-cost k-means clustering could cluster each row in only two ways; One of these corresponds to S_i being included in the cover, while the other means it should be excluded. We will then show that U has a cover of size m if and only if X has a clustering of cost less than a specific value L. Furthermore, our choice of embedding makes sure that the optimal clustering satisfies the γ-margin property for $\gamma = \sqrt{3.4} \approx 1.84$.

2.4.2 Reduction design

Given an instance of X3C, that is the elements $U = \{1, \ldots, 3m\}$ and the collection $\mathcal{S}$, we construct a set of points X in the Euclidean plane which we want to cluster. Particularly, X consists of a set of points $H_{l,m}$ in a grid-like manner, and the sets Z_i corresponding to S_i. In other words, $X = H_{l,m} \cup (\cup_{i=1}^{l-1} Z_i)$.

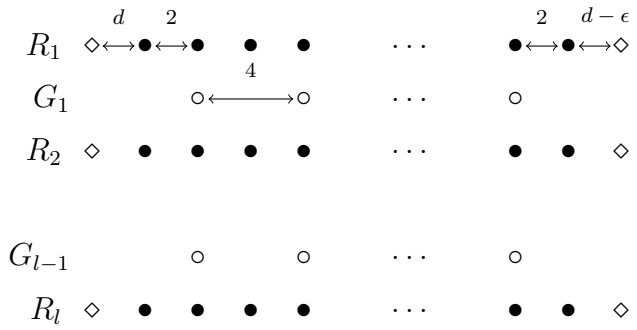


Figure 2.1: Geometry of $H_{l,m}$. This figure is similar to Fig. 1 in [Vattani, 2009]. Reading from left to right, each row R_i consists of a diamond (s_i), $6m+1$ bullets $(r_{i,1}, \ldots, r_{i,6m+1})$, and another diamond (f_i). Each rows G_i consists of $3m$ circles $(g_{i,1}, \ldots, g_{i,3m})$.

The set $H_{l,m}$ is as described in Fig. 2.1. The row R_i is composed of $6m+3$ points $\{s_i, r_{i,1}, \ldots, r_{i,6m+1}, f_i\}$. Row G_i is composed of $3m$ points $\{g_{i,1}, \ldots, g_{i,3m}\}$. The distances between the points are also shown in Fig. 2.1. Also, all these points have weight w, simply meaning that each point is actually a set of w points on the same location.

Each set Z_i is constructed based on S_i. In particular, $Z_i = \cup_{j\in[3m]} B_{i,j}$, where $B_{i,j}$ is a subset of $\{x_{i,j}, x'_{i,j}, y_{i,j}, y'_{i,j}\}$ and is constructed as follows: $x_{i,j} \in B_{i,j}$ iff $j \notin S_i$, and $x'_{i,j} \in B_{i,j}$ iff $j \in S_i$. Similarly, $y_{i,j} \in B_{i,j}$ iff $j \notin S_{i+1}$, and $y'_{i,j} \in B_{i,j}$ iff $j \in S_{i+1}$. Furthermore, $x_{i,j}, x'_{i,j}, y_{i,j}$ and $y'_{i,j}$ are specific locations as depicted in Fig. 2.2. In other words, exactly one of the locations $x_{i,j}$ and $x'_{i,j}$, and one of $y_{i,j}$ and $y'_{i,j}$ will be occupied. We set the following parameters.

$$h = \sqrt{5}, d = \sqrt{6}, \epsilon = \frac{1}{w^2}, \lambda = \frac{2}{\sqrt{3}}h, k = (l-1)3m + l(3m+2)$$

$$L_1 = (6m+3)wl, L_2 = 4m(l-1)wh^2, L = L_1 + L_2 - m\alpha, \alpha = \frac{d}{w} - \frac{1}{2w^3}$$

In order to start, we first need to establish some properties about the Euclidean embedding of X.

Definition 2.10 (A- and B-Clustering of R_i)**.** *A-Clustering of row R_i is a clustering in the form of $\{\{s_i\}, \{r_{i,1}, r_{i,2}\}, \ldots, \{r_{i,6m-1}, r_{i,6m}\}, \{r_{i,6m+1}, f_i\}\}$. B-Clustering of row R_i is a clustering in the form of $\{\{s_i, r_{i,1}\}, \{r_{i,2}, r_{i,3}\}, \ldots, \{r_{i,6m}, r_{i,6m+1}\}, \{f_i\}\}$.*

Definition 2.11 (Good point for a cluster)**.** *A cluster C is good for a point $z \notin C$ if adding z to C increases cost by exactly $\frac{2w}{3}h^2$*

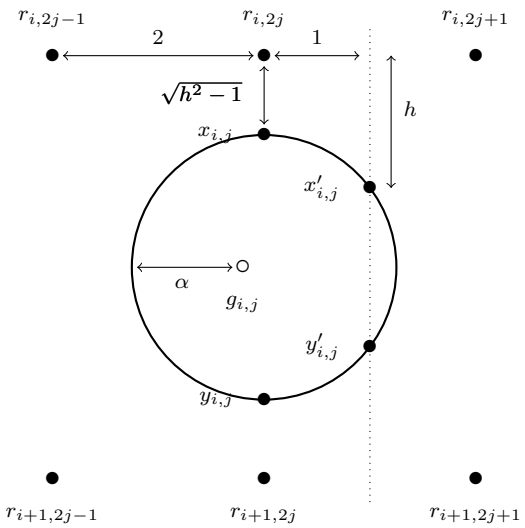


Figure 2.2: The locations of $x_{i,j}$, $x'_{i,j}$, $y_{i,j}$ and $y'_{i,j}$ in the set Z_i. Note that the point $g_{i,j}$ is not vertically aligned with $x_{i,j}$ or $r_{i,2j}$. This figure is adapted from [Vattani, 2009].

Given the above definition, the following simple observations can be made.

- The clusters $\{r_{i,2j-1}, r_{i,2j}\}$, $\{r_{i,2j}, r_{i,2j+1}\}$ and $\{g_{i,j}\}$ are good for $x_{i,j}$ and $y_{i-1,j}$.
- The clusters $\{r_{i,2j}, r_{i,2j+1}\}$ and $\{g_{i,j}\}$ are good for $x'_{i,j}$ and $y'_{i-1,j}$.

Definition 2.12 (Nice Clustering)**.** *A k-clusteirng is nice if every $g_{i,j}$ is a singleton cluster, each R_i is grouped in the form of either an A-clustering or a B-clustering, and each point in Z_i is added to a cluster which is good for it.*

It is straightforward to see that a row grouped in a A-clustering costs $(6m+3)w - \alpha$ while a row in B-clustering costs $(6m+3)w$. Hence, a nice clustering of $H_{l,m} \cup Z$ costs at most $L_1 + L_2$. More specifically, if t rows are group in a A-clustering, the nice-clustering costs $L_1 + L_2 - t\alpha$. Also, observe that any nice clustering of X has only the following four different types of clusters.

(1) Type E - $\{r_{i,2j-1}, r_{i,2j+1}\}$
The cost of this cluster is $2w$ and the contribution of each location to the cost (i.e., $\frac{cost}{\#locations}$) is $\frac{2w}{2} = w$.

(2) Type F - $\{r_{i,2j-1}, r_{i,2j}, x_{i,j}\}$ or $\{r_{i,2j-1}, r_{i,2j}, y_{i-1,j}\}$ or $\{r_{i,2j}, r_{i,2j+1}, x'_{i,j}\}$ or $\{r_{i,2j}, r_{i,2j+1}, y'_{i-1,j}\}$. The cost of any cluster of this type is $2w(1 + \frac{h^2}{3})$ and the contribution of each location to the cost is at most $\frac{2w}{9}(h^2+3)$. This is equal to $\frac{16}{9}w$ because we had set $h = \sqrt{5}$.

(3) Type I - $\{g_{i,j}, x_{i,j}\}$ or $\{g_{i,j}, x'_{i,j}\}$ or $\{g_{i,j}, y_{i,j}\}$ or $\{g_{i,j}, y'_{i,j}\}$
The cost of any cluster of this type is $\frac{2}{3}wh^2$ and the contribution to the cost of each location is $\frac{w}{3}h^2$. For our choice of h, the contribution is $\frac{5}{3}w$.
(4) Type J - $\{s_i, r_{i,1}\}$ or $\{r_{i,6m+1}, f_i\}$
The cost of this cluster is $3w$ (or $3w - \alpha$) and the contribution of each location to the cost is at most $1.5w$.

Hence, observe that in a nice-clustering, any location contributes at most $\leq \frac{16}{9}w$ to the total clustering cost. This observation will be useful in the proof of the lemma below.

Lemma 2.13. *For large enough $w = poly(l, m)$, any non-nice clustering of $X = H_{l,m} \cup Z$ costs at least $L + \frac{w}{3}$.*

Proof. We will show that any non-nice clustering C of X costs at least $\frac{w}{3}$ more than any nice clustering. This will prove our result. The following cases are possible.

- C contains a cluster C_i of cardinality $t > 6$ (i.e., contains t weighted points)
Observe that any $x \in C_i$ has at least $t - 5$ locations at a distance greater than 4 to it, and 4 locations at a distance at least 2 to it. Hence, the cost of C_i is at least $\frac{w}{2t}(4^2(t-5) + 2^2 4)t = 8w(t-4)$. C_i allows us to use at most $t - 2$ singletons. This is because a nice clustering of these $t + (t-2)$ points uses at most $t - 1$ clusters and the clustering C uses $1 + (t-2)$ clusters for these points. The cost of the nice cluster on these points is $\leq \frac{16w}{9}2(t-1)$. While the non-nice clustering costs at least $8w(t-4)$. For $t \geq 6.4 \implies 8(t-4) > \frac{32}{9}(t-1)$ and the claim follows. Note that in this case the difference in cost is at least $\frac{8w}{3}$.
- Contains a cluster of cardinality $t = 6$
Simple arguments show that amongst all clusters of cardinality 6, the following has the minimum cost. $C_i = \{r_{i,2j-1}, r_{i,2j}, x_{i,j}, y_{i-1,j}, r_{i,2j+1}, r_{2j+2}\}$. The cost of this cluster is $\frac{176w}{6}$. Arguing as before, this allows us to use 4 singletons. Hence, a nice cluster on these 10 points costs at most $\frac{160w}{9}$. The difference of cost is at least $34w$.
- Contains a cluster of cardinality $t = 5$
Simple arguments show that amongst all clusters of cardinality 5, the following has the minimum cost. $C_i = \{r_{i,2j-1}, r_{i,2j}, x_{i,j}, y_{i-1,j}, r_{i,2j+1}\}$. The cost of this cluster is $16w$. Arguing as before, this allows us to use 3 singletons. Hence, a nice cluster on these 8 points costs at most $16w\frac{8}{9}$. The difference of cost is at least $\frac{16w}{9}$.
- Contains a cluster of cardinality $t = 4$
It is easy to see that amongst all clusters of cardinality 4, the following has the minimum cost. $C_i = \{r_{i,2j-1}, r_{i,2j}, x_{i,j}, r_{i,2j+1}\}$. The cost of this cluster is $11w$. Arguing as before, this allows us to use 2 singletons. Hence, a nice cluster on these 6 points costs at most $\frac{32w}{3}$. The difference of cost is at least $\frac{w}{3}$.

- All the clusters have cardinality ≤ 3
  Observe that amongst all non-nice clusters of cardinality 3, the following has the minimum cost. $C_i = \{r_{i,2j-1}, r_{i,2j}, r_{i,2j+1}\}$. The cost of this cluster is $8w$. Arguing as before, this allows us to use at most 1 more singleton. Hence, a nice cluster on these 4 points costs at most $\frac{64w}{9}$. The difference of cost is at least $\frac{8w}{9}$.
  It is also simple to see that any non-nice clustering of size 2 causes an increase in cost of at least w.

□

Lemma 2.14. *The set $X = H_{l,n} \cup Z$ has a k-clustering of cost less or equal to L if and only if there is an exact cover for the X3C instance.*

Proof. The proof is identical to the proof of Lemma 11 in [Vattani, 2009]. Note that the parameters that we use are different with those utilized by [Vattani, 2009]; however, this is not an issue, because we can invoke our lemma 2.13 instead of the analogous result in Vattani (i.e., lemma 10 in Vattani's paper). The sketch of the proof is that based on lemma 2.13, only nice clusterings of X cost $\leq L$. On the other hand, a nice clustering corresponds to an exact 3-set cover. Therefore, if there exists a clustering of X of cost $\leq L$, then there is an exact 3-set cover. The other way is simpler to proof; assume that there exists an exact 3-set cover. Then, the corresponding construction of X makes sure that it will be clustered *nicely*, and therefore will cost $\leq L$. □

Lemma 2.15. *Any k-clustering of $X = H_{l,n} \cup Z$ with cost $\leq L$ has the γ-margin property where $\gamma = \sqrt{3.4}$. Furthermore, $k = \Theta(n^\epsilon)$.*

Proof. As argued before, any nice clustering has four different types of clusters. We will calculate the minimum ratio $a_i = \frac{d(y,\mu)}{d(x,\mu)}$ for each of these clusters C_i (where $x \in C_i$, $y \notin C_i$ and μ is mean of all the points in C_i.) Then, the minimum a_i will give the desired γ.

(1) For Type E clusters $a_i = h/1 = \sqrt{5}$.

(2) For Type F clusters. $a_i = \frac{\frac{\sqrt{4+16(h^2-1)}}{3}}{2h/3} = \sqrt{\frac{17}{5}} \approx 1.84$.

(3) For Type I clusters, standard calculation show that $a_i > 2$.

(4) For Type J clusters $a_i = \frac{2+\frac{\sqrt{6}}{2}}{\frac{\sqrt{6}}{2}} > 2$.

Furthurmore, $|\mathcal{X}| = (12lm + 3l - 6m)w$ and $k = 6lm + 2l - 3m$. Hence for $w =$poly(l, m) our hardness result holds for $k = |\mathcal{X}|^\epsilon$ for any $0 < \epsilon < 1$. □

Lemmas 2.14 and 2.15 together show that X has a k-clustering of cost $\leq L$ satisfying the γ-margin property (for $\gamma = \sqrt{3.4}$) if and only if there is an exact cover by 3-sets for the X3C instance. This completes the proof of our main result (Thm. 2.8).

2.5 Conclusions and Future Directions

In this work we introduced a framework for semi-supervised clustering with same-cluster queries. Those queries can be viewed as a natural way for a clustering mechanism to gain domain knowledge, without which clustering is an under-defined task. The focus of our analysis was the computational and query complexity of such problems, when the input data set satisfies a clusterability condition – the γ-margin property.

Our main result shows that access to a limited number of such query answers (logarithmic in the size of the data set and quadratic in the number of clusters) allows efficient successful clustering under conditions (margin parameter between 1 and $\sqrt{3.4} \approx 1.84$) that render the problem NP-hard without the help of such a query mechanism.

Subsequently, the framework of same-cluster queries have been applied to other clustering problems as well. [Ailon et al., 2017] considered the problem of k-means clustering without the margin assumption. They provide an efficient algorithm which finds an approximate solution using same-cluster queries. We can observe that even for this problem same-cluster queries can help us tackle an otherwise computationally intractable problem. [Mazumdar and Saha, 2017] considered the case of noisy oracles, where the experts' answers are correct with probability p. In the next chapter, we study a different clustering problem where same-cluster queries can be useful.

Chapter 3

Semi-supervised clustering for deduplication

Modern businesses generate a tremendous amount of data. These are stored in large databases which are then used to make many critical decisions. Therefore, ensuring the quality of these datasets becomes extremely important. As the data accumulates from multiple sources over time, many errors creep into the data. For example, many records end up having duplicate entries. Record de-duplication is a central task in managing large scale databases. The goal is to detect records in a database that correspond to the same real word entity.

The problem of data de-duplication can be viewed as a clustering task. Here, the goal is to put records corresponding to the same physical entity in the same cluster while separating the records corresponding to different entities into different clusters. Clustering for de-duplication has many characteristics which are different from standard clustering problems. Many popular clustering algorithms like k-means or k-median receive as input the value k, that is the number of clusters to output. This information is unknown in de-duplication applications. In any dataset, the number of different-cluster pairs (i.e. different entity pairs) is order of magnitude greater than the number of positive or same-cluster pairs (i.e. same entity pairs). Hence, common machine learning tools of classification prediction (learning a binary classifier over the set of pairs of instances) do not automatically transfer to this domain as the dataset is heavily skewed towards the negative pairs.

The framework of correlation clustering is very natural for modelling the problem of data de-duplication [Bansal et al., 2004]. Here, de-duplication is viewed as an optimization problem over graphs. More formally, given a dataset X and a complete graph G over the

set. The edges of the graph are labelled 0 or 1. An edge label of zero indicates that the corresponding vertices have been deemed to be in different cluster while an edge label of one indicates that the corresponding vertices should be in the same cluster. The motivation for edge labelling is the following. Often the practitioner can design a pairwise similarity function over the pairs of points. The pairs whose similarity is above a certain threshold are deemed as positive (or same-cluster) and the remaining pairs are deemed to be negative (or different-cluster). Sometimes, the similarity metric is also learned from training data.

Given the graph, the goal of correlation clustering is to find a clustering of the dataset which correlates 'as much as possible' with the given edges. In other words, find a clustering which minimizes the correlation loss w.r.t the given edges. Correlation loss is defined as the sum of the number of zero edges within a cluster plus the number of one edges across different clusters. However, solving this optimization problem is NP-hard [Bansal et al., 2004].

In this chapter, we offer a formal modelling of such record de-duplication tasks. Our framework is the same as correlation clustering but with the added *promise* that the input graph edges E is 'close to' the optimal correlation clustering of the given dataset. We analyse the computational complexity of this problem and show that even under strong promise, correlation clustering is NP-hard. Moreover, the problem remains NP-hard (assuming the ETH hypothesis) even when we are allowed to make queries to a human expert (or an *oracle*) as long as the number of queries is not too large (sub-linear in the number of points in the dataset).

Given these negative results, we introduce the framework of restricted correlation clustering (RCC). This framework has two important differences from the standard (and promise) correlation clustering formulation. Firstly, the optimization problem is restricted over a given class $\mathcal{F}$ of clusterings. Secondly, the goal is to find a clustering which correlates as much as possible with the unknown target (or ground truth) clustering C^*. That is, C^* is the clustering where only records corresponding to same entity are in the same cluster. We offer an algorithmic approach (which uses the help of an oracle) with success guarantees for the restricted version. The 'success guarantee' depends on the complexity of the class $\mathcal{F}$ (measured by VC-Dim($\mathcal{F}$)) as well as the 'closeness' of the distance (or similarity) metric d to the target clustering. We complement our theoretical results by carrying out extensive experimental evaluation of our framework on a diverse class of clustering algorithms and across multiple real world datasets.

3.1 Related Work

The most relevant work is the framework of correlation clustering that we discussed in the previous section [Bansal et al., 2004]. Other variations of correlation clustering have been considered. For example [Demaine et al., 2006], consider a problem where the edges can be labelled by a real number instead of just 0 or 1. Edges with large positive weights encourage those vertices to be in the same cluster while edges with large negative weights encourage those points to be in different clusters. They showed that the problem is NP-hard and gave a $O(\log n)$ approximation to the weighted correlation clustering problem. [Charikar et al., 2005] made several contributions to the correlation clustering problem. For the problem of minimizing the correlation clustering loss (for unweighted complete graphs), they gave an algorithm with factor 4 approximation. They also proved that the minimization problem is APX-Hard.

More recently, [Ailon et al., 2018] considered the problem of correlation clustering in the presence of an oracle. If the number of clusters k is known, they proposed an algorithm which makes $O(k^{14} \log n)$ queries to the oracle and finds a $(1+\epsilon)$-approximation to the correlation clustering problem. They showed that the problem is NP-hard to approximate with $o\big(\frac{k}{poly \log k}\big)$ queries to an oracle. In this work, we obtain similar results for the promise correlation clustering problem.

Supervision in clustering has been addressed before. For example, [Kulis et al., 2009, Basu et al., 2004, Basu et al., 2002] considered *link/don't-link* constraints. This is a form of non-interactive clustering where the algorithm gets as input a list of pairs which should be in the same cluster and a list pairs which should be in different clusters. The framework of interactive clustering was developed by [Balcan and Blum, 2008] where the supervision is provided in the form of *split/merge* queries. The algorithm gives the current clustering to the oracle. The oracle responds by telling the which clusters to merge and which clusters to split.

[Ashtiani et al., 2016] developed the framework of same-cluster queries which we use in this chapter. At any given instant, the clustering algorithm asks the same-cluster oracle about two points in the dataset. The oracle replies by answering either 'yes' or 'no' depending upon whether the two points lie in the same or different clusters.

On de-duplication side, most prevalent are approaches that are based on designing a similarity measure (or distance) over the records, such that records that are highly similar according to that measure are likely to be duplicates and records that measure as significantly dissimilar are likely to represent different entities. For example, to handle duplicate records created due typographical mistakes, many character-based similarity

metrics have been considered. Examples of such metrics include the edit or levenshtein distance [Levenshtein, 1966], smith-waterman distance [Smith and Waterman, 1981] and jaro distance metric [Jaro, 1980]. Token-based similarity metrics try to handle rearrangement of words, for example [Monge et al., 1996] and [Cohen, 1998]. Other techniques include phonetic-based metrics and numerical metrics (to handle numeric data). A nice overview of these methods can be found in [Elmagarmid et al., 2007].

While the above approaches relied on designing a good similarity metric, some works try to 'learn' the distance function from a labelled training dataset of pairs of records. Examples of such works include [Cochinwala et al., 2001] and [Bilenko et al., 2003]. Clustering for de-duplication has been mostly addressed in application oriented works. One work assumes that the duplicate records are transitive [Hernández and Stolfo, 1995]. The clustering problem now reduces to finding the connected components in a graph. An extensive experimental evaluation of different graph-based clustering algorithms on a simulated dataset of strings was carried out by [Hassanzadeh et al., 2009].

3.2 Preliminaries

Given X, a clustering C of the set X partitions it into k disjoint subsets or clusters. The clustering C can also be viewed as a $\{0,1\}$-function over the domain $X^{[2]} := \{(x_1, x_2) : x_1 \neq x_2\}$. Here, $C(x_1, x_2) = 1$ iff x_1, x_2 belong to the same cluster according to C. Similarly, we also view the edges of a graph $G = (X, E)$ as a $\{0,1\}$-function over $X^{[2]}$.

We allow a clustering algorithm to make queries to a human oracle in the following way.

Definition 3.1 (Same-cluster oracle [Ashtiani et al., 2016]). *Given a set X and an unknown target clustering C^*. A same-cluster C^*-oracle receives a pair $(x_1, x_2) \in X^{[2]}$ as input and outputs 1 if and only if x_1, x_2 belong to the same cluster according to C^*.*

From the perspective of de-duplication, a same-cluster oracle receives two records x_1 and x_2. The oracle returns 1 if x_1 and x_2 correspond to the same real-world entity. Otherwise, the oracle responds 0.

Definition 3.2 (Correlation loss[Bansal et al., 2004]). *Given graph $G = (X, E)$ where X is the set of vertices (the given dataset to be clustered) and E is the set of edges. The*

correlation loss of a clustering C w.r.t the edges E is defined as

$$\begin{aligned} corrL_E(C) &= corrN_E(C) + corrP_E(C), \text{ where} \\ corrN_E(C) &= |\{(x,y) : C(x,y) = 1 \text{ and } E(x,y) = 0\}|, \\ corrP_E(C) &= |\{(x,y) : C(x,y) = 0 \text{ and } E(x,y) = 1\}| \end{aligned} \tag{3.1}$$

A weighted version of the loss function places weights of w_1 and w_2 on the two terms and is defined as

$$corrL^w_E(C) = w_1\, corrN_E(C) + w_2\, corrP_E(C) \tag{3.2}$$

The goal of correlation clustering is to find a clustering which minimizes the (weighted) correlation loss. We also consider the normalized version of this loss function w.r.t a target clustering C^* (rather than the edges E).

Definition 3.3 (Normalized correlation loss)**.** *Given domain X, a target clustering C^* and a parameter μ. The loss of a clustering C w.r.t the target C^* is defined as*

$$\begin{aligned} L_{C^*}(C) = \;& \mu\; L_{P^+}(C) \;+\; (1-\mu)\; L_{P^-}(C), \text{ where} \\ L_{P^+}(C) = \;& \mathop{\mathbf{P}}_{(x,y)\sim P^+}\big[C(x,y) = 0\big], \\ L_{P^-}(C) = \;& \mathop{\mathbf{P}}_{(x,y)\sim P^-}\big[C(x,y) = 1\big] \end{aligned} \tag{3.3}$$

where P^+ is the uniform distribution over $X^{[2]}_+ = \{(x,y) : C^*(x,y) = 1\}$ and P^- is the uniform distribution over $X^{[2]}_- = \{(x,y) : C^*(x,y) = 0\}$.

The normalized correlation loss measures two quantities for the clustering C. The first is the fraction of the true positive pairs that C gets wrong (or loss over the positive pairs). The second is the fraction of true negative pairs that C gets wrong (or the loss over the negative pairs). It then obtains a weighted sum of the two losses.

Lets observe the relation between Defns. 3.2 and 3.3. Define $\gamma_0 := \mathbf{P}[C^*(x,y) = 1]$, that is the probability of true positive (or same-cluster pairs) in the dataset. Using the notation of Defn. 3.2, we see that $corrP_{C^*}(C) = \gamma_0 |X^{[2]}| L_{P^+}(C)$ and $corrN_{C^*}(C) = (1-\gamma_0)|X^{[2]}| L_{P^-}(C)$. Normalising by $|X^{[2]}|$ and choosing $\mu = \frac{w_2\gamma_0}{w_1(1-\gamma_0)+w_2\gamma_0}$ gives us the normalized version of the loss function.

Definition 3.4. *Given a metric space (X, d), a target clustering C^* and a parameter λ. We say that the metric d is (α, β)-informative w.r.t C^* and λ if*

$$\mathop{\mathbf{P}}_{(x,y)\sim U^2} \left[d(x,y) > \lambda \mid C^*(x,y) = 1 \right] \leq \alpha \tag{3.4}$$

$$\mathop{\mathbf{P}}_{(x,y)\sim U^2} \left[C^*(x,y) = 1 \mid d(x,y) \leq \lambda \right] \geq \beta \tag{3.5}$$

Here U^2 is the uniform distribution over $X^{[2]}$.

In deduplication applications, often the distance function is such that pairs with distance within a certain threshold are likely to be in the same cluster. The definition of an informative metric formalizes this intuition. It says that most of the true positive pairs have a distance of at most λ between them. Also, amongst all pairs with distance $\leq \lambda$, at least a β fraction of them belong to the same cluster.

Definition 3.5 (γ-skewed)**.** *Given X and a target clustering C^*. We say that X is γ-skewed w.r.t C^* if*

$$\mathop{\mathbf{P}}_{(x,y)\sim U^2} \left[C^*(x,y) = 1 \right] \leq \gamma$$

In de-duplication applications, most of the pairs are negative (or belong to different clusters). The above definition states this property formally. In the next section, we introduce our framework of *promise correlation clustering* and discuss the computational complexity of the problem both in the absence and presence of an oracle.

Definition 3.6 (VC-dimension)**.** *Given a domain X and a hypothesis class H. We say that a set $C \subset X$ is shattered by H if $|H_C| = 2^{|C|}$. That is, H restricted to C is the set of all possible functions from C to $\{0, 1\}$. The vc dimension of H denoted by* VC-Dim(H) *is the maximum size of $C \subset X$ which can be shattered by H.*

$$\text{VC-Dim}(H) = \max_{C \subset X} \ H \text{ shatters } C$$

3.3 Promise Correlation Clustering

Definition 3.7 (Promise correlation clustering (PCC))**.** *Given a clustering instance $G = (X, E)$. Let C^* be such that*

$$C^* = \arg\min_{C \in \mathcal{F}} \ corrL_E(C) \tag{3.6}$$

where $\mathcal{F}$ is the set of all possible clusterings C such that $M(C) \leq p$ (where that $M(C)$ denotes the maximum size of a cluster in the clustering C). Given that E is (α, β)-informative w.r.t C^*. Find the clustering C^*.

When the edges E correspond to a clustering C then $\beta = 1$ and $\alpha = 0$. We show in the subsequent sections that even for this 'restricted' class of clusterings (when the size of the maximum cluster is at most a constant M) and given the prior knowledge, PCC is still NP-hard. Furthermore, PCC is NP-hard even when we are allowed to make $o(|X|)$ queries to a C^*-oracle.

3.3.1 PCC is NP-hard

Theorem 3.8. *Finding the optimal solution to the Promise Correlation Clustering problem is NP-hard for all $p \geq 3$ and for $\alpha = 0$ and $\beta = \frac{1}{2}$.*

To prove the result, we will use a reduction from exact cover by 3-sets problem which is known to be NP-hard.

(X3C) Given a universe of elements $U = \{x_1, \ldots, x_{3q}\}$ and a collections of subsets $S = \{S_1, \ldots, S_m\}$. Each $S_i \subset U$ and contains exactly three elements. Does there exist $S' \subseteq S$ such that each element of U occurs exactly once in S'?

This decision problem is known to be NP-hard [Garey and Johnson, 2002]. We will now reduce an instance of X3C to the promise correlation clustering problem using local replacement described in Fig. 3.1. The details of the construction are as follows. For every element $x \in U$, we add a corresponding vertex $x \in V$. For every three set $S_i = \{x_{i1}, x_{i2}, x_{i3}\}$, we construct B_{ij} for $i \leq t$ and $j \leq p$. Here, t is a constant which will be specified later. Each B_{ij} is a clique of size p. The 'connection' between the different blocks are specified in Fig. 3.1. Let A be an algorithm which solves the promise problem described in Eqn. 3.6. Then, we can use this algorithm to decide exact cover by three sets as follows.

If A outputs a clustering C such that all the clusters have size exactly p and E_C makes no negative errors w.r.t E (that is $\alpha(E_C) = 0$) then output YES. Otherwise, output NO. Next, we will prove that this procedure decides X3C.

Let there exists an exact cover for the X3C instance. Let C be the clustering corresponding to the exact cover. That is, the edges colored blue and black correspond to this clustering and the corresponding vertices are in the same cluster (Fig. 3.1). Note that this

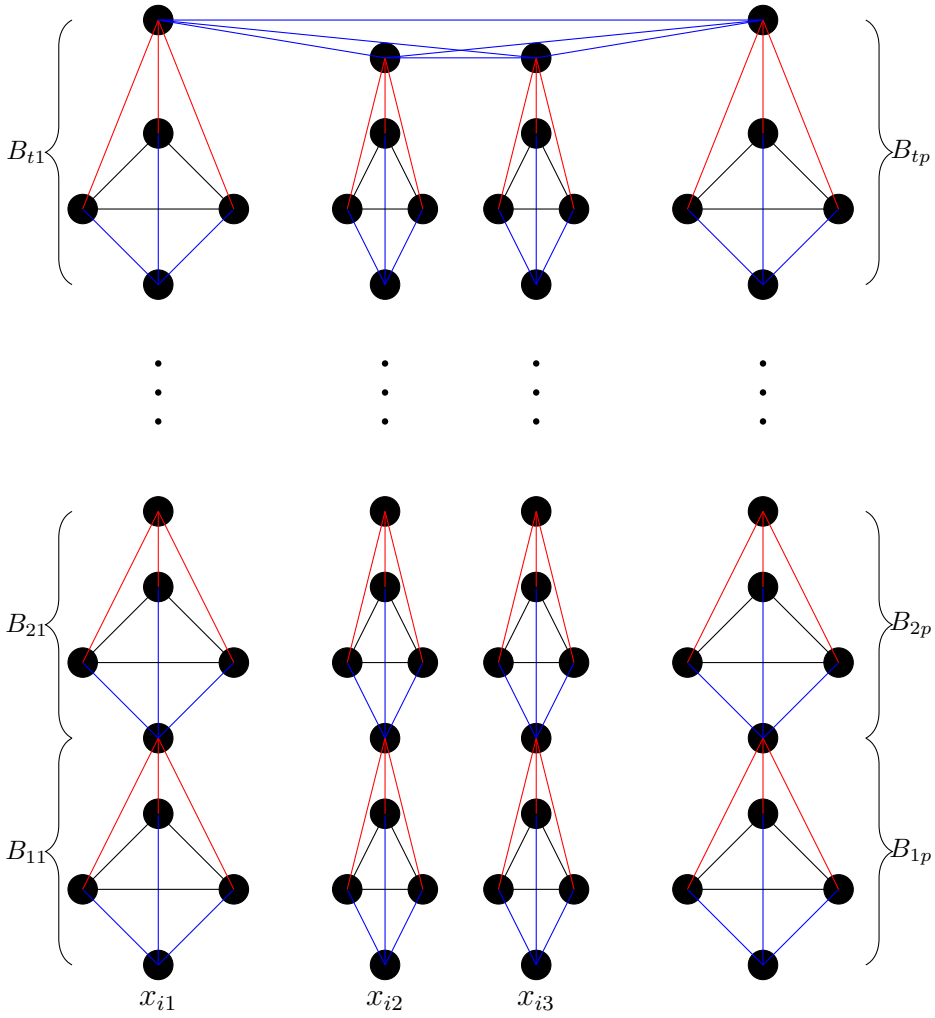


Figure 3.1: Part of graph G constructed for the subset $S_i = \{x_{i1}, x_{i2}, x_{i3}\}$. The graph is constructed by local replacement for $p = 4$. The vertices labeled x_{ij} correspond to the elements in the universe U. If S_i is included in the exact cover then the edges colored black and the edges colored blue represent the corresponding clustering of this part of the graph G. If S_i is not included in the exact cover then the edges colored red and the edges colored black represent the clustering of this part of the graph.

clustering makes no negative errors. Furthermore, each point is in a cluster of size exactly p. Thus, the positive error corresponding to any vertex is the degree of that vertex minus $p - 1$. Since, the size of a cluster is at most p, this is the minimum possible positive error

for any vertex. Hence, any other clustering strictly makes more positive errors than C.

It is easy to see from the construction that if A finds a clustering which has no negative errors and all the clusters have size p, then this corresponds to exact cover of the X3C instance and hence we output YES. If this does not happen then there does not exist any exact cover for (U, S). This is because if there was an exact cover then the corresponding clustering would satisfy our condition. Thus, A decides X3C. Since, X3C is NP-hard, no polynomial time algorithm A exists unless $P = NP$.

In the construction, for each clause, we have $p^2t + (p-3)$ vertices and a vertex for each of the variables. Therefore, $|V| = m(p^2t + (p-3)) + 3q$ and $|E| = pt(\binom{p}{2} + p - 1) + \binom{p}{2}$. Consider a clustering C which places all the x_i's and r_i's in singleton clusters and places rest of the points in clusters of size p. For $t \geq 2$,

$$\beta = \frac{pt\binom{p}{2}}{pt(\binom{p}{2} + p - 1) + \binom{p}{2}} = \frac{1}{1 + \frac{2}{p} + \frac{1}{pt}} > \frac{1}{2} \text{ and } \alpha = 0$$

3.3.2 Hardness of PCC in the presence of an oracle

In the previous sections, we have shown that the PCC problem is NP-hard without queries. It is trivial to see that if $\alpha = 0$ then making $\beta|X|$ queries to the same-cluster oracle allows us to solve (in polynomial time) the Promise Correlation Clustering problem for all p. In this section, we prove that the linear dependence on $n = |X|$ is tight if we assume that the exponential time hypothesis (ETH) is correct.

ETH is an assumption about computational hardness of the 3SAT problem. The non-uniform version of ETH (which we will use in this section) posits that there does not exist an algorithm that can solve 3-SAT in $2^{o(n)}$ time. We prove that if the exponential time hypothesis (ETH) holds then any algorithm that runs in polynomial time makes at least $\Omega(n)$ same-cluster queries. The main idea is to reduce 3SAT to our problem. We do this by the following sequence of reductions, 3SAT to 3DM to X3C to PCC. These reductions show that (under ETH) the running time of PCC is $2^{o(n)}$. Using a 'simulation trick' we then prove that any query-based algorithm for PCC needs to make at least $\Omega(n)$ queries.

Theorem 3.9. *Given that the Exponential Time Hypothesis (ETH) holds then any algorithm for the Promise Correlation Clustering problem that runs in polynomial time makes $\Omega(|X|)$ same-cluster queries for all $p \geq 3$ and for $\alpha = 0$ and $\beta = \frac{1}{2}$.*

We will now prove the above theorem. We state the definition of the different problems used in the reduction.

Definition 3.10. *3-SAT.*
Input: A boolean formulae ϕ in 3CNF with n literals and m clauses. Each clause has exactly three literals.
Output: YES if ϕ is satisfiable, NO otherwise.

Definition 3.11. *3 Dimensional Matching (3DM)*
Input: Sets W, X and Y and a set of matches $M \subseteq W \times X \times Y$ of size m.
Output: YES if there exists $M' \subseteq M$ such that each element of W, X, Y appears exactly once in M'. NO otherwise.

We first reduce 3-SAT to 3-dimensional matching problem. 3DM is already known to be NP-hard. However, the standard reduction of 3-SAT to 3DM constructs a set with $|M| \in \Theta(m^2n^2)$. Hence, using the standard reduction, the exponential time hypothesis would imply there does not exist an algorithm for 3DM which runs in $\Omega(m^{\frac{1}{4}})$. Our reduction is based on the standard reduction. However, we make some clever optimizations especially in the way we encode the clauses. This helps us improve the lower bound to $\Omega(m)$.

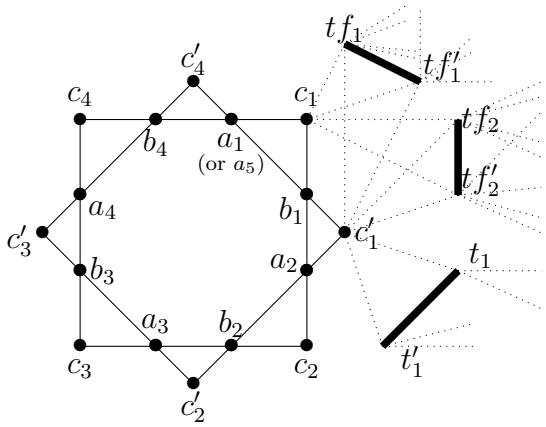


Figure 3.2: Part of graph G constructed for the literal x_1. The figure is an illustration for when x_1 is part of four different clauses. The triangles (or hyper-edge) (a_i, b_i, c_i) capture the case when x_1 is true and the other triangle (b_i, c'_i, a_{i+1}) captures the case when x_1 is false. Assuming that a clause $C_j = \{x_1, x_2, x_3\}$, the hyper-edges containing tf_i, tf'_i and t_1, t'_1 capture different settings. The hyper-edges containing t_1, t'_1 ensure that at least one of the literals in the clause is true. The other two ensure that two variables can take either true or false values.

Our gadget is described in Fig. 3.2. For each literal x_i, let m_i be the number of clauses in which the literal is present. We construct a "truth-setting" component containing $2m_i$

hyper-edges (or triangles). We add the following hyper-edges to M.

$$\{(a_k[i], b_k[i], c_k[i]) : 1 \leq k \leq m_i\} \cup \{(a_{k+1}[i], b_k[i], c'_k[i]) : 1 \leq k \leq m_i\}$$

Note that one of (a_k, b_k, c_k) or (a_{k+1}, b_k, c'_k) have to be selected in a matching M'. If the former is selected that corresponds to the variable x_i being assigned true, the latter corresponds to false. This part is the same as the standard construction.

For every clause $C_j = \{x_1, x_2, x_3\}$ we add three types of hyper-edges. The first type ensures that at least one of the literals is true.

$$\{(c_k[i], t_1[j], t'_1[j]) : x'_i \in C_j\} \cup \{(c'_k[i], t_1[j], t'_1[j]) : x_i \in C_j\}$$

The other two types of hyper-edges (conected to the tf_i's) say that two of the literals can be either true or false. Hence, we connect them to both c_k and c'_k

$$\begin{aligned}&\{(c_k[i], tf_1[j], tf'_1[j]) : x'_i \text{ or } x_i \in C_j\} \cup \{(c_k[i], tf_2[j], tf'_2[j]) : x_i \text{ or } x'_i \in C_j\} \\ &\cup \{(c'_k[i], tf_1[j], tf'_1[j]) : x'_i \text{ or } x_i \in C_j\} \cup \{(c'_k[i], tf_2[j], tf'_2[j]) : x_i \text{ or } x'_i \in C_j\}\end{aligned}$$

Note that in the construction k refers to the index of the clause C_j in the truth-setting component corresponding to the literal x_i. Using the above construction, we get that

$$\begin{aligned}W &= \{c_k[i], c'_k[i]\} \\ X &= \{a_k[i]\} \cup \{t_1[j], tf_1[j], tf_2[j]\} \\ Y &= \{b_k[i]\} \cup \{t'_1[j], tf'_1[j], tf'_2[j]\}\end{aligned}$$

Hence, we see that $|W| = 2\sum_i m_i = 6m$. Now, $|X| = |Y| = \sum_i m_i + 3m = 6m$. And, we have that $|M| = 2\sum_i m_i + 15m = 21m$. Thus, we see that this construction is linear in the number of clauses.

Now, if the 3-SAT formula ϕ is satisfiable then there exists a matching M' for the 3DM problem. If a variable $x_i = T$ in the assignment then add $(c_k[i], a_k[i], b_k[i])$ to M' else add $(c'_k[i], a_{k+1}[i], b_k[i])$. For every clause C_j, let x_i (or x'_i) be the variable which is set to true in that clause. Add $(c'_k[i], t_1[j], t'_1[j])$ (or $(c_k[i], t_1[j], t'_1[j])$) to M'. For the remaining two clauses, add the hyper-edges containing $tf_1[j]$ and $tf_2[j]$ depending upon their assignments. Clearly, M' is a matching.

Now, the proof for the other direction is similar. If there exists a matching, then one of (a_k, b_k, c_k) or (a_{k+1}, b_k, c'_k) have to be selected in a matching M'. This defines a truth assignment of the variables. Now, the construction of the clause hyper-edges ensures that every clause is satisfiable.

Theorem 3.12. *If the exponential time hypothesis holds then there does not exist an algorithm which decides the three dimensional matching problem 3DM and runs in time* $2^{o(m)}$.

Corollary 3.13. *If the exponential time hypothesis holds then there does not exist an algorithm which decides exact cover by 3-sets problem (X3C) and runs in time* $2^{o(m)}$.

Hence, from the discussion in this section, we know that X3C is not only NP-hard but the running time is lower bounded by $\Omega(2^m)$. Now, using the same reduction of X3C to PCC as before, gives the same lower bound on the running time of PCC. Using this, we can now lower bound the number of queries required by PCC.

For the sake of contradiction, let us assume that there exists an algorithm which solves PCC in polynomial time by making $o(n)$ same-cluster queries (n is the number of vertices). Then by simulating all possible answers for the oracle, we get a non-query algorithm which solves PCC in $2^{o(n)}$. However, combining Cor. 3.13 with the reduction of X3C to PCC, we get that any algorithm that solves PCC takes $\Omega(2^n)$. Hence, no such query algorithm exists.

3.4 Restricted Correlation Clustering

The results in the previous section show that even under strong promise, correlation clustering is still NP-hard. Furthermore, it is hard even when given access to an oracle.

Observe that the requirement of correlation clustering is very demanding. The algorithm is required to find a clustering over the set of all possible clusterings of the domain X. In the restricted framework, we change the goalpost slightly. The algorithm is now required to find a clustering C from a class $\mathcal{F}$ (of clusterings of X).

Definition 3.14 (Restricted correlation clustering (RCC))**.** *Given a clustering instance* (X, d)*, an unknown target clustering* C^* *and weighting parameter* μ*. Given a finite class* $\mathcal{F}$ *of clusterings of the set* X*. Find* $C \in \mathcal{F}$ *such that*

$$\hat{C} = \arg\min_{C \in \mathcal{F}} \; L_{C^*}(C) \tag{3.7}$$

3.4.1 Relation to practical applications

Consider the following scenario from the practitioner's point of view. The practitioner wants to implement correlation clustering. However, he/she knows that the problem is NP-hard. The practitioner has prior knowledge that one of the many hierarchical clustering

algorithms (like single-linkage or max-linkage or average-linkage or complete-linkage) is suitable for his/her dataset[1]. A hierarchical clustering algorithm outputs a clustering tree. Every pruning of the tree is a clustering of the original dataset. He/she would now like to know which amongst these clustering algorithms is suitable for his task. After having fixed the algorithm, the practitioner would then like to know which amongst these many prunings he/she should chose.

The framework of restricted correlation clustering is applicable in such scenarios. When $\mathcal{F} = \{T\}$ where T is a hierarchical clustering of X, the goal of RCC is to find the pruning from the tree T which has minimum normalized correlation loss. When $\mathcal{F} = \{T_1, \ldots, T_s\}$ where each T_i is a hierarchical clustering of X. Then the goal of RCC is to find a pruning with minimum loss amongst the prunings of all the s trees. Note that finding the pruning of the tree is the same as choosing the stopping point criteria when running linkage-based algorithms. Hence, the framework can help us choose the right stopping point for a particular hierarchical clustering algorithm.

If $\mathcal{F} = \{C_1, \ldots, C_s\}$ where each C_i is a clustering of the set X then the goal is to find a clustering with minimum loss. Note that $\mathcal{F}$ can be any of the examples as defined above or a union of these or some other finite class.

3.4.2 Solution strategy

In the RCC framework, we wish to minimize the loss which depends on the unknown target clustering C^*. However, in the absence of any information about C^*, there is no hope to find a clustering that minimizes L_{C^*}. Hence, to solve the RCC problem we allow the clustering (or learning) algorithm to make queries to a C^*-oracle.

Our goal is to calculate quantities $L_{P^+}(C)$ and $L_{P^-}(C)$ (Defn. 3.3) for each of the clusterings $C \in \mathcal{F}$ and then choose the clustering with minimum loss. To calculate both these quantities exactly, for each pair of points in our dataset, we would need to know whether they belong to the same-cluster or different-cluster. In other words, we would need access to the complete ground truth clustering C^*. Thus, instead of calculating these two quantities exactly we want to estimate them from a small sample, sampled according to the distributions P^+ and P^-.

One strategy to estimate $L_{P^+}(C)$ (and L_{P^-}) could be the following. Sample a set S_+ (and S_-) of pairs using the distribution P^+ (and P^-). Compute the fraction of mistakes made by each clustering C on S_+ (and S_-). Using the standard results from vc-dimension

[1] A nice overview of hierarchical clustering techniques can be found in [Maimon and Browarnik, 2009]

theory (Thm. 1.2), it is known that using this procedure we can estimate L_{P^+} for each of the clusterings $C \in \mathcal{F}$. Similarly, we could also estimate L_{P^-}. Using the two estimates, we could then estimate the loss L_{C^*} for each of the clusterings in our class and choose the clustering which has the smallest loss.

The main problem in this approach is that the distributions P^+ and P^- are unknown (as the target clustering C^* is not known). In Section 3.5, we discuss two approaches which (approximately) sample according to these distributions. Then in Section 3.6, we show how these sampling procedures can be used to estimate L_{C^*} for all the clusterings in our class $\mathcal{F}$.

3.5 Sampling for RCC

We first describe the procedure $\mathcal{P}_0$ which samples according to P^-. Then we describe the procedure $\mathcal{P}_1$ which samples approximately according to the distribution P^+.

Algorithm 2: Procedure $\mathcal{P}_0$ for negative pairs

Input: A set X and a C^*-oracle.
Output: (x, y) such that $C^*(x, y) = 0$

```
while TRUE do
    Sample (x, y) using U^2
    if C*(x, y) = 0 then
        Output (x, y)
    end
end
```

$\mathcal{P}_0$ is a straightforward rejection sampling procedure. It samples a pair uniformly at random. Then using the oracle it checks if the sampled pair is negative and terminates if such a pair is found. If not then the process is repeated again.

Lemma 3.15. *Given X and a C^*-oracle. The procedure $\mathcal{P}_0$ samples a pair (x, y) according to the distribution P^-.*

Proof. The probability that a negative pair is sampled during a trial is $U^2(X^{[2]-}) =: q$. Fix a negative pair (x, y) and let $U^2(x, y) = p$. Hence, the probability that the pair (x, y) is sampled $= p + (1-q)p + (1-q)^2 p + \ldots = p \sum_{i=0}^{\infty} (1-q)^i = \frac{p}{q} = \frac{U^2(x,y)}{U^2(X^{[2]-})} = P^-(x, y)$. □

Note that to sample one negative pair, procedure $\mathcal{P}_0$ might need to ask more than one same-cluster query. However, since our input X is γ-skewed, we 'expect' the number of 'extra' queries to be 'small'.

Lemma 3.16. *Given set X and a C^*-oracle. Let X be γ-skewed and Let q be the number of same-cluster queries made by $\mathcal{P}_0$ to the C^*-oracle. Then, $\mathbf{E}[q] \leq \frac{1}{1-\gamma}$.*

Proof. Let p denote the probability that a negative pair is sampled during an iteration. We know that $p \geq (1-\gamma)$. Let q be a random variable denoting the number of iterations (or trials) before a negative pair is sampled. Then, q is a geometric random variable. $\mathbf{E}[q] = \frac{1}{p} \leq \frac{1}{1-\gamma}$. □

Lemma 3.16 shows that for $\gamma < \frac{1}{2}$, to sample a negative pair, procedure $\mathcal{P}_0$ makes at most two queries to the oracle in expectation. Moreover, the number of queries is tight around the mean. Note that this sampling strategy is not useful for positive pairs. This is because the fraction of positive pairs in the dataset is small. Hence, to sample a single positive pair we would need to make 'many' same-cluster queries.

3.5.1 Sampling positive pairs for general metrics

Algorithm 3: Sampling procedure $\mathcal{P}_{11}$ for positive pairs (general metrics)

Input: A set X, a C^*-oracle and a parameter λ.
Output: One pair $(x, y) \in X^{[2]}$ such that $\mathcal{C}^*(x, y) = 1$

Pre-compute: For all $x \in X$, compute $S_x := \{y : d(x, y) \leq \lambda\}$.
while *TRUE* **do**
    Sample $x \in X$ with probability $\propto |S_x|$.
    Sample y uniformly at random from S_x.
    if $\mathcal{C}^*(x, y) = 1$ **then**
        Output (x, y).
    end
end

Given a clustering instance (X, d). Assume that the metric d is (α, β)-informative w.r.t target C^* and parameter λ. This means that 'most' of the positive pairs are within distance

λ. Our sampling strategy is to "construct" a set $K = \{(x, y) \in X^2 : d(x, y) \leq \lambda\}$ and then sample uniformly from this set.

The sampling algorithm is described in Alg. 3. In the pre-compute stage, for all points x we construct its set of 'neighbours' (S_x). We then choose a point with probability proportional to the size of its neighbour-set and then choose the second point uniformly at random from amongst its neighbours. This guarantees that we sample uniformly from the set K.

Below we prove that the procedure $\mathcal{P}_{11}$ samples according to a distribution T which approximates P^+. The ideas used in the proof could actually be used to show that the total variation distance between T and P^+ is at most 2α. Thm. 3.17 could then be obtained as a corollary of the more general theorem. However, for our purposes the result of Thm. 3.17 suffices and we don't need the more general statement.

Theorem 3.17. *Given set (X, d), a C^*-oracle and parameter λ. Let d be (α, β)-informative w.r.t λ and C^*. Then the sampling procedure $\mathcal{P}_{11}$ induces a distribution T over $X^{[2]}$ such that for any labelling function h over $X^{[2]}$ we have that*

$$\left| \mathop{\mathbf{P}}_{(x,y)\sim P^+} \left[h(x, y) = 0\right] - \mathop{\mathbf{P}}_{(x,y)\sim T} \left[h(x, y) = 0\right] \right| \leq 2\alpha.$$

Proof. Let $K = \{(x, y) : d(x, y) \leq \lambda\}$ and D be a distribution over K defined by $D(x, y) := \frac{|S_x|}{\sum_{x'} |S_{x'}|} \cdot \frac{1}{|S_x|} = \frac{U^2(x,y)}{U^2(K)}$. Let $K^+ = \{(x, y) : d(x, y) \leq \lambda \text{ and } C^*(x, y) = 1\}$. Let T be the distribution induced by $\mathcal{P}_{11}$. It's easy to see that for $(x, y) \notin K^+$, $T(x, y) = 0$. For $(x, y) \in K^+$, let $D(x, y) = p$ and $D(K^+) = q$. Then, $T(x, y) = p + (1 - q)p + \ldots = \frac{p}{q} = \frac{D(x,y)}{D(K^+)} = \frac{U^2(x,y)}{U^2(K^+)}$. Using Defn. 3.4, we know that

$$\begin{aligned} 1 - \alpha &\leq \mathop{\mathbf{P}}_{(x,y)\sim U^2}[d(x, y) \leq \lambda \mid C^*(x, y) = 1] \\ &= \frac{\mathop{\mathbf{P}}_{(x,y)\sim U^2}[d(x, y) \leq \lambda, C^*(x, y) = 1]}{\mathop{\mathbf{P}}_{(x,y)\sim U^2}[C^*(x, y) = 1]} = \frac{U^2(K^+)}{U^2(X^{[2]+})} \end{aligned} \tag{3.8}$$

Now, we will use the above inequality to prove our result.

$$\begin{aligned}
&\mathop{\mathbf{P}}_{(x,y)\sim T}\big[h(x,y)=0\big] = \sum_{(x,y)\in K^+} T(x,y)\mathbf{1}_{h(x,y)=0} \\
&= \sum_{(x,y)\in K^+} \frac{U^2(x,y)}{U^2(K^+)}\mathbf{1}_{h(x,y)=0} \leq \frac{1}{1-\alpha}\sum_{(x,y)\in K^+}\frac{U^2(x,y)}{U^2(X^{[2]+})}\mathbf{1}_{h=0} \\
&\leq (1+2\alpha)\sum_{(x,y)\in X_+^2} P^+(x,y)\mathbf{1}_{h(x,y)=0} = (1+2\alpha)\mathop{\mathbf{P}}_{(x,y)\sim P^+}\big[h(x,y)=0\big]
\end{aligned}$$

Now, for the other direction, we have that

$$\begin{aligned}
&\mathop{\mathbf{P}}_{(x,y)\sim P^+}\big[h(x,y)=0\big] = \sum_{(x,y):X^{[2]+}} P^+(x,y)\mathbf{1}_{h(x,y)=0} \\
&= \sum_{(x,y)\in K^+}\frac{U^2(x,y)}{U^2(X^{[2]+})}\mathbf{1}_{h(x,y)=0} + \sum_{(x,y)\in X_+^2\setminus K^+}\frac{U^2(x,y)}{U^2(X^{[2]+})}\mathbf{1}_{h=0} \\
&\leq \sum_{(x,y)\in K^+}\frac{U^2(x,y)}{U^2(K^+)}\mathbf{1}_{h(x,y)=0} + \sum_{(x,y)\in X^{[2]+}\setminus K^+}\frac{U^2(x,y)}{U^2(X^{[2]+})}\mathbf{1}_{h=0} \\
&\leq \mathop{\mathbf{P}}_{(x,y)\sim T}\big[h(x,y)=0\big] + \sum_{(x,y)\in X^{[2]+}\setminus K^+}\frac{U^2(x,y)}{U^2(X^{[2]+})} \leq \mathop{\mathbf{P}}_{(x,y)\sim T}\big[h(x,y)=0\big]+\alpha
\end{aligned}$$

Hence, we have shown that both the directions hold and this completes the proof of the lemma. Note that this shows that our sampling procedure approximates the distribution P^+. It is easy to see that pre-computing S_x for all x takes $|X|^2$ time. Once the pre-computation is done, the sampling can be done in constant time. □

Note that to sample one positive pair, procedure $\mathcal{P}_{11}$ might need to ask more than one same-cluster query. However, since the metric d is β-informative, we 'expect' the number of 'extra' queries to be 'small'.

Lemma 3.18. *Given set (X,d), a C^*-oracle and a parameter λ. Let d be β-informative w.r.t λ and let q be the number of same-cluster queries made by $\mathcal{P}_{11}$ to the C^*-oracle. Then, $\mathbf{E}[q] \leq \frac{1}{\beta}$.*

Proof. Let p denote the probability that a positive pair is sampled during an iteration. We know that $p \geq \beta$. Let q be a random variable denoting the number of iterations (or trials) before a positive pair is sampled. Then, q is a geometric random variable. $\mathbf{E}[q] = \frac{1}{p} \leq \frac{1}{\beta}$. □

3.5.2 Sampling positive pairs for LSHable metrics

The strategy in the previous section was to construct a set $K = \{(x, y) : d(x, y) \leq \lambda\}$ and then sample uniformly from the set K till a positive sample is found. Since most of the positive pairs have distance $\leq \lambda$, this sampling procedure approximates P^+ (the uniform distribution over the set of true positives). However, constructing the set K requires $\Theta(|X|^2)$ pre-processing time. This can be prohibitive in many practical applications.

In this section, we show that if the metric d has some additional structure (is hashable) then we can reduce the pre-processing time to $O(|X|)$. We develop a sampling procedure $\mathcal{P}_{12}$ using techniques from locality sensitive hashing (LSH) combined with rejection sampling. We will show that $\mathcal{P}_{12}$ needs only linear pre-processing time (to build the hash maps) and outputs a positive pair sampled approximately according to P^+. Note that it is still an open question whether this pre-processing time can be reduced further to $o(|X|)$.

Locality Sensitive Hashing (LSH)

Before we describe our technique, we introduce some relevant notation. A hash function $h : X \to \mathbf{N}$ maps the set X onto the set of natural numbers. Thus, a hashing function partitions the input of size n into $m \leq n$ different buckets (or blocks) $B_1, \ldots, B_m$ where each $B_i = \{x : h(x) = b_i\}$ for some b_i. Given (X, d), a Locality Sensitive Hashing (LSH) scheme w.r.t the distance metric d (or a similarity metric) aims to partition X into buckets such that 'similar' items map to the same bucket with high probability and 'dissimilar' items end up in different buckets with high probability. For example, MinHash scheme w.r.t Jaccard similarity measure [Broder et al., 2000, Broder, 1997] is a common LSH-based hashing scheme. Another example is SimHash scheme w.r.t hamming similarity measure [Charikar, 2002].

Definition 3.19 (LSH-based hashing algorithm)**.** *Given a set (X, d) and parameter s. An LSH-based hashing algorithm (or scheme) $\mathcal{A}$ outputs s different partitions $P_1, \ldots, P_s$ of X. Denote $P_i = \{B_{i1}, \ldots, B_{in_i}\}$. We say that $\mathcal{A}$ is (ϵ, ϵ')-tight w.r.t d and λ, λ' if*

- *If $d(x, y) \leq \lambda$ then $\mathbf{P}[b(x, y) = 1] > 1 - \epsilon$*
- *If $d(x, y) > \lambda'$ then $\mathbf{P}[b(x, y) = 1] < \epsilon'$*

where $b(x, y) = 1$ if and only if x, y are together in at least one of the blocks B_{ij}.

Infact, we show that by choosing s (and other parameters) appropriately, we can construct LSH schemes which are $(\epsilon, \epsilon' = s\ln(1+\epsilon))$-tight w.r.t λ and $\lambda' = 2\lambda\ln(1+1/\epsilon)$. Thus,

for simplicity of notation, we say that $\mathcal{A}$ is ϵ-tight w.r.t λ to mean that it is (ϵ, ϵ')-tight w.r.t λ, λ' as chosen above.

Throughout the remainder of this section, we will assume that the hashing scheme satisfies ϵ-tightness. In the appendix, we provide details about why this assumption is justified. However, these results are orthogonal to the current discussion. Hence, we omit it here and only include it in the appendix (Thm. B.3).

We now describe our sampling procedure. Let $\mathcal{B} := \{P_1, \ldots, P_s\} = \{B_{ij} : 1 \leq i \leq s, 1 \leq j \leq |P_i|\}$ be the set of blocks outputted by the hashing scheme and let $Q := \{(x, y) \in B_{ij}\}$. We first choose a block $B \in \mathcal{B}$ with probability proportional to $|B|^2$ (the number of pairs in the block). Then we sample a pair uniformly at random from this block B. Note that this strategy doesn't give us a uniform sample from Q. This is because a pair (x, y) may be present in multiple blocks. To get the uniform sample, we reject the pair with probability inversely proportional to $a(x, y)$ (the number of blocks in which x, y are together). This approach based on rejection sampling ensures that we have a uniform sample from Q.

Next, we check if the pair satisfies $d(x, y) \leq \lambda$. Note that the LSH-based scheme tries to put similar points in the same bucket, hence the probability of success at this step is 'high'. Finally, we check if $C^*(x, y) = 1$. Our sampling procedure $\mathcal{P}_1$ is described in Alg. 12.

Thm. 3.20 shows that with high probability the procedure $\mathcal{P}_{12}$ samples a pair according to a distribution $\mathcal{T}$ which approximates P^+.

Theorem 3.20. *Given (X, d), a C^*-oracle and parameter λ. Let d satisfy (α, β)-informative w.r.t C^*. Let the hashing algorithm $\mathcal{A}$ satisfy ϵ-tightness w.r.t λ. Then with probability at least $1 - \exp(-2(\nu(1-\alpha)|X_+^2|)^2)$ (over the randomness in the hashing algorithm), $\mathcal{P}_{12}$ samples pairs (x, y) according to distribution $\mathcal{T}$ over $X^{[2]}$ such that for any labelling function $C : X^{[2]} \to \{0, 1\}$, we have that*

$$\begin{aligned} &\mathop{\mathbf{P}}_{(x,y)\sim P^+}\big[C(x, y) = 0\big] - \alpha - \epsilon - \nu \leq \mathop{\mathbf{P}}_{(x,y)\sim T}\big[C(x, y) = 0\big] \\ &\leq (1 + 2\nu)(1 + 2\alpha) \mathop{\mathbf{P}}_{(x,y)\sim P^+}\big[C(x, y) = 0\big] \end{aligned}$$

Proof. Let $Q := \{(x, y) : b(x, y) = 1\} = \{(x, y) \in B^2 : B \in \mathcal{B}\}$. Let $K = \{(x, y) : d(x, y) \leq \lambda\}$, let $K_Q = K \cap Q$, let $K^+ := \{(x, y) \in K : C^*(x, y) = 1\}$ and finally let $K_Q^+ = \{(x, y) \in K_Q : C^*(x, y) = 1\}$. Note that the choice of Q depends upon the hashing algorithm $\mathcal{A}$. However, after the pre-compute stage, the set Q is fixed and the sampling procedure samples from the set Q. Our procedure works in four steps.

Algorithm 4: Sampling procedure $\mathcal{P}_{12}$ for positive pairs

Input: A set $\mathcal{X}$, a hashing algorithm $\mathcal{A}$, a C^*-oracle and parameter λ.
Output: (x, y) such that $C^*(x, y) = 1$

Pre-compute:

Use an LSH-based hashing scheme $\mathcal{A}$ to obtain partitions $\{P_1, \ldots, P_s\}$.
$\mathcal{B} := \{P_1, \ldots, P_s\} = \{B_{ij} : 1 \leq i \leq s, 1 \leq j \leq |P_i|\}$.

Sampling:

while *TRUE* **do**
    Sample a block B from $\mathcal{B}$ with probability $\propto |B|^2$.
    Sample (x, y) uniformly at random from B^2.
    Let $a(x, y) = \{(x, y) \in B^2 : B \in \mathcal{B}\}$.
    Sample u uniformly at random from $[0, 1]$.
    if $u > \frac{1}{|a(x,y)|}$ **then**
        continue.
    end
    if $d(x, y) \leq \lambda$ *and* $C^*(x, y) = 1$ **then**
        Output (x, y).
    end
end

S.1 $\mathcal{P}_1$ samples a point (x, y) from the set Q and induces a distribution D_1 on Q.

$$D_1(x, y) = \sum_{B \in a(x,y)} \frac{|B|^2}{\sum_{B' \in \mathcal{B}} |B'|^2} \frac{1}{|B|^2} = \frac{|a(x, y)|}{\sum_{B' \in \mathcal{B}} |B'|^2}$$

S.2 Next, we reject the sampled point with some probability thereby inducing another distribution D_2 on Q. Now, $D_2(x, y)$ satisfies the following recurrence

$$D_2(x, y) = D_1(x, y) \frac{1}{|a(x, y)|} + \Big(1 - \sum_{(x', y') \in Q} D_1(x', y') \frac{1}{|a(x', y')|}\Big) D_2(x, y)$$

The recurrence basically says that the probability that the pair (x, y) is sampled is equal to the probability that it is sampled during the current round or nothing is sampled during the current round and then (x, y) is sampled. Simplifying the above

equation, we get that

$$D_2(x,y) = \frac{\frac{1}{\sum_{B' \in \mathcal{B}} |B'|^2}}{\sum_{(x',y') \in Q} \frac{1}{\sum_{B' \in \mathcal{B}} |B'|^2}} = \frac{1}{|Q|}$$

S.3 We reject the sampled point if $(x,y) \notin K_Q$. In this step, we induce a distribution D_3 on K_Q. It is easy to see that $D_3(x,y) = \frac{1}{|K_Q|}$

S.4 Next, we reject the sampled point if $(x,y) \notin K_Q^+$. After this step, we induce a distribution D_4 on K_Q^+. $T(x,y) := D_4(x,y) = \frac{1}{|K_Q^+|}$

Another observation, which will be useful later in the proof is that for any $(x,y) \in K^+$, $\mathbf{P}[(x,y) \notin K_Q^+] < \delta$ (Thm. B.3). Hence, hoeffding's inequality we get that, $\mathbf{P}\left[|K^+ \setminus K_Q^+| < (\delta+\nu)|K^+|\right] \geq 1 - \exp(-2\nu^2|K^+|^2)$

Next, we will show that the distribution T is an approximation of P^+. First, observe that $\mathcal{X}$ satisfies μ-nazdeek property. Hence, we get that

$$1 - \alpha \leq \mathbf{P}[d(x,y) \leq \lambda | C^*(x,y) = 1] = \frac{|K^+|}{|X_+^2|} \tag{3.9}$$

Now, let h be any labelling function over $\mathcal{X}^2$.

$$\mathop{\mathbf{P}}_{(x,y)\sim T}[C(x,y) = 0] = \frac{1}{|K_Q^+|} \sum_{(x,y) \in K_Q^+} \mathbf{1}_{[C(x,y)=0]} \leq \frac{1}{|K_Q^+|} \sum_{(x,y) \in X_2^+} \mathbf{1}_{[C(x,y)=0]}$$

Now, with probability at least $1 - \exp(-2\nu^2|K^+|^2) \geq 1 - \exp(-2\nu^2(1-\alpha)^2|X_+^2|^2)$ over the randomness in $\mathcal{A}$, we have that $|K_Q^+| > (1-\nu-\delta)|K^+|$. Substituting this in the above equation gives

$$\mathop{\mathbf{P}}_{(x,y)\sim T}[C(x,y) = 0] \leq \frac{1}{(1-\nu)(1-\delta)|K^+|} \sum_{(x,y) \in X_+^2} \mathbf{1}_{[C(x,y)=0]} \leq \frac{\mathop{\mathbf{P}}_{(x,y)\sim P^+}[C(x,y) = 0]}{(1-\nu-\delta)(1-\alpha)}$$

Now for the other direction, we have that

$$\begin{aligned}
&\mathop{\mathbf{P}}_{(x,y)\sim P^+}[C(x,y) = 0] = \frac{1}{|X_+^2|} \sum_{(x,y) \in X_+^2} \mathbf{1}_{[C(x,y)=0]} \\
&\leq \frac{1}{|K_Q^+|} \sum_{(x,y) \in K_Q^+} \mathbf{1}_{[C(x,y)=0]} + \frac{|X_+^2 \setminus K_Q^+|}{|X_+^2|} \\
&\leq \mathop{\mathbf{P}}_{(x,y)\sim T}[C(x,y) = 0] + \frac{|X_+^2 \setminus K^+|}{|X_+^2|} + \frac{|K^+ \setminus K_Q^+|}{|K^+|} \leq \mathop{\mathbf{P}}_{(x,y)\sim T}[C(x,y) = 0] + \alpha + \nu + \delta
\end{aligned}$$

Now choosing, $\delta = \epsilon$ gives the result of the theorem. □

To sample one same-cluster pair, we might need to make more than one same-cluster query to the C^*-oracle. Lemma 3.21 shows that with high probability, the number of queries made by $\mathcal{P}_{12}$ to sample one positive pair is upper bounded by a small constant.

Lemma 3.21. *Given set X, a C^*-oracle and parameter λ. Let d be (α, β)-informative w.r.t λ and C^*. Let $\mathcal{A}$ satisfy ϵ-tightness w.r.t λ. Let q be the number of same-cluster queries made by $\mathcal{P}_{12}$. Then with probability at least $1 - \exp(-\nu^2(1-\alpha)^2|X_+^2|^2)$ (over the randomness in the hashing algorithm)*

$$\mathbf{E}[q] \leq \frac{1}{\beta(1-\epsilon-\nu)}$$

Proof. Let K, Q, K^+, K_Q and K_Q^+ be as defined in the proof of Thm. 3.17. Also, let the distributions D_1, D_2, D_3 and D_4 be as defined in the same theorem. Also, from the analysis in bullet **S.4** in Thm. 3.17 with probability at least $1 - \exp(-2\nu^2(1-\alpha)^2|X_+^2|^2)$ we have that $\frac{|K_Q^+|}{|K^+|} \geq (1-\nu-\epsilon)$. Now, we have that X satisfies β-balanced property. Hence,

$$\beta \leq \mathbf{P}[h^*(x,y) = 1 | d(x,y) \leq \lambda] = \frac{|K^+|}{|K|} \leq \frac{|K^+|}{|K_Q|}$$

Combining this we get that $\frac{|K_Q^+|}{|K_Q|} \geq \frac{\beta|K_Q^+|}{|K^+|} \geq (1-\nu-\epsilon)\beta$. Thus, given that a same-cluster query is made, the probability p that it succeeds is at least $(1-\nu-\epsilon)\beta$. □

The pre-compute stage uses a hashing algorithm to obtain s different partitions. From the discussion in the appendix (Thm. B.2), we its easy to see that this runs in $O(n)$ time. Next, we analyse the time taken to sample one same-cluster pair. Thm. 3.22 shows that under reasonable assumptions, the time taken is upper bounded by a constant with high probability.

Theorem 3.22. *Given set X, a C^*-oracle and parameter λ. Let d be (α, β)-informative w.r.t λ and C^*. Let $\mathcal{A}$ satisfy ϵ-tightness w.r.t λ.*

Define $\lambda' = 2\lambda\log(1+\frac{1}{\epsilon})$ and $\epsilon' = \lceil\log(\frac{1}{\epsilon})\rceil(1+\log(\frac{1}{\epsilon}))$. Let $K = \{(x,y) : d(x,y) \leq \lambda\}$ is the set of all pairs of points with distance $\leq \lambda$. Similarly, define sets $K_1 = \{(x,y) : \lambda < d(x,y) \leq \lambda'\}$ and $K_2 = \{(x,y) : d(x,y) > \lambda'\}$. Let $|K_1| \leq \rho_1|K|$ and $\epsilon'|K_2| \leq \rho_2|K|$.

Let t be the time taken to sample one point by procedure $\mathcal{P}_{12}$. Then with probability at least $1 - \exp\left(\frac{-\nu^2(1-\epsilon)(1-\alpha)|X_+^2|}{2}\right) - \exp\left(\frac{-\nu^2\rho_2|K|}{3}\right)$ (over the randomness in the hashing algorithm), we have that

$$\mathbf{E}[t] \leq s^2(1 + \frac{1}{\eta})$$

where $\eta := \frac{(1-\nu)(1-\epsilon)\beta}{(1+\nu)(1+\rho_1+\rho_2)}$.

Proof. Using Thm. B.2, we know that one iteration of the pre-compute stage of $\mathcal{P}_1$ runs in $O(nrsm(\mathcal{B})) = O(n \log\left(\frac{2}{\epsilon}\right) \frac{-1}{\log(1-\lambda)} m(\mathcal{B})) = O(n)$. Next, we analyse the time taken to sample one point.

Let Q, K^+, K_Q^+ be as defined in the proof of Thm. 3.17. Let $\delta = \frac{\epsilon}{2}$. Using Thm. B.3, we know that if $(x, y) \in K$ then $\mathbf{P}[(x, y) \in Q] > 1 - \delta$. Also, if $(x, y) \in K_2$ then $\mathbf{P}[(x, y) \in Q] < \delta'$. For the purposes of analysing the time complexity of the sampling procedure, we can think of $\mathcal{P}_1$ as consisting of the following two steps.

T.1 $\mathcal{P}_1$ samples a pair (x, y) uniformly at random from Q. Thus, the probability of success at this step is $p_1 = \frac{1}{a(x,y)} \geq \frac{1}{s}$.

T.2 If the point also lies in K^+, that is $(x, y) \in K^+$ then it outputs that point. Thus, probability of success at this step is $p_2 := \frac{|K^+ \cap Q|}{|Q|} = \frac{|K_Q^+|}{|Q|}$. Consider the following four sets, $K^+, K' := K \setminus K^+, K_1$ and K_2.

Using multiplicative chernoff's bounds, we get that $\mathbf{P}[|K_Q^+| > (1-\nu)|K^+|(1-\delta)] \geq 1 - \exp(\frac{-\nu^2(1-\delta)|K^+|}{2})$. Similarly, we have that $\mathbf{P}[|K_2 \cap Q| < (1+\nu)|K|\rho_2] \geq 1 - \exp(\frac{-\nu^2\rho_2|K|}{3})$. Also, note that $\mathbf{P}[|K' \cap Q| < (1+\nu)|K'|] = 1$ and $\mathbf{P}[|K_1 \cap Q| < (1+\nu)|K_1|] = 1$. Using these results, we have that $\mathbf{P}[|K_Q^+| > (1-\nu)|K^+|(1-\delta)$ and $|(K_Q^+)^c \cap Q| < (1+\nu)(|K'| + |K_1| + |K|\rho_2)] \geq 1 - \exp(\frac{-\nu^2(1-\delta)|K^+|}{2}) - \exp(\frac{-\nu^2\rho_2|K|}{3})$

$$\mathbf{P}\left[\frac{|K_Q^+|}{|(K_Q^+)^c \cap Q|} > \frac{(1-\nu)|K^+|(1-\delta)}{(1+\nu)(|K'| + |K_1| + |K_2|\delta')}\right]$$
$$\geq 1 - \exp\left(\frac{-\nu^2(1-\delta)|K^+|}{2}\right) - \exp\left(\frac{-\nu^2\delta'|K_2|}{3}\right)$$

From the assumptions in the theorem, we get that $|K_1| \leq \rho_1|K|$. Now, $|K'| = |K| - |K^+| \leq (1-\beta)|K|$. Hence, $\frac{(1-\nu)|K^+|(1-\delta)}{(1+\nu)(|K'|+|K_1|+|K|\rho_2)} \geq \frac{(1-\nu)(1-\delta)\beta}{(1+\nu)(1-\beta+\rho_1+\rho_2)}$. Define

$\eta := \frac{(1-\nu)(1-\delta)\beta}{(1+\nu)(1+\rho_1+\rho_2)}$. Then we get that,

$$\mathbf{P}\left[\frac{|K_Q^+|}{|Q|} > \frac{\eta}{\eta+1}\right] \geq 1 - \exp\left(\frac{-\delta^2|K^+|}{(1-\delta)}\right) - \exp\left(\frac{-\delta^2\rho_2|K|}{(1-\delta)^2}\right) =: \eta'$$

Hence, we see that with probability at least η' over the randomness in the hashing procedure $p_2 \geq \eta$.

Using all the above, we get that $p \geq \frac{\eta}{(\eta+1)s}$. Recall that, p is the probability that the current iteration terminates. Thus, expected number of iterations to sample one point is $\leq \frac{s}{\eta}$. Note that the one iteration takes $\Theta(s)$ time (computing $|a(x,y)|$). The expected time to sample one point is $\leq s^2(1+\frac{1}{\eta})$ □

3.6 Sample and query complexity of RCC

In the previous section we saw how to sample (approximately) according to the distributions P^+ and P^-. We sample a 'small' set of true positive (or same-cluster) and true negative (or different-cluster) pairs using our distributions. We then choose the clustering $\hat{C} \in \mathcal{F}$ with the minimum number of mistakes on the sampled pairs. We prove that the true normalized correlation loss $L_{C^*}(C)$ is close to the loss of $\hat{C}^*$ (the clustering with minimum loss in $\mathcal{F}$). Thus, our solution strategy shows that by only having a small amount of information about C^* (making a small number of queries) we can find a clustering which is close (in terms of loss) to the optimal clustering in $\mathcal{F}$. We describe this procedure in Alg. 5.

Note that in this section, we have assumed that procedure $\mathcal{P}_{11}$ is used for sampling positive pairs. Similar results can be obtained when instead procedure $\mathcal{P}_{12}$ is used for sampling positive pairs. However, we include those results only in the appendix.

Thm. 3.23 analyses the sample complexity of our approach. That is, the number of labelled positive and negative pairs our algorithm needs as input, so that the estimates of the loss based on this sample are close to their true values. We show that as long as the number of sampled pairs are in $O(\frac{\text{VC-Dim}(\mathcal{F})}{\epsilon^2})$ then our algorithm finds a clustering $\hat{C}$ which is close to the best clustering in $\mathcal{F}$. Here, VC-Dim is a combinatorial property which measures how 'complex' or rich the class of clusterings is. Note that the number of samples needed is independent of the size of the dataset X.

For common classes, like $\mathcal{F} = \{T_1, \ldots, T_s\}$ where each T_i is a hierarchical clustering of X, we also analyze their VC-Dim($\mathcal{F}$) and prove that it is in $o(\log^2 s)$. Thus for such classes

Algorithm 5: Empirical Risk Minimization

Input: (X, d), a set of clusterings $\mathcal{F}$, a C^*-oracle, parameter λ and sizes m_+ and m_-.

Output: $C \in \mathcal{F}$

1 Sample a sets S_+ and S_- of sizes m_+ and m_- using procedures $\mathcal{P}_{11}$ and $\mathcal{P}_0$ respectively.

2 For every $C \in \mathcal{F}$, compute

$$\hat{P}(C) = \frac{|\{(x,y) \in S_+ : C(x,y) = 0\}|}{|S_+|}$$
$$\hat{N}(C) = \frac{|\{(x,y) \in S_- : C(x,y) = 0\}|}{|S_-|}$$

3 Define $\hat{L}(C) = \mu\hat{P}(C) + (1-\mu)\hat{N}(C)$.

4 Output $\arg\min_{C \in \mathcal{F}} \hat{L}(C)$

a small number of samples suffice to find a clustering which is close to the best clustering in $\mathcal{F}$.

Theorem 3.23. *Given metric space (X, d), a class of clusterings $\mathcal{F}$ and a threshold parameter λ. Given $\epsilon, \delta \in (0,1)$ and a C^*-oracle. Let d be (α, β)-informative and X be γ-skewed w.r.t λ and C^*. Let $\mathcal{A}$ be the ERM-based approach as described in Alg. 5 and $\hat{C}$ be the output of $\mathcal{A}$. If*

$$m_-, m_+ \geq a\frac{\text{VC-Dim}(\mathcal{F}) + \log(\frac{2}{\delta})}{\epsilon^2} \tag{3.10}$$

where a is a global constant then with probability at least $1-\delta$ (over the randomness in the sampling procedure), we have that

$$L_{C^*}(\hat{C}) \leq \min_{\mathcal{C} \in \mathcal{F}} L_{C^*}(\mathcal{C}) + 3\alpha + \epsilon$$

Proof. Let T_0 be the distribution induced by $\mathcal{P}_0$ and T_1 be the distribution induced by $\mathcal{P}_{11}$. Denote by $E(h) = \mathbf{P}_{(x,y)\sim P^+}\left[h(x,y) = 0\right]$ and by $G(h) = \mathbf{P}_{(x,y)\sim P^-}\left[h(x,y) = 1\right]$.

Using Thm. 1.2, we know that if $m_+ > a\frac{\text{VC-Dim}(\mathcal{F})+\log(\frac{1}{\delta})}{\epsilon^2}$ then with probability at least

$1-\delta$, we have that for all h

$$\begin{aligned}
&|\hat{E}(h) - \mathop{\mathbf{P}}_{(x,y)\sim T_1}[h(x,y)=0]| \leq \epsilon \\
\implies &\hat{E}(h) \leq \epsilon + \mathop{\mathbf{P}}_{(x,y)\sim T_1}[h(x,y)=0] \leq \epsilon + 2\alpha + E(h) \quad \text{and} \\
&E(h) - 2\alpha - \epsilon \leq \hat{E}(h)
\end{aligned} \tag{3.11}$$

Note that we obtain upper and lower bounds for $\mathop{\mathbf{P}}_{(x,y)\sim T_1}[h(x,y)=0]$ using Thm. 3.17. Similarly, if $m_- > a\frac{\text{VC-Dim}(\mathcal{F})+\log(\frac{1}{\delta})}{\epsilon^2}$, then with probability at least $1-\delta$, we have that for all h,

$$\begin{aligned}
&|\hat{G}(h) - \mathop{\mathbf{P}}_{(x,y)\sim T_0}[h(x,y)=1]| \leq \epsilon \\
\implies &\hat{G}(h) \leq \epsilon + G(h) \quad \text{and} \quad G(h) - \epsilon \leq \hat{G}(h)
\end{aligned} \tag{3.12}$$

Combining Eqns. 3.11 and 3.12, we get that with probability at least $1-2\delta$, we have that for all $C \in \mathcal{F}$

$$\begin{aligned}
&\hat{L}(C) \leq \mu[\epsilon + E(h) + 2\alpha] + (1-\mu)(\epsilon + G(h)) \\
&\leq L(h) + \epsilon + 2\alpha \\
&\text{And } \hat{L}(C) \geq \mu(E(h) - \epsilon - \alpha) + (1-\mu)(G(h) - \epsilon) \\
&\geq L(h) - \epsilon - \alpha
\end{aligned}$$

Now, let $\hat{C}$ be the output of $\mathcal{A}$ and let $\hat{C}^*$ be $\arg\min_{C\in\mathcal{F}} L(C)$. Then, we have that with probability at least $1-2\delta$

$$L(\hat{C}) \leq \hat{L}(\hat{C}) + \alpha + \epsilon \leq \hat{L}(\hat{C}^*) + \alpha + \epsilon \leq L(\hat{C}^*) + 2\epsilon + 3\alpha$$

Choosing $\epsilon = \frac{\epsilon}{2}$ and $\delta = \frac{\delta}{2}$ throughout gives the result of the theorem. □

Finally, we analyse the query complexity of our approach. That is the number of queries that our algorithm makes to the C^*-oracle. Our algorithm makes queries during the sampling procedure. We see that to sample m_- negative and m_+ positive pairs the number of queries is 'close' to $m_+ + m_-$ with very high probability. Thus, the number of 'wasted' queries is small.

Theorem 3.24. *[Query Complexity] Let the framework be as in Thm. 3.23. With probability at least* $1 - \exp\big(-\frac{\nu^2 m_-}{4}\big) - \exp\big(-\frac{\nu^2 m_+}{4}\big)$ *over the randomness in the sampling procedure, the number of same-cluster queries* q *made by* $\mathcal{A}$ *is*

$$q \leq (1+\nu)\left(\frac{m_-}{(1-\gamma)} + \frac{m_+}{\beta}\right)$$

Proof. Let q_+ denote the number queries to sample the set S_+. We know that $\mathbf{E}[q_+] \leq \frac{1}{\beta}$. Given that the expectation is bounded as above, using Thm. 1.3, we get that $q_+ \leq \frac{(1+\nu)m_+}{\beta}$ with probability at least $1 - \exp(\frac{-\nu^2 m_+}{4})$. Similarly, we get that with probability at least $1 - \exp(\frac{-\nu^2 m_-}{4})$, $q_- \leq \frac{(1+\nu)m_-}{(1-\gamma)}$. □

3.6.1 VC-Dimension of some common classes of clusterings

In the previous section, we proved that the sample complexity of learning a class of clusterings $\mathcal{F}$ depends upon VC-Dim($\mathcal{F}$). Recall that $\mathcal{F}$ is the class of labellings induced by the clusterings in $\mathcal{F}$. In this section, we prove upper bounds on the VC-Dimension for some common class of clusterings.

Theorem 3.25. *Given a finite set* $\mathcal{X}$ *and a finite class* $\mathcal{F} = \{C_1, \ldots, C_s\}$ *of clusterings of* $\mathcal{X}$*.*

$$\text{VC-Dim}(\mathcal{F}) \leq g(s)$$

where $g(s)$ *is the smallest integer* n *such that* $B_{\sqrt{n}} \geq s$ *where* B_i *is the* i^{th} *bell number [A000108,].*

Proof. Let n be as defined in the statement of the theorem. Let $M^2 \subseteq \mathcal{X}^2$ be a set of size $> n$. Define $M := \{x : (x,y) \in M^2 \text{ or } (y,x) \in M^2\}$. We know that $|M| > \sqrt{n}$. The number of clusterings (partitions) on n elements is given by the n^{th} bell number. Thus, for $s \leq B_{\sqrt{n}}$ there exists a clustering $C' \notin \mathcal{F}$ of the set $\mathcal{X}$. Hence, $l_{\mathcal{F}}$ can't shatter any set of size $> n$. □

Note that $B_{\sqrt{n}} \in o(2^n)$. Thus, the VC-Dim of a list of clusterings is in $o(\log s)$. Next, we discuss another common class of clusterings, namely hierarchical clustering trees.

Definition 3.26 (Hierarchical clustering tree)**.** *Given a set* X*. A hierarchical clustering tree* T *is a rooted binary tree with the elements of* X *as the leaves.*

Every pruning of a hierarchical clustering tree is a clustering of the set X. A clustering tree contains exponentially many (in the size of $\mathcal{X}$) clusterings. Given $\mathcal{F} = \{T_1, \ldots, T_s\}$ consists of s different hierarchical clustering trees, the following theorem bounds the VC-Dimension of $\mathcal{F}$.

Lemma 3.27. *Let $\mathcal{X}$ be a finite set, $S \subseteq \mathcal{X}$ be a set of n points and T be any hierarchical clustering tree of $\mathcal{X}$. There exists a set $\mathcal{C} = \{C_1, \ldots, C_s\}$ where each C_i is a clustering of S with the following properties*

- $|\mathcal{C}| \geq \frac{n!}{\lfloor n/2 \rfloor! \ 2^{\lfloor n/2 \rfloor}}$
- *T contains at most one clustering from $\mathcal{C}$.*

Proof. Consider clusterings C_i of S of the following type. Each cluster in C_i contains exactly two points (except possibly one cluster which contains one point if n is odd). One such clustering along with a tree $T$is shown in Fig. 3.3. Let $\mathcal{C}$ be the set of all such clusterings C_i. The number of such clusterings $|\mathcal{C}|$ is

$$\begin{cases} \dfrac{n!}{2^{\frac{n-1}{2}} \frac{n-1}{2}!} & n \text{ is odd} \\ \dfrac{n!}{2^{\frac{n}{2}} \frac{n}{2}!} & n \text{ is even} \end{cases} = \frac{n!}{2^{\lfloor \frac{n}{2} \rfloor} (\lfloor \frac{n}{2} \rfloor)!}$$

For the sake of contradiction, assume that T is a hierarchical clustering tree T of $\mathcal{X}$ which contains C_i and C_j. Since $C_i \neq C_j$, there exists points s_1, s_2 and s_3 such that the following happens. (i) s_1, s_2 are in the same cluster in C_i. s_2, s_3 as well as s_1, s_3 are in different clusters in C_i. (ii) s_1, s_3 are in the same cluster in C_j. s_2, s_3 as well as s_1, s_2 are in different clusters in C_j.

Now, T contains C_i. Hence, there exists a node v such that $s_1, s_2 \in C(v)$ but $s_3 \notin C(v)$. T also contains C_j. Hence, there exists a node u such that $s_1, s_3 \in C(u)$ and $s_2 \notin C(u)$. Both u and v contain the point s_1. Hence, either u is a descendant of v or the other way around. Observe that $s_2 \in C(v)$ but $s_2 \notin C(u)$. Hence, v is not a descendant of u. Similarly, $s_3 \in C(u)$ and $s_3 \notin C(v)$ so u is not a descendant of v. This leads to a contradiction. Hence, no such tree T can exist. □

Theorem 3.28. *Given a finite set $\mathcal{X}$ and a finite class $\mathcal{F} = \{T_1, \ldots, T_s\}$ where each T_i is a hierarchical clustering over $\mathcal{X}$. Then*

$$\text{VC-Dim}(\mathcal{F}) \leq g(s)$$

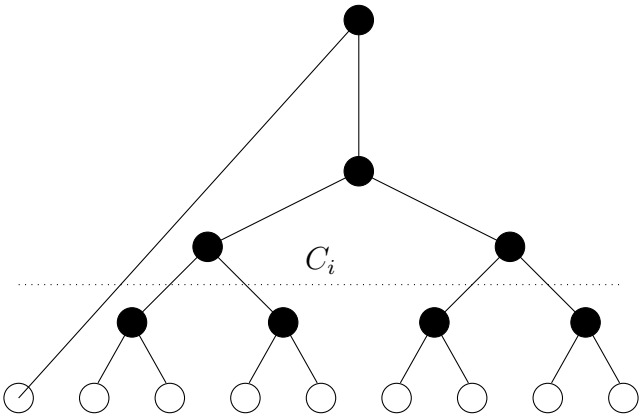


Figure 3.3: A hierarchical clustering tree of $n = 9$ points. This tree contains the clustering C_i described in the proof of Lemma 3.27.

where $g(s)$ *is the smallest integer* n *such that* $\frac{\sqrt{n}!}{\lfloor\sqrt{n}/2\rfloor!\ \ 2^{\lfloor\sqrt{n}/2\rfloor}} \geq s$

Proof. Let n be as defined in the statement of the theorem. Let $M^2 \subseteq \mathcal{X}^2$ be a set of size $> n^2$. Define $M := \{x : (x, y) \in M^2 \text{ or } (y, x) \in M^2\}$. We know that $|M| > n$. Using lemma 3.27, there exists a set of clusterings $\mathcal{C} = \{C_1, \ldots, C_{s'}\}$ of size $s' > \frac{n!}{\lfloor n/2\rfloor!\ \ 2^{\lfloor n/2\rfloor}} \geq s$ such that each $T_i \in \mathcal{F}$ contains at most one $C_j \in \mathcal{C}$. Thus, there exists a clustering C_j which is not captured by any $T_i \in \mathcal{F}$. Hence, $l_{\mathcal{F}}$ can't shatter any set of size $> n^2$. □

Using standard techniques it is easy to see that the VC-Dim of a list of hierarchical clustering trees is in $o(\log^2 s)$.

3.7 Computation complexity of RCC for common clustering classes

Alg. 5 described the general ERM approach to find the 'best' clustering for any class $\mathcal{F}$ of clusterings. Consider the case when $\mathcal{F} = \{C_1, \ldots, C_s\}$ (a finite list of clusterings). Then to implement Alg. 5, we first sample a set $S \subseteq X^2$ and then compute the error of each clustering on this set S. Computing the error of a clustering on S takes $\Theta(|S|)$ time. Hence, for finite $\mathcal{F}$, the ERM approach can be implemented in time $\Theta(|\mathcal{F}||S|)$.

Now, let's focus on the case when $\mathcal{F} = \{T_1, \ldots, T_s\}$ (a finite set of clustering trees). Each tree T_i can contain exponentially many clusterings. Hence, it is not clear if we can still

Algorithm 6: ERM approach for a hierarchical clustering tree

Input: A set X, a set $S \subseteq X^2$ labelled according to C^*. Given a clustering tree T on $S_u = \{x : (x, y) \text{ or } (y, x) \in S\}$.

Output: A clustering $\hat{C} \in T$ which implements ERM over T.

Initialize $e(\nu) = a(\nu) = 0$ for all the leaf nodes ν.
for *all non-leaf nodes* ν *(in a bottom-up manner)* **do**
    Let ν_l be left sub-tree and ν_r the right sub-tree.
    Initialize $s_a = d_a = 0$.
    for $(x, y) \in S$ **do**
        if $x, y \in nl(\nu)$ *and* $C^*(x, y) = 0$ **then**
            $d_a = d_a + 1$.
        end
        if $!(x \in nl(\nu_l)$ *and* $y \in nl(\nu_r))$ *and* $!(y \in nl(\nu_l)$ *and* $x \in nl(\nu_r))$ **then**
            continue
        end
        if $C^*(x, y) = 1$ **then**
            $s_a = s_a + 1$.
        end
    end
    $a(\nu) = d_a$
    $e(\nu) = \min\{e(\nu_l) + e(\nu_r) + s_a, a(\nu)\}$
end

implement the ERM approach in polynomial-time. Consider the problem of implementing the ERM approach when $\mathcal{F} = \{T\}$. The goal is to find the pruning in the tree which minimizes the loss $\hat{L}$. Given tree T, the clustering with the smallest loss is either the clustering which assigns all the points to a single cluster or the clustering with the best loss in T_l (left subtree) concatenated with the clustering with the best loss in T_r (right subtree). Before we describe our approach lets introduce a bit of notation.

For every node ν in the hierarchical clustering tree, let $nl(\nu)$ be the leaves which are descendants of ν. For a leaf node ν, $nl(\nu) = \{\nu\}$. Let $\mu_1 = \mu|S_-|$ and $\mu_2 = (1 - \mu)|S_+|$. Let

$$e(\nu) = \arg\min_{C \in T_\nu} \mu_1 |S_+| \hat{P}(C) + \mu_2 |S_-| \hat{N}(C)$$

where T_ν is the tree T restricted to the descendants of ν. Let $a(\nu) = \mu_1 |S_+| \hat{P}(C) +$

	simulated		publications		products I		products II		restaurants	
Clustering	true loss	estimated loss	true loss	estimated loss	true loss	estimated loss	true loss	estimated loss	true loss	estimated loss
ArtPt	0.091	0.105	0.023	0.005	0.206	0.170	0.153	0.160	0.094	0.110
Star	0.052	0.060	0.100	0.050	0.207	0.190	0.231	0.170	0.041	0.045
ApproxCorr	NA[2]	NA	0.180	0.145	0.380	0.310	0.373	0.340	0.094	0.065
Markov	0.011	**0.000**	0.017	0.010	0.159	0.130	0.125	0.085	0.045	0.030
NaïveDedup	0.397	0.365	0.497	0.495	0.413	0.405	0.394	0.380	0.094	0.080
C1 (single)	0.019	0.025	0.016	0.018	0.150	0.110	0.131	0.120	0.022	**0.015**
C2 (complete)	0.005	0.005	0.009	0.009	0.150	0.130	0.135	0.065	0.034	0.040
C3 (weighted)	0.002	**0.000**	**0.005**	**0.006**	**0.110**	0.110	0.107	0.070	**0.019**	0.020
C4 (average)	**0.001**	**0.000**	0.007	0.017	0.120	**0.100**	**0.099**	**0.060**	**0.019**	0.020
Mean loss difference		0.016		0.014		0.027		0.035		0.010

Table 3.1: True loss and the loss estimated by our framework.

		25, 25 samples		100, 100 samples		500, 500 samples	
Clustering	true loss	# queries	estimated loss	# queries	estimated loss	# queries	estimated loss
C1 (single)	**0.06107**	51	0.06	204	0.025	1023	0.024
C2 (complete)	**0.04177**	50	0.02	210	0.005	1024	0.016
C3 (weighted)	**0.03831**	50	0.02	203	0.015	1027	0.016
C4 (average)	**0.03489**	52	0.02	207	0.020	1043	0.013

Table 3.2: Simulated dataset: Impact of number of samples on the loss of the clustering

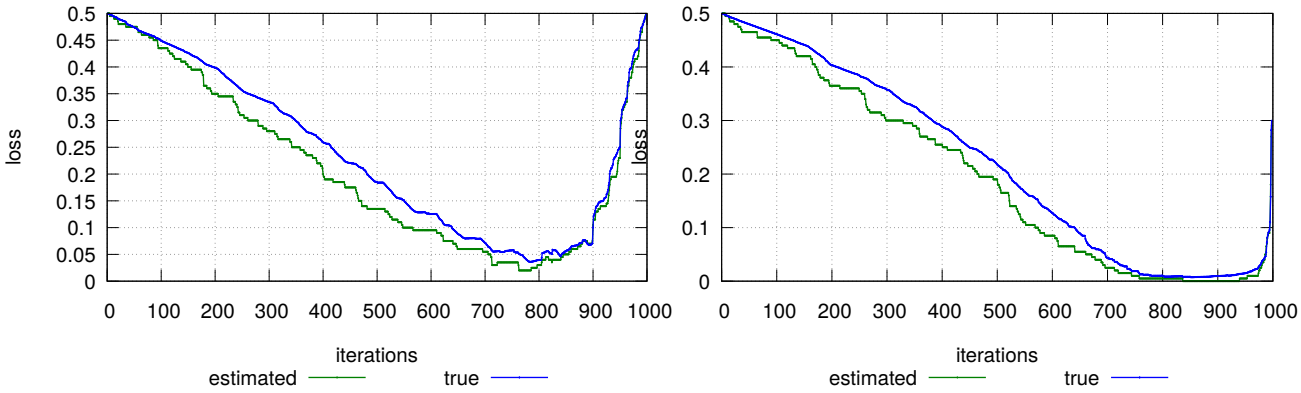
$\mu_2|S_-|\hat{N}(C)$ where C is the clustering which assigns all the descendants of ν to a single cluster. We are now ready to describe our bottom-up approach in Alg. 6. Its easy to see that the running time of the approach is $\Theta(|S||T|)$.

3.8 Experimental evaluation

We now present the evaluation of our framework on a simulated and four real world datasets. In Section 3.8.2 we show that our framework is generic and can be used to choose amongst many of the classes of algorithms for de-duplication. We also show that our framework can always choose a clustering which is close to the best clustering (algorithm) from a given class of clustering (algorithms) and our estimated loss for each of the clustering is very close to the true loss of these clustering algorithms. In Section 3.8.3 we show that our framework is robust to upto 10% of oracle mistakes, which far exceeds the intended settings dealing with human experts. Finally, in Section 3.8.4 we show that in our framework a relatively small number of samples are enough to accurately estimate the loss of a clustering.

		25, 25 samples		100, 100 samples		500, 500 samples	
Clustering	true loss	# queries	estimated loss	# queries	estimated loss	# queries	estimated loss
C1 (single)	**0.11075**	51	0.08	208	0.055	1031	0.041
C2 (complete)	**0.37172**	50	0.34	204	0.315	1035	0.334
C3 (weighted)	**0.29622**	51	0.14	203	0.260	1037	0.239
C4 (average)	**0.26877**	50	0.20	204	0.195	1027	0.202

Table 3.3: Publications dataset: Impact of number of samples on the loss of the clustering



(a) Single linkage

(b) Complete linkage

Figure 3.4: Simulated dataset: Loss reported for every iteration of hierarchical clustering

3.8.1 Evaluation setup

Algorithms In our evaluation we use four graph based clustering algorithms: (1) Articulation point clustering (*ArtPt*) [Cormen et al., 2009], (2) [Aslam et al., 2006] Star clustering (*Star*), (3) Approximate correlation clustering (*ApproxCorr*) [Bansal et al., 2004], (4) Markov clustering (*Markov*) [van Dongen, 2000]. These graph based algorithms have been used for de-duplication problems as shown in previous work [Hassanzadeh et al., 2009]. Hierarchical clustering algorithms are very effective and have been widely used to perform de-duplication. We consider 4 different linkage methods for hierarchical clustering: single linkage (C1), complete linkage (C2), weighted linkage (C3), and average linkage (C4). In addition to this we also implemented a heuristic based de-duplication algorithm (*NaïveDedup*) where any two data points are considered similar if their distance is below a certain threshold. The output of this algorithm is pairs of data points which are marked similar.

Datasets For our evaluation we use five datasets. First dataset is a simulated dataset

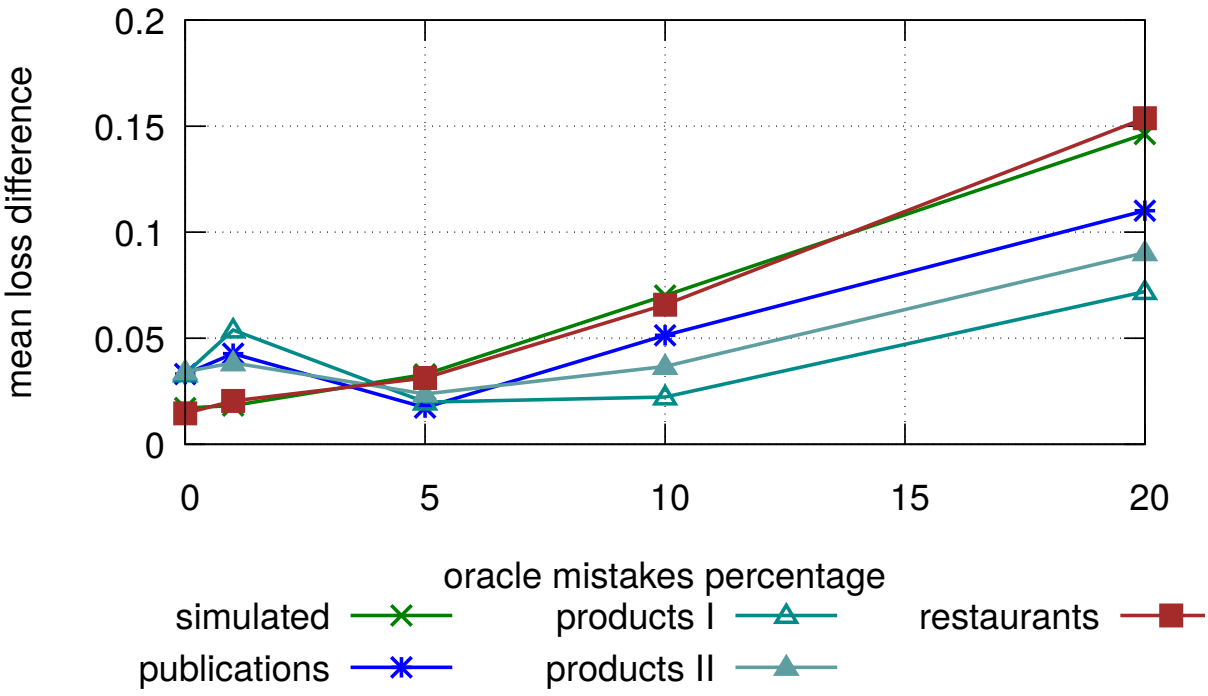
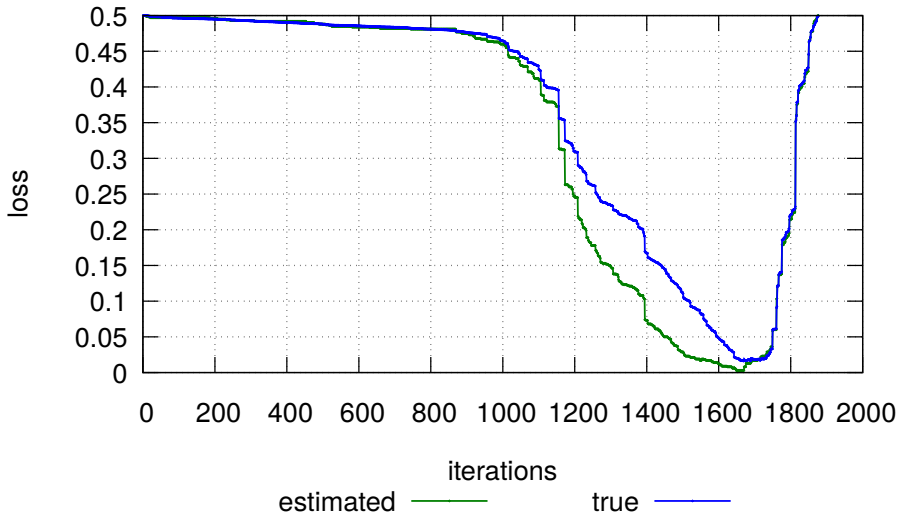
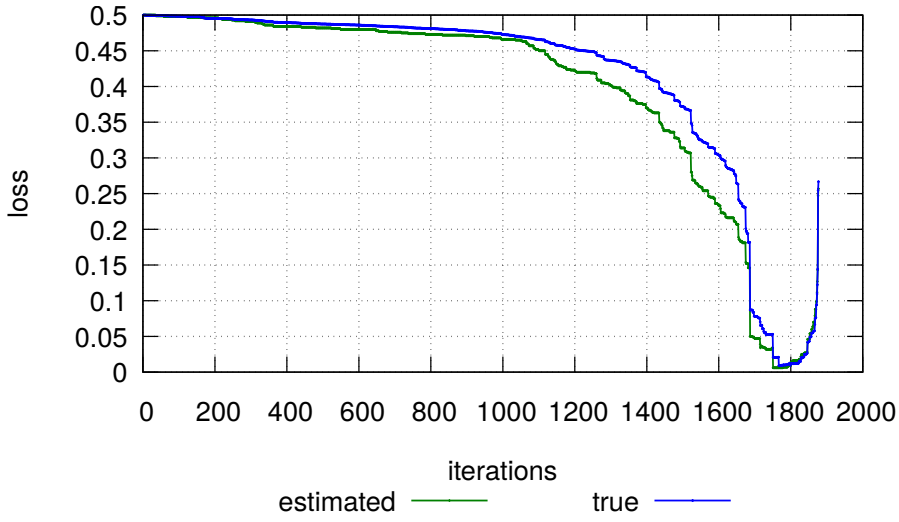


Figure 3.5: Impact of oracle mistakes

of ten thousand strings of length 20 where we simulate a clustering over the set of strings and use it as our ground truth. We use Jaro distance [Jaro, 1980] as the distance metric for strings. To simulate a clustering we generate some seed strings and then for each seed string we generate multiple secondary strings by slightly editing the seed string. Each cluster of strings resembles a single entity. Second dataset is a real-world bibliographical information of scientific publications [pub,]. The dataset has 1879 publication records with duplicates. The ground truth of duplicates is available. To perform clustering on this dataset we first tokenized each publication record and extracted 3-grams from them. Then, on 3-grams we used Jaccard distance to define distance between two records. Next two datasets are lists of E-commerce products: First dataset contains 1,363 products from Amazon, and 3,226 products from Google, and the ground truth has 1,300 matching products. Second dataset contains 1,082 products from Abt, and 1,093 products from Buy, and the ground truth has 1,098 matching products. Both these products datasets are publicly available at [pro,]. The fifth dataset is a list of 864 restaurants from the Fodor's and Zagat's restaurant guides that contains 112 duplicates. This dataset is also publicly available at [res,]. To perform



(a) Single linkage



(b) Complete linkage

Figure 3.6: Publications dataset: Loss reported for every iteration of hierarchical clustering

clustering on the products and restaurants datasets we normalized the records (product or restaurant description) using standard techniques from natural language processing, namely; denoising text, word tokenization, normalization, and stemming and lemmatiza-

tion. Given a record, this process gives us a list of word tokens. For each token, we first obtained a vector representation of the word using *Global Vectors* for word representations (GloVe [Pennington et al., 2014]). We averaged this representation across word tokens to obtain the representation of a single record. We use cosine similarity to define the distance between two records. For the simulated and publications datasets, our distance metric was Jaccard and hence we use the MinHash [Broder et al., 2000] as the hashing scheme. For the rest of the datasets, we used SimHash [Charikar, 2002] as the hashing scheme. For all the datasets we use ground truth as the oracle that can answer *same-cluster queries*. To calculate the true loss of a clustering (i.e. $L_{C^*}(C)$) we access all of the ground truth. Our framework uses only a sample of the ground truth to estimate the loss of a clustering. To judge the performance of our framework we compare the estimated loss $\hat{L}_{C^*}(C)$ against the true loss.

3.8.2 Clustering selection

In this experiment we demonstrate that our framework is generic and can be used to choose the best clustering algorithm amongst any of the classes of algorithms for de-duplication. We used our framework on all the algorithms mentioned in Section 3.8.1. The results on five datasets are summarized in Table 3.1. For each dataset we report the loss of the true-best clustering ($L_{C^*}(C)$) and the estimated loss of the best clustering selected by our framework $\hat{L}_{C^*}(C)$. This experiment highlights two main features of our framework: (i) our framework can always choose a clustering close to the best clustering algorithm from a given class of clustering algorithms using only a small number of samples, which is 200 (100 positive samples, and 100 negative samples) for all datasets and all the algorithms in Table 3.1. (ii) Our estimated loss for each clustering is very close to the true loss of these clustering algorithms. At the bottom of the table we report the mean loss difference between estimated loss and true loss computed over all the algorithms.

We would also like to emphasize that in our framework we sample only once for each dataset and use that sample to estimate the loss of all the clusterings. In Figure 3.4 and 3.6 we show that our sample can very closely estimate the loss of every clustering generated at each iteration of the hierarchical clustering. Similarly, for Table 3.1 we sampled only once for each dataset and evaluated all the clusterings generated by each algorithm. Note that, each of the graph based algorithms have a hyper-parameter, i.e. the threshold on the edge weights. Edges with weights above this threshold represent dissimilar items and are pruned from the graph. For each of the graph based clustering algorithm we applied our framework on multiple values of the hyper-parameter and report only the ones with least

true loss. However, for every choice of the hyper-parameter we observed that the estimated loss was very close to the true loss.

3.8.3 Effect of oracle mistakes

In this experiment we show that our framework is effective in real-world scenarios where the oracle may not be perfect and can make mistakes. Whenever the oracle classifies a similar pair as dissimilar or a dissimilar pair as similar we count it as a mistake. In our datasets we artificially introduce such mistakes and vary their ratio from from 0%, to 20%. In Figure 3.5 we show that our framework can closely estimate the clustering loss up to 10% of oracle mistakes, which, in real-world far exceeds the intended settings dealing with human experts. The Y-axis in Figure 3.5 reports the mean difference between true loss and estimated loss over all the clusterings selected in Table 3.1.

3.8.4 Impact of sample size

In this experiment we show that even a small number of samples are enough to estimate the true loss ($L_{C^*}(C)$). We consider four different clusterings, each one picked at random from the four hierarchical clustering methods (C1 - C4). Table 3.2 and 3.3 reports the loss for simulated and publications dataset, respectively. For each dataset we increased the number of positive and negative samples and measured the loss. The table also shows the true loss of the clustering. It can be seen that the estimated loss calculated by our framework is close to the true loss even with 25 positive samples and 25 negative samples. In addition to this, the loss does not change much by increasing the number of samples. Which means that there is no incentive to sample more. We also show that the number of queries performed by our framework are close to the sample size (as claimed in Thm. B.5), which are orders of magnitude less than $O(|X|^2)$. For example, in the simulated dataset and single linkage clustering (C1) with 25 positive and 25 negative samples our framework performed 51 queries, that means only one query was wasted. Similarly, 4 queries were wasted for 100 positive and 100 negative samples, and so on.

Chapter 4

Finding cluster structure amidst background noise

Clustering is an umbrella term for a wide variety of unsupervised data processing techniques. A relatively comprehensive description of clustering is that it aims to group together data instances that are similar, while separating dissimilar objects. Most of the common clustering tools output a partitioning of the input data into groups, clusters, that share some form of cohesiveness or between-cluster separation requirement[1]. However, in many cases, real data sets, in particular large ones, have on top of such cohesive separated groups, a significant amount of "background" unstructured data. Clustering in such situations is the focus of this work. Maybe surprisingly, this topic has received relatively little attention in the clustering research community, and even less so when it comes to theoretical work.

The discussion of finding clustering structure in data sets that also contain subsets that do not conform well to that structure usually falls under the terminology of noise robustness (see for example, [Balcan and Liang, 2012],[Ackerman and Ben-David, 2009],[Dave, 1993], [Cuesta-Albertos et al., 1997],[García-Escudero et al., 2008]). However, noise robustness, at least in that context, addresses the noisy part of the data as either generated by some specific generative model (like uniform random noise, or Gaussian perturbations) or refers to worst-case adversarially generated noisy data. In this chapter we take a different approach. What distinguishes the noise that we consider from the "clean" part of the input data is that it is *structureless*. The exact meaning of such a notion of structurlessness may

[1]The assignment to clusters can sometimes be probabilistic, and clusters may be allowed to intersect, but these aspects are orthogonal to the discussion in this chapter.

vary depending on the type of structure the clustering algorithm is aiming to detect in the data. We focus on defining structurelessness as not having significantly large dense subsets. We believe that such a notion is well suited to address "gray background" contrasting with cohesive subsets of the data that are the objects that the clustering aims to detect.

The distinction between structured and unstructured parts of the data requires, of course, a clear notion of relevant structure. For that, we resort to a relatively large body of recent work proposing notions of clusterable data sets. That work was developed mainly to address the gap between the computational hardness of (the optimization problem of) many common clustering objectives and the apparent feasibility of clustering in practical applications. We refer the reader to [Ben-David, 2015] for a survey of that body of work. Here, we focus on two such notions, one based on the α-center-proximity introduced by [Awasthi et al., 2012] and the other, λ-separation, introduced by [Ben-David and Haghtalab, 2014].

Our approach diverges from previous discussions of clusterable inputs in yet another aspect. Much of the theoretical research of clustering algorithms views clustering as an optimization problem. For some predetermined objective function (or clustering cost), the algorithm's task is to find the data partitioning that minimizes that objective. In particular, this approach is shared by all the works surveyed in [Ben-David, 2015]. However, in many practical situations the reality is different. Given a large data set to cluster, oftentimes there is no way a user may know what is the cost of the optimal clustering of that data, or how close to optimal the algorithm's outcome is. Instead, a user might have a notion of meaningful cluster structure, and will be happy with any outcome that meets such a requirement. Consequently, our algorithms aim to provide meaningful clustering solutions (where "meaningful" is defined in a way inspired by the above mentioned notions of clusterability) without reference to any particular optimization objective function. Our algorithms efficiently compute a hierarchical clustering tree that captures all such meaningful solutions. One should notice that all of those notions of clusterability (those under which it can be shown that an objective-minimizing clustering can be found efficiently) assume that there exists an optimal solution that satisfies the meaningfulness condition (such as being perturbation robust, or having significantly smaller distances of points to their own cluster centers than to other centers). Under those assumptions, an algorithm that outputs a tree capturing all meaningful solutions, allows efficient detection of the cost-optimal clustering (in fact, the algorithms of [Balcan and Liang, 2012] also yield such trees, for clean, noiseless inputs). Consequently, under the assumptions of those previous works, our algorithms yield an efficient procedure for finding such an optimal solution.

4.1 Related Work

The goal of clustering is to partition a set of objects into *dissimilar* subsets of *similar* objects. Based on the definition of similarity, the optimal solution to a clustering task is achieved by optimizing an objective function. Although solving this optimization problem is usually NP-hard, the clustering task is routinely and successfully employed in practice. This gap between theory and practice recommends characterizing the real world data sets by defining mathematical notions of *clusterable* data. As a result, provably efficient clustering algorithms can be found for these so called *nice* data.

In the past few years, there has been a line of work on defining notions of clusterability. The goal of all these methods has been to show that clustering is computationally efficient if the input X enjoys some nice structure. In [Bilu and Linial, 2012], a clustering instance is considered to be *stable* if the optimal solution to a given objective function does not change under small multiplicative perturbations of distances between the points. Using this assumption, they give an efficient algorithm to find the max-cut clustering of graphs which are resilient to $O(\sqrt{|X|})$ perturbations. Using a similar assumption, [Ackerman and Ben-David, 2009] considered additive perturbations of the underlying metric and designed an efficient algorithm that outputs a clustering with near-optimal cost.

In terms of clusterability conditions, the most relevant previous papers are those addressing clsutering under α-*center proximity* condition (see Def. A.3). Assuming that the centers belong to X (*proper* setting), [Awasthi et al., 2012] shows an efficient algorithm that outputs the optimal solution of a given center-based objective assuming that optimal solution satisfies the $(\alpha > 3)$-center proximity. This result was improved to $(\alpha = \sqrt{2}+1 \approx 2.4)$ when the objective is k-median [Balcan and Liang, 2012]. In [Ben-David and Reyzin, 2014] it was shown that unless P=NP such a result cannot be obtained for $(\alpha < 2)$-center proximal inputs.

However, as mentioned above, these results apply only to the noiseless case. Few methods have been suggested for analyzing clusterability in the presence of noise. For example, [Balcan and Liang, 2012] consider a dataset which has α-center proximity except for an ϵ fraction of the points. They give an efficient algorithm which provides a $1 + O(\epsilon)$-approximation to the cost of the k-median optimal solution when $\alpha > 2 + \sqrt{7} \approx 4.6$. Note that, while this result applies to adversarial noise as well, it only yields an approximation to the desired solution and the approximation guarantee is heavily influenced by the size of noise.

In a different line of work, [Ben-David and Haghtalab, 2014] studied the problem of robustifying any center-based clustering objective to noise. To achieve this goal, they

introduce the notion of *center separation* (look at Def. 4.7). Informally, an input has center separation when it can be covered by k well-separated set of balls. Given such an input, they propose a paradigm which converts any center-based clustering algorithm into a clustering algorithm which is robust to small amount of noise. Although this framework works for any objective-based clustering algorithm, it requires a strong restriction on the noise and clusterability of the data. For example, when the size of the noise is $\frac{5}{100}|X|$, their algorithm is able to obtain a robustified version of 2-median, only if X is covered by k unit balls which are separated with distance 10.

In this work, we consider a natural relaxation of [Balcan and Liang, 2012] and [Ben-David and Haghtalab, 2014], with the goal to capture more realistic domains containing arbitrary amount of noise, assuming that noise is *structureless* (in a precise sense defined below). For example, in [Balcan and Liang, 2012], the size of the noise $|\mathcal{N}| \leq \frac{m(C)}{8}$ (where $m(C)$ is size of the smallest cluster). Our algorithms can handle much larger amount of noise as long as they satisfy the *structureless* condition.

We define a novel notion of "gray background" noise. Informally, we call noise *structureless* if it does not have similar structure to a *nice* cluster at any part of the domain. Under that definition (look at Def. 4.6), our positive, efficient clustering results, do not depend on any restriction on the size of the noise.

Given a clusterable input X which contains *structureless* noise, we propose an efficient algorithm that outputs a hierarchical clustering tree of X that captures all *nice* clusterings of X. Our algorithm perfectly recovers the underlying *nice* clusterings of the input and its performance is independent of number of noisy points in the domain.

We complement our algorithmic results by proving that under more relaxed conditions, either on the level of clusterability of the clean part of the data, or on the unstructuredness requirements on the noise, such results become impossible.

4.1.1 Outline

The rest of this chapter is structured as follows. In Section 4.2, we present our notation and formal definitions. In Section 4.3 we show that the type of noise that we address in this paper is likely to arise under some natural assumptions on the data generating process. In Section 4.4, we present an efficient algorithm that, for any input set X which contains structureless noise, recovers all the underlying clusterings of non-noise subset of X that satisfies α-center proximity for $\alpha > 2+\sqrt{7} \approx 4.6$. We complement these results by proving that for $\alpha \leq 2\sqrt{2}+3 \approx 5.8$ in the case that we have arbitrary noise and for $\alpha \leq \sqrt{2}+3 \approx 4.4$

in the case of structureless noise, efficient discovery of all nicely structured subsets is not possible.

In Section 4.5.1, we describe an efficient algorithm that, for any input X, recovers all the underlying clusterings of X that satisfy λ-center separation for $\lambda \geq 3$. We also prove that it is NP-hard to improve this to $\lambda \leq 2$. In Section 4.5.2, we consider a similar problem in the presence of either arbitrary or structureless noise. We propose an efficient algorithm that, for any input X which contains structureless noise, recovers all the underlying clusterings of non-noise subset of X that satisfy λ-center separation for $\lambda \geq 4$. We will also show that this result is tight for the case of structureless noise. We complement our results by showing that, under arbitrary noise assumption, no similar positive result can be achieved for $\lambda \leq 6$. Note that all our missing proofs can be found in the appendix.

4.2 Notation and definition

Let $(\mathbf{M}, d)$ be a metric space. Given a data set $X \subseteq \mathbf{M}$ and an integer k. A k-clustering of X denoted by $\mathcal{C}_X$ is a partition of X into k disjoints sets. Given points $c_1, \ldots, c_k \in \mathbf{M}$, we define the clustering induced by these points (or *centers*) by assigning each $x \in X$ to its nearest center. In the *steiner* setting, the centers can be arbitrary points of the metric space $\mathbf{M}$. In the *proper* setting, we restrict our centers to be members of the data set X. In this paper, we will be working in the **proper** setting.

For any set $\mathcal{A} \subseteq X$ with center $c \in \mathbf{M}$, we define the radius of $\mathcal{A}$ as $r_c(\mathcal{A}) = \max_{x \in \mathcal{A}} d(x, c)$. Throughout the chapter, we will use the notation $\mathcal{C}_X$ to denote the clustering of the set X and $\mathcal{C}_S$ to denote the clustering of some $S \subseteq X$.

Definition 4.1 ($r(\mathcal{C}_X)$, $m(\mathcal{C}_X)$)**.** *Given a clustering $\mathcal{C}_X = \{C_1, \ldots, C_k\}$ induced by centers $c_1, \ldots, c_k \in \mathbf{M}$, we define $m(\mathcal{C}_\mathcal{X}) = \min_i |C_i|$ and $r(\mathcal{C}_\mathcal{X}) = \max_i r(C_i)$.*

Definition 4.2 ($\mathcal{C}_X$ restricted to a set)**.** *Given $S \subseteq X$ and a clustering $\mathcal{C}_X = \{C_1, \ldots, C_k\}$ of the set X. We define $\mathcal{C}_X$ restricted to the set S as $\mathcal{C}_{X|_S} = \{C_1 \cap S, \ldots, C_k \cap S\}$.*

Definition 4.3 ($\mathcal{C}_X$ respects $\mathcal{C}_S$)**.** *Given $S \subseteq X$, clusterings $\mathcal{C}_X = \{C_1, \ldots, C_k\}$ and $\mathcal{C}_S = \{S_1, \ldots, S_{k'}\}$. We say that $\mathcal{C}_X$ respects $\mathcal{C}_S$ if $\mathcal{C}_{X|_S} = \mathcal{C}_S$.*

Definition 4.4 ($\mathcal{T}$ or $\mathcal{L}$ captures $\mathcal{C}_S$)**.** *Given a hierarchical clustering tree $\mathcal{T}$ of X and a clustering $\mathcal{C}_S$ of $S \subseteq X$. We say that $\mathcal{T}$ captures $\mathcal{C}_S$ if there exists a pruning $\mathcal{P}$ which respects $\mathcal{C}_S$.*

Similarly, given a list of clusterings $\mathcal{L}$ of X and a clustering $\mathcal{C}_S$ of $S \subseteq X$. We say that $\mathcal{L}$ captures $\mathcal{C}_S$ if there exists a clustering $\mathcal{C}_X \in \mathcal{L}$ which respects $\mathcal{C}_S$.

Definition 4.5 (α-center proximity [Awasthi et al., 2012]). *A clustering $\mathcal{C}_X = \{C_1, \ldots, C_k\}$ satisfies α-center proximity w.r.t. X and k if there exist centers $c_1, \ldots, c_k \in \mathbf{M}$ such that the following holds. For all $x \in C_i$ and $i \neq j$, $\alpha d(x, c_i) < d(x, c_j)$*

Next, we formally define our notion of structureless noise. Roughly, such noise should be scattered sparsely, namely, there should be no significant amount of noise in any small enough ball. Note that such a restriction does not impose any upper bound on the number of noise points.

Definition 4.6 ((α, η)-center proximity). *Given $S \subseteq X$, a clustering $\mathcal{C}_S = \{S_1, \ldots, S_k\}$ has (α, η)-center proximity w.r.t. X, S and k if there exists centers $s_1, \ldots, s_k \in \mathbf{M}$ such that the following holds.*

- $\diamond$ **α-center proximity**: *For all $x \in S_i$ and $i \neq j$, $\alpha d(x, s_i) < d(x, s_j)$*
- $\diamond$ **η-sparse noise**: *For any ball B, $r(B) \leq \eta\, r(\mathcal{C}_S) \implies |B \cap (X \setminus S)| < \frac{m(\mathcal{C}_S)}{2}$*

Definition 4.7 (λ-center separation [Ben-David and Haghtalab, 2014]). *A clustering $\mathcal{C}_X = \{C_1, \ldots, C_k\}$ has λ-center separation w.r.t. X and k if there exists centers $c_1, \ldots, c_k \in \mathbf{M}$ such that $\mathcal{C}_X$ is the clustering induced by these centers and the following holds. For all $i \neq j$, $d(c_i, c_j) > \lambda r(\mathcal{C}_X)$*

Definition 4.8 ((λ, η)-center separation). *Given $S \subseteq X$, a clustering $\mathcal{C}_S$ has (λ, η)-center separation w.r.t. X, S and k if there exists centers $s_1, \ldots, s_k \in \mathbf{M}$ such that $\mathcal{C}_X$ is the clustering induced by these centers and the following holds.*

- $\diamond$ **λ-center separation**: *For all $i \neq j$, $d(s_i, s_j) > \lambda r(\mathcal{C}_S)$*
- $\diamond$ **η-sparse noise**: *For any ball B, $r(B) \leq \eta\, r(\mathcal{C}_S) \implies |B \cap (X \setminus S)| < \frac{m(\mathcal{C}_S)}{2}$*

We denote a ball of radius r at center c by $B(c, r)$. We denote by $P_i(c)$ a collection of i many points sitting on the same location c. If the location is clear from the context, we will use the notation P_i.

4.3 Justification of sparse noise

In this section, we examine our sparseness condition. We will show that if the set of points $\mathcal{N}$ are generated by a non concentrated distribution in a ball in $\mathbf{R}^d$ then with high probability, as long as $\mathcal{N}$ is not too large (so as to" drown" the original data set), it will satisfy the sparse noise condition. The proof is based on the epsilon approximation theorem for classes of finite VC-dimension, applied to the set of balls in $\mathbf{R}^d$. The following, rather natural, definition of non concentrated distribution was introduced in [Balcan et al., 2012].

Definition 4.9. *A distribution over the d-dimensional unit ball is non-concentrated if, for some constant c, the probability density of any point x is at most c times its density under the uniform distribution over that ball.*

Theorem 4.10 (Noise by non concentrated distribution is sparse)**.** *Let X be a ball of radius R in $\mathbf{R}^d$ and $S \subseteq X$. Let $\mathcal{C}$ be a clustering of S which satisfies α-center proximity (or λ-center separation). Given parameters $\epsilon, \delta \in (0,1)$. Let $\mathcal{N} \subseteq X$ be picked i.i.d. according to a non concentrated probability distribution. If $|\mathcal{N}| < c\left(\frac{R}{r(C)\eta}\right)^d m(C)$ then with high probability, $S \cup \mathcal{N}$ satisfies (α, η)-center proximity (the (λ, η)-center separation, respectively).*

Proof. Let $H = \{B \text{ is a ball } : B \subseteq X\}$. Observe that VC-Dim$(H) = d+1$. Let $\gamma := \frac{r(C)}{R}$. Since the noise-generating distribution P is c-concentrarted, for every ball B, $P(B) \leq c\frac{\text{vol}(B)}{\text{vol}(X)} = c\gamma^d$. Now, the fundamental ϵ-approximation theorem (Theorem 1.1) establishes the result. □

Note that Theorem 4.10 shows that the cardinality of the noise set, $|\mathcal{N}|$, can be much bigger than the size of the smallest cluster $m(\mathcal{C})$.

4.4 Center Proximity

In this section, we study the problem of recovering (α, η)-center proximal clusterings of a set X, in the presence of noise. The goal of our algorithm is to produce an efficient representation (hierarchical clustering tree) of all possible (α, η)-center proximal nice clusterings rather than to output a single clustering or to optimize an objective function. Here is a more precise overview of the results of this section:

- *Positive result under sparse noise* - In Section 4.4.1, we give our main result under sparse noise. If $\alpha \geq 2 + \sqrt{7} \approx 4.6$ and $\eta \geq 1$; for any value of t, Alg. 7 outputs a tree which captures all clusterings $\mathcal{C}^*$ (of a subset of X) which satisfy (α, η)-center proximity and $m(C^*) = t$.
- *Lower bound under sparse noise* - In Section 4.4.2, we show that if $\alpha \leq 2 + \sqrt{3} \approx 3.7$ and $\eta \leq 1$ then there is no tree and no list of 'small' size ($< 2^{k/2}$) which can capture all clusterings $\mathcal{C}$ (of a subset of X) which satisfy (α, η)-center proximity even for a fixed value of the size of the smallest cluster ($m(C) = t$).

- *Lower bound with arbitrary noise* - In Section 4.4.3, we show that for a given value of a parameter t, if $\alpha \leq 2\sqrt{2}+3 \approx 5.8$ and the number of noisy points exceeds $\frac{3}{2}t$ then no tree can capture all clusterings $\mathcal{C}$ (of a subset of X) which satisfy α-center proximity even for fixed $m(\mathcal{C}) = t$. Identical result holds for 'small' ($< 2^{k/2}$) lists if the number of noisy points exceeds $\frac{3k}{2}t$.

4.4.1 Positive result under sparse noise

Given a clustering instance (X, d) and a parameter t, we introduce an efficient algorithm which outputs a hierarchical clustering tree $\mathcal{T}$ of X with the following property. For every k, for every $S \subseteq X$ and for every k-clustering $\mathcal{C}_S$ which satisfies (α, η)-center proximity (for $\alpha \geq 2+\sqrt{7}$ and $\eta \geq 1$) and $m(\mathcal{C}_S) = t$, $\mathcal{T}$ captures $\mathcal{C}_S$. It is important to note that our algorithm only knows X and has no knowledge of the set S.

Our algorithm has a linkage based structure similar to [Balcan and Liang, 2012]. However, our method benefits from a novel *sparse distance condition*. We introduce the algorithm in Alg. 7 and prove its efficiency and correctness in Theorem 4.13 and Theorem 4.12 respectively.

Definition 4.11 (Sparse distance condition)**.** *Given a clustering $\mathcal{C} = \{C_1, \ldots, C_k\}$ of the set X and a parameter t. Let $B = B(p, d(p,q))$ be a ball centered at $p \in X$ and of radius $d(p,q)$ where $q \in X$. We say that the ball B satisfies the sparse distance condition w.r.t. clustering $\mathcal{C}$ when the following holds.*

- $|B| \geq t$.
- *For any $C_i \in \mathcal{C}$, if $C_i \cap B \neq \emptyset$, then $C_i \subseteq B$ or $|B \cap C_i| \geq t/2$.*

Intuitively, Alg. 7 works as follows. It maintains a clustering $\mathcal{C}^{(l)}$, which is initialized so that each point is in its own cluster. It then goes over all pairs of points p, q in increasing order of their distance $d(p,q)$. If $B(p, d(p,q))$ satisfies the sparse distance condition w.r.t. $\mathcal{C}^{(l)}$, then it merges all the clusters which intersect with this ball into a single cluster and updates $\mathcal{C}^{(l)}$. Furthermore, the algorithm builds a tree with the nodes corresponding to the merges performed so far. We will show that for all $S \subseteq X$ which are (α, η)-proximal t-min nice and for all clusterings $\mathcal{C}_S$ which have (α, η)-center proximity, Alg. 7 outputs a tree which captures $\mathcal{C}_S$.

Theorem 4.12. *Given a clustering instance (X, d) and a parameter t. Alg. 7 outputs a tree $\mathcal{T}$ with the following property. For all k, $S \subseteq X$ and for all k-clusterings $\mathcal{C}^*_S = \{S^*_1, \ldots, S^*_k\}$ which satisfy $(2+\sqrt{7}, 1)$-center proximity the following holds. If $m(\mathcal{C}^*_S) = t$ then $\mathcal{T}$ captures $\mathcal{C}_S$.*

Algorithm 7: Alg. for (α, η)-center proximity with parameter t

Input: (X, d) and t
Output: A hierarchical clustering tree T of X.

1 Let $\mathcal{C}^{(l)}$ denote the clustering X after l merge steps have been performed. Initialize $\mathcal{C}^{(0)}$ so that all points are in their own cluster. That is, $\mathcal{C}^{(0)} = \{\{x\} : x \in X\}$.
2 Go over all pairs of points p, q in increasing order of the distance $d(p, q)$. If $B = B(p, d(p, q))$ satisfies the sparse distance condition then
3 Merge all the clusters which intersect with B into a single cluster.
4 Output clustering tree T. The leaves of T are the points in dataset X. The internal nodes correspond to the merges performed.

Proof. Fix any $\mathcal{S} \subseteq \mathcal{X}$. Let $\mathcal{C}^*_S = \{S^*_1, \ldots, S^*_k\}$ be a clustering of $\mathcal{S}$ such that $m(\mathcal{C}^*_S) = t$ and $\mathcal{C}^*_S$ has (α, η)-center proximity. Denote by $r_i := r(S^*_i)$ and $r = \max r_i$. Define $Y^{\mathcal{C}}_B := \{C_i \in \mathcal{C} : C_i \subseteq B \text{ or } |B \cap C_i| \geq t/2\}$. Note that whenever a ball B satisfies the sparse-distance condition, all the clusters in $Y^{\mathcal{C}^{(l)}}_B$ are merged together and the clustering $\mathcal{C}^{(l+1)}$ is updated. We will prove the theorem by proving two key facts.

F.1 If the algorithm merges points from a good cluster S^*_i with points from some other good cluster, then at this step the distance being considered $d = d(p, q) > r_i$.

F.2 When the algorithm considers the distance $d = r_i$, it merges all points from S^*_i (and possibly points from $\mathcal{X} \setminus \mathcal{S}$) into a single cluster C_i. Hence, there exists a node in the tree N_i which contains all the points from S^*_i and no points from any other good cluster S^*_j.

Note that the theorem follows from these two facts. Similar reasoning was also used in proof of Lemma 3 in [Balcan and Liang, 2012]. We now prove both of these facts formally.

Proof of Fact. ***F.1*** Let $\mathcal{C}^{(l)} = \{C_1, \ldots, C_{k'}\}$ be the current clustering of $\mathcal{X}$. Let $l + 1$ be the first merge step which merges points from the good cluster S^*_i with points from some other good cluster. Let $p, q \in \mathcal{X}$ be the pair of points being considered at this step and $B = B(p, d(p, q))$ the ball that satisfies the sparse distance condition at this merge step. Denote by $Y = Y^{C^{(l)}}_B$. We need to show that $d(p, q) > r_i$. To prove this, we need Claim 1 below.

Claim 1. *Let $p, q \in \mathcal{X}$ and B, Y, S^*_i and $C^{(l)}$ be as defined above. If $d(p, q) \leq r$, then $B \cap S^*_i \neq \emptyset$ and there exists $n \neq i$ such that $B \cap S^*_n \neq \emptyset$.*

$l+1$ is the first step which merges points from S_i^* with some other good cluster. Hence, $\exists C_i \in Y$ such that $C_i \cap S_i^* \neq \emptyset$ and $\forall n \neq i$, $C_i \cap S_n^* = \emptyset$. Also, $\exists C_j \in Y$ such that $C_j \cap S_j^* \neq \emptyset$ for some S_j^* and $C_j \cap S_i^* = \emptyset$.

$C_i \in Y$. Hence, $C_i \subseteq B$ or $|C_i \cap B| \geq t/2$. The former is trivial. In the latter, for the sake of contradiction, assume that B contains no points from S_i^*. This implies that $B \cap C_i \subseteq B \cap \{\mathcal{X} \setminus \mathcal{S}\}$ and $|B \cap \{\mathcal{X} \setminus \mathcal{S}\}| \geq t/2$. This is a contradiction. The case when $C_j \in Y$ is identical. □

Claim 2. *Let the framework be as given in Claim 1. Then, $d(p,q) > r_i$.*

If $d(p,q) > r$, then the claim follows trivially. In the other case, from Claim 1, B contains $p_i \in S_i^*$ and $p_j \in S_j^*$. Let $r_i = d(c_i, q_i)$ for some $q_i \in S_i^*$.

$d(c_i,q_i) < \frac{1}{\alpha} d(q_i,c_j) < \frac{1}{\alpha}[\frac{1}{\alpha} d(p_i,p_j) + \frac{1}{\alpha} d(c_i,q_i) + d(p_i,p_j) + 2d(c_i,q_i)]$ This implies that $(\alpha^2 - 2\alpha - 1) d(q_i,c_i) < (\alpha+1) d(p_i,p_j)$. For $\alpha \geq 2 + \sqrt{7}$, this implies that $d(c_i,q_i) < d(p_i,p_j)/2$ which implies $d(c_i,q_i) < d(p,q)$. [Balcan and Liang, 2012] also stated this result. □

*Proof of Fact **F.2*** Let $\mathcal{C}^{(l)} = \{C_1, \ldots, C_{k'}\}$ be the current clustering of $\mathcal{X}$. Let $l+1$ be the merge step when $p = s_i$ and $q = q_i$ such that $d(s_i, q_i) = r_i$. We will prove that the ball $B = B(s_i, q_i)$ satisfies the sparse-distance condition.

Claim 3. *Let s_i, q_i, r_i, B and Y be as defined above. Then, B satisfies the sparse distance condition and for all $C \in Y$, for all $j \neq i, C \cap S_j^* = \emptyset$.*

$|B| = |S_i^*| \geq t$. Observe that, for all $C \in \mathcal{C}^{(l)}$, $|C| = 1$ or $|C| \geq t$.

- Case 1. $|C| = 1$. If $C \cap B \neq \emptyset \implies C \subseteq B = S_i^*$.
- Case 2. $|C| \geq t$. $C \cap B \neq \emptyset$. Let $h(C)$ denote the height of the cluster in the tree T.
  - Case 2.1. $h(C) = 1$. In this case, there exists a ball B' such that $B' = C$. We know that $r(B') \leq r_i \leq r$. Hence using Claim 2, we get that for all $j \neq i$, $B' \cap S_j^* = \emptyset$. Thus, $|B' \setminus S_i^*| \leq t/2 \implies |B \cap C| = |C| - |C \setminus B| = |C| - |B' \setminus S_i^*| \geq t/2$. Hence, $C \in Y$.
  - Case 2.2. $h(C) > 1$. Then there exists some C' such that $h(C') = 1$ and $C' \subset C$. Now, using set inclusion and the result from the first case, we get that $|B \cap C| \geq |B \cap C'| \geq t/2$. Hence, $C \in Y$. Using Claim 2, we get that for all $j \neq i$, $C \cap S_j^* = \emptyset$.

□

Theorem 4.13. *Given clustering instance (X,d) and t. Alg. 7 runs in poly$(|X|)$.*

Proof. Let $n = |X|$. Checking if B satisfies the sparse-distance condition takes $O(n)$ time and hence the algorithm runs in $O(n^3)$ time. □

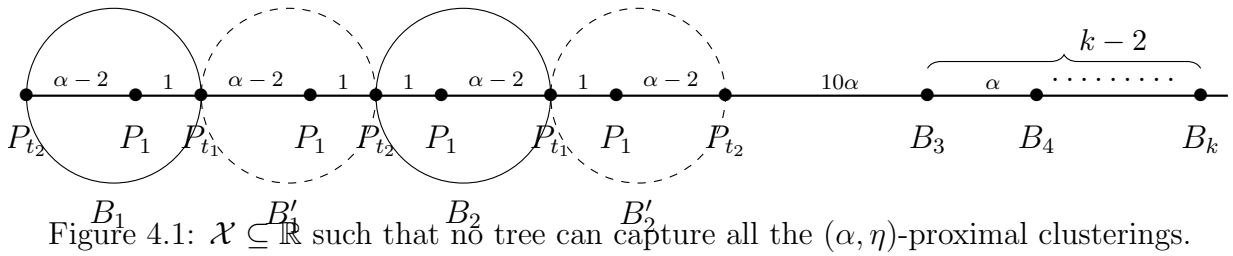


Figure 4.1: $\mathcal{X} \subseteq \mathbb{R}$ such that no tree can capture all the (α, η)-proximal clusterings.

4.4.2 Lower bound under sparse noise

Theorem 4.14. *Given the number of clusters k and parameter t. For all $\alpha \leq 2+\sqrt{3}$ and $\eta \leq 1$ there exists a clustering instance (X, d) such that any clustering tree $\mathcal{T}$ of X has the following property. There exists $S \subseteq X$ and clustering $\mathcal{C}_S$ which satisfies (α, η)-center proximity and $m(\mathcal{C}_S) = t$ but $\mathcal{T}$ doesn't capture $\mathcal{C}_S$.*

Proof. Let $\mathcal{X}, B_1, B_2, B_1', B_2'$ be as shown in Fig. 4.1. Let $t_1 = \frac{t}{2} + 1$ and $t_2 = \frac{t}{2} - 2$. For $\alpha \leq 2 + \sqrt{3}$, clusterings $\mathcal{C}_{\mathcal{S}} = \{B_1, B_2, B_3, \ldots, B_k\}$ and $\mathcal{C}_{\mathcal{S}'} = \{ B_1', B_2', B_3, \ldots, B_k\}$ satisfy $(\alpha, 1)$-center proximity and $m(\mathcal{C}_{\mathcal{S}}) = m(\mathcal{C}_{\mathcal{S}}') = t$. Now, a simple proof by contradiction shows that there doesn't exist a tree T and prunings P and P' such that P respects $\mathcal{C}_{\mathcal{S}}$ and P' respects $\mathcal{C}_{\mathcal{S}'}$. □

Theorem 4.15. *Given the number of clusters k and parameter t. For all $\alpha \leq 2+\sqrt{3}$, $\eta \leq 1$ there exists (X, d) such that any list $\mathcal{L}$ (of clusterings of X) has the following property. If $|\mathcal{L}| < 2^{\frac{k}{2}}$ then there exists $S \subseteq X$ and clustering $\mathcal{C}_S$ which satisfies (α, η)-center proximity and $m(\mathcal{C}_S) = t$ but $\mathcal{L}$ doesn't capture $\mathcal{C}_S$.*

Proof. The clustering instance $\mathcal{X}$ is an extension of Fig. 4.1. Let $G_1 = \{B_1, B_1', B_2, B_2'\}$ be the balls as in Fig. 4.1. Now, construct $G_2 = \{B_3, B_3', B_4, B_4'\}$ exactly identical to G_1 but far. In this way, we construct $k/2$ copies of G_1. □

4.4.3 Lower bound under arbitrary noise

Theorem 4.16. *Given the number of clusters k and a parameter t. For all $\alpha < 2\sqrt{2}+3$ there exists (X, d) such that any clustering tree $\mathcal{T}$ of X has the following property. There exists $S \subseteq X$ and there exists clustering $\mathcal{C}_S$ which satisfies α-center proximity such that $m(\mathcal{C}_S) = t$ and the following holds. If $|X \setminus S| \geq \frac{3m(\mathcal{C}_S)}{2} + 5$, then $\mathcal{T}$ doesn't capture $\mathcal{C}_S$.*

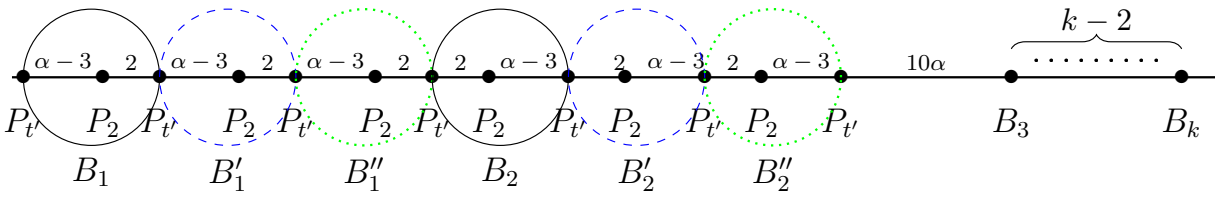


Figure 4.2: $\mathcal{X} \subseteq \mathbb{R}$ such that no algorithm can capture all the α-proximal clusterings.

Proof. Let $\mathcal{X} \subseteq \mathbb{R}$ be as shown in Fig. 4.2. Let $t' = \frac{t}{2} - 1$ and let $B_1, B_2, B_3, B_1', B_2', B_3', B_1'', B_2''$ and B_3'' be as shown in Fig. 4.2. For $\alpha \leq 2\sqrt{2} + 3$, clusterings $\mathcal{C}_S = \{B_1, B_2, B_3, \ldots, B_k\}$, $\mathcal{C}_{S'} = \{ B_1', B_2', B_3, \ldots, B_k\}$ and $\mathcal{C}_S'' = \{ B_1'', B_2'', B_3, \ldots, B_k\}$ satisfy $(\alpha, 1)$-center proximity. Also, $m(\mathcal{C}_S) = m(\mathcal{C}_S') = m(\mathcal{C}_S'') = t$. Arguing similarly as in Theorem 4.14 completes the proof. □

Theorem 4.17. *Given the number of clusters k and parameter t. For all $\alpha \leq 2\sqrt{2} + 3$ there exists (X, d) such that any list $\mathcal{L}$ (of clusterings of X) has the following property. There exists $S \subseteq X$ and there exists clustering $\mathcal{C}_S$ which satisfies α-center proximity such that $m(\mathcal{C}_S) = t$ and the following holds. If $|\mathcal{L}| < 2^{\frac{k}{2}}$ and $|X \setminus S| \geq \frac{k}{2}(\frac{3m(\mathcal{C}_S)}{2} + 5)$, then $\mathcal{L}$ doesn't capture $\mathcal{C}_S$.*

Proof. To prove the lower bound in the list model, instance constructed in Theorem 4.15 is a simple extension of the instance in Theorem 4.14. The instance for the proof of Theorems 4.17 is similarly constructed as extension of the corresponding tree lower bound instance (Theorems 4.16). □

4.5 Center Separation

4.5.1 Center Separation without noise

In this section, we study the problem of recovering λ-center separated clusterings of a set X, in the absence of noise. We do not want to output a single clustering but to produce an efficient representation (hierarchal clustering tree) of all possible λ-center separated nice clusterings. In Section 4.5.1 we give an algorithm that generates a tree of all possible λ-center separated clusterings of X for $\lambda > 3$. In Section 2, we prove that for $\lambda < 2$, it is NP-hard to find any such clustering.

Algorithm 8: Alg. for λ-center separation

Input: (X, d)
Output: A hierarchical clustering tree T of X.

1 Initialize the clustering so that each point is in its own cluster.
2 Run single-linkage till only a single cluster remains. Output clustering tree T.

Positive result under no noise

Given a clustering instance (X, d), our goal is to output a hierarchical clustering tree T of X which has the following property. For every k and for every k-clustering $\mathcal{C}_X$ which satisfies λ-center separation, there exists a pruning $\mathcal{P}$ of the tree which equals $\mathcal{C}_X$. Our algorithm (Alg. 8) uses single-linkage to build a hierarchical clustering tree of X. We will show that when $\lambda \geq 3$ our algorithm achieves the above mentioned goal.

Theorem 4.18. *Given (X, d). For all $\lambda \geq 3$, Alg. 8 outputs a tree $\mathcal{T}$ with the following property. For all k and for all k-clusterings $\mathcal{C}_X^* = \{C_1^*, \ldots, C_k^*\}$ which satisfy λ-center separation w.r.t. X and k, the following holds. For every $1 \leq i \leq k$, there exists a node N_i in the tree T such that $C_i^* = N_i$.*

Proof. We will show that $\mathcal{C}_{\mathcal{X}}^*$ has strong stability ([Balcan et al., 2008]) which will complete the proof (Theorem 8 in [Balcan et al., 2008]). Let $A \subset C_i^*$ and $B \subseteq C_j^*$. Let $p \in A$ and $q \in C_i^* \setminus A$ be points which achieve the minimum distance between A and $C_i^* \setminus A$. If $c_i \in A$ then $d(p, q) \leq d(c_i, q) \leq r$. If $c_i \in C_i^* \setminus A$ then $d(p, q) \leq d(p, c_i) \leq r$. Hence, $d_{min}(A, C_i^* \setminus A) \leq r$. Similarly, we get that $d_{min}(A, B) > r$. □

Lower bound with no noise

We will prove that for $\lambda \leq 2$, finding any solution for λ-center separation is NP-hard. [Reyzin, 2012] proved that finding any solution for α-center proximity is NP-hard for $\alpha < 2$. Our reduction is same as the reduction used in Theorem 1 in [Reyzin, 2012] and hence we omit the proof.

Theorem 4.19. *Given a clustering instance (X, d) and the number of clusters k. For $\lambda < 2$, finding a clustering which satisfies λ-center separation is NP-hard.*

4.5.2 Center Separation in the presence of noise

In this section, we study the problem of recovering (λ, η)-center separated clusterings of a set X, in the presence of noise. Here is a more precise overview of the results of this section:

- *Positive result under sparse noise* - In Section 4.5.2, we show that if $\lambda \geq 4$ and $\eta \geq 1$; for any value of parameters r and t, Alg. 9 outputs a clustering which respects all clusterings $\mathcal{C}^*$ (of a subset of X) which satisfies (λ, η)-center proximity and $m(C^*) = t$ and $r(C^*) = r$.
- *Lower bound under sparse noise* - In Section 10, we show that, if $\lambda < 4$ and $\eta \leq 1$ then there is no tree and no list of 'small' size ($< 2^{k/2}$) which can capture all clusterings $\mathcal{C}$ (of subset of X) which satisfy (λ, η)-center proximity even for fixed values of the size of the smallest cluster ($m(C) = t$) and maximum radius ($r(C) = r$).
- *Lower bound with arbitrary noise* - In Section 10, we show that for a given value of parameters r and t, if $\lambda \leq 6$ and the number of noisy points exceeds $\frac{3}{2}t$ then no tree can capture all clusterings $\mathcal{C}$ (of a subset of X) which satisfy λ-center separation even for fixed $m(\mathcal{C}) = t$ and $r(\mathcal{C}) = r$. Identical result holds for 'small' ($< 2^{k/2}$) lists if the number of noisy points exceeds $\frac{3k}{2}t$.

Positive result under sparse noise

We are given a clustering instance (X, d) and parameters r and t. Our goal is to output a clustering $\mathcal{C}_X$ which has the following property. For every k, for every $S \subseteq X$ and for every k-clustering $\mathcal{C}_S$ which satisfies (λ, η)-center separation, the clustering $\mathcal{C}_X$ restricted to S equals $\mathcal{C}_S$.

In the next section, we propose a clustering algorithm (Alg. 9) and prove (Theorem 4.20) that our algorithm indeed achieves the above mentioned goal (under certain assumptions on the parameters λ and η). It is important to note that our algorithm only knows X and has no knowledge of the set S.

Intuitively, Alg. 9 works as follows. In the first phase, it constructs a list of balls which have radius at most r and contain at least t points. It then constructs a graph as follows. Each ball found in the first phase is represented by a vertex. If two balls have a 'large' intersection then there is an edge between the corresponding vertices in the graph. We then find the connected components in the graph which correspond to the clustering of the original set X.

Theorem 4.20. *Given a clustering instance (X, d) and parameters r and t. For every k, for every $S \subseteq X$ and for all k-clusterings $\mathcal{C}_S^* = \{S_1^*, \ldots, S_k^*\}$ which satisfy $(4, 1)$-center*

Algorithm 9: Alg. for (λ, η)-center separation with parameters t and r

Input: $(X, d), t$ and r
Output: A clustering $\mathcal{C}$ of the set X.

Phase 1
Let $\mathcal{L}$ denote the list of balls found so far. Initialize $\mathcal{L}$ to be the empty set. $\mathcal{L} = \emptyset$.
Go over all pairs of points $p, q \in X$ in increasing order of the distance $d(p, q)$. Let $B := B(p, d(p, q))$. If $|B| \geq t$ and $r(B) \leq r$ then
    $\mathcal{L} = \mathcal{L} \cup B$
Output the list of balls $\mathcal{L} = \{B_1, \ldots, B_l\}$ to the second phase of the algorithm.

Phase 2
Construct a graph $G = (V, E)$ as follows. $V = \{v_1, v_2, \ldots, v_l\}$. If $|B_i \cap B_j| \geq t/2$ then construct an edge between v_i and v_j.
Find connected components $(G_1, \ldots, G_k)$ in the graph G.
Merge all the points in the same connected component together to get a clustering $\mathcal{C} = \{C_1, \ldots, C_k\}$ of the set X.
Assign $x \in X \setminus \cup_i B_i$ to the closest cluster C_i. That is, $i := \arg\min_{j \in [k]} \min_{y \in C_j} d(x, y)$.
Output $\mathcal{C}$.

separation such that $m(\mathcal{C}^*_S) = t$ and $r(\mathcal{C}^*_S) = r$, the following holds. Alg. 9 outputs a clustering $\mathcal{C}_X$ such that $\mathcal{C}_X|_S = \mathcal{C}^*_S$.

Proof. Fix $\mathcal{S} \subseteq \mathcal{X}$. Denote by $r_i := r(S^*_i)$. Let $\mathcal{C}_\mathcal{X} = \{C_1, \ldots, C_k\}$ be the clustering outputed by the algorithm. Let $\mathcal{L} = \{B_1, \ldots, B_l\}$ be the list of balls as outputed by Phase 1 of Alg. 9. Let G be the graph as constructed in Phase 2 of the algorithm. Observe that $B = B(s_i, r_i) = S^*_i \in \mathcal{L}$. WLOG, denote this ball by $B^{(i)}$ and the corresponding vertex in the graph G by $v^{(i)}$. We will prove the theorem by proving two key facts.

F.1 If B_{i1} and B_{i2} intersect S^*_i then the vertices v_{i1} and v_{i2} are connected.
F.2 If B_{i1} intersects S^*_i and B_{j1} intersects S^*_j then v_{i1} and v_{j1} are disconnected in G.
Claim 4. *Let $\mathcal{L}, G, B^{(i)}$ and $v^{(i)}$ be as defined above. Let balls $B_{i1}, B_{i2} \in \mathcal{L}$ be such that $B_{i1} \cap S^*_i \neq \emptyset$ and $B_{i2} \cap S^*_i \neq \emptyset$. Then there exists a path between v_{i1} and v_{i2}.*

Assume that v_{i1} and $v^{(i)}$ are not connected by an edge. Hence, $|B_{i1} \setminus B^{(i)}| \geq t/2$. Since $\lambda > 4$, for all $j \neq i$, $B_{i1} \cap S^*_j = \emptyset$. Thus, $B_{i1} \setminus B^{(i)} \subseteq \mathcal{X} \setminus \mathcal{S}$. which contradicts

$|B_{i1} \cap \{\mathcal{X} \setminus \mathcal{S}\}| < t/2$. □

Claim 5. *Let the framework be as in Claim 4. Let $B_{i1} \in \mathcal{L}$ be such that $B_{i1} \cap S_i^* \neq \emptyset$ and B_{j1} be such that $B_{j1} \cap S_j^* \neq \emptyset$. Then v_{i1} and v_{j1} are disconnected in G.*

Assume that v_{i1} and v_{j1} are connected. Hence, there exists vertices v_i and v_n such that v_i and v_n are connected by an edge in G and $B_i \cap S_i^* \neq \emptyset$ and $B_n \cap S_n^* \neq \emptyset$ for some $n \neq i$. $|B_i \cap B_n| \geq t/2$. Now, $\lambda \geq 4$, thus $B_i \cap \{\mathcal{S} \setminus S_i^*\} = \emptyset$ and $B_n \cap \{\mathcal{S} \setminus S_n^*\} = \emptyset$. Thus, $B_i \cap B_n \subseteq \mathcal{X} \setminus \mathcal{S}$ which contradicts the sparseness assumption. □

Theorem 4.21. *Given (X, d) and parameters r and t. Alg. 9 runs in poly$(|X|)$.*

Proof. Let $n = |X|$. Phase 1 of Alg. 9 runs in $O(n^2)$ time. Phase 2 gets a list of size l. Constructing G and finding connected components takes $O(l^2)$ time. Hence, the algorithm runs in $O(n^2)$ time. □

Lower bound under sparse noise

Theorem 4.22. *Given the number of clusters k and parameters r and t. For all $\lambda < 4$ and $\eta \leq 1$, there exists a clustering instance (X, d) such that any clustering tree $\mathcal{T}$ of X has the following property. There exists $S \subseteq X$ and a k-clustering $\mathcal{C}_S = \{S_1, \ldots, S_k\}$ which satisfies (λ, η)-center separation such that $m(\mathcal{C}_S) = t$ and $r(\mathcal{C}_S) = r$, but $\mathcal{T}$ doesn't capture $\mathcal{C}_S$.*

Proof. The construction used is the similar to that in the proof of Thm. 4.14 with balls of unit radius. □

Theorem 4.23. *Given the number of clusters k and parameters r and t. For all $\lambda \leq 4$ and $\eta \leq 1$ there exists a clustering instance (X, d) such that any list $\mathcal{L}$ (of clusterings of X) has the following property. If $|\mathcal{L}| < 2^{\frac{k}{2}}$ then there exists $S \subseteq X$ and clustering $\mathcal{C}_S$ which satisfies (λ, η)-center separation and $m(\mathcal{C}_S) = t$ and $r(\mathcal{C}_S) = r$, but $\mathcal{L}$ doesn't capture $\mathcal{C}_S$.*

Proof. The construction used is the similar to that in the proof of Thm. 4.15 with balls of unit radius. □

Lower bound with arbitrary noise

The constructions are similar to the corresponding constructions for center proximity.

Theorem 4.24. *Given the number of clusters k and parameters r and t. For all $\lambda < 6$, there exists a clustering instance (X, d) such that any clustering tree $\mathcal{T}$ of X has the following property. There exists $S \subseteq X$ and there exists k-clustering $\mathcal{C}_S$ which satisfies λ-center separation such that $m(\mathcal{C}_S) = t$, $r(\mathcal{C}_S) = r$ and the following holds. If $|X \setminus S| \geq \frac{3t}{2}+5$, then $\mathcal{T}$ doesn't capture $\mathcal{C}_S$.*

Theorem 4.25. *Given the number of clusters k and parameters r and t. For all $\lambda \leq 6$ there exists (X, d) such that any list $\mathcal{L}$ (of clusterings of X) has the following property. There exists $S \subseteq X$ and there exists clustering $\mathcal{C}_S$ which satisfies λ-center separation such that $m(\mathcal{C}_S) = t$, $r(\mathcal{C}_S) = r$ and the following holds. If $|\mathcal{L}| < 2^{\frac{k}{2}}$ and $|X \setminus S| \geq \frac{k}{2}(\frac{3m(\mathcal{C}_S)}{2}+5)$, then $\mathcal{L}$ doesn't capture $\mathcal{C}_S$.*

Chapter 5

Noise-robust k-means clustering

Clustering aims to group similar data instances together while separating dissimilar ones. However, often many datasets have, on top of cohesive groups, a subset of "unstructured" points as well. In such cases, the goal is to detect the structure while simultaneously separating the unstructured data points. Clustering algorithms that achieve this goal are said to be *robust* to noise.

The most common approach to clustering views it as an optimization problem. The idea is to associate a cost (or objective) with each possible partition (into k subsets) of the input dataset and then try to find a partition which has minimum cost. Examples of common objective functions include k-means and k-median cost functions. However, it is easy to see that even the presence of one outlier can cause significant damage to the centers outputted by these methods.

In this chapter, we propose a generic method of regularization that can transform any clustering objective which outputs k clusters to one that outputs $k+1$ clusters. The algorithm is now allowed to 'discard' points into an extra 'garbage' or noise cluster by paying a constant regularization penalty. The intuition is that allowing the clustering algorithm to discard a few points should make it easier to detect the structure in the remaining non-noisy points. However, we prove that finding the optimal solution to the regularized objective is NP-hard [1].

From a theoretical perspective, one common approach to dealing with such hardness results is the following. (i) Consider a convex relaxation of the original objective function (which can be efficiently solved using standard techniques). (ii) Prove that under cer-

[1] Throughout this chapter, we consider the standard and regularized versions of the k-means objective

tain data niceness conditions the solution obtained by solving the relaxed objective function coincides with the optimal solution of the original objective function. For example, [Peng and Wei, 2007] used this strategy to design an algorithm based on the sdp-relaxation of the k-means objective function. [Awasthi et al., 2015] proved that under the 'stochastic ball assumption', the solution of the sdp-based approach is indeed the optimal k-means clustering.

One advantage of studying convex relaxations of clustering objectives over other clustering heuristics (like the Llyod's algorithm [Lloyd, 1982]) is that we can verify when the output solution is optimal. There was been lot of work in analyzing clustering algorithms under distributional and clusterability assumptions (see [Balcan and Liang, 2012], [Awasthi et al., 2012], [Kalai et al., 2010], [Achlioptas and McSherry, 2005], [Sanjeev and Kannan, 2001] and [Hsu and Kakade, 2013] and the references therein). However, often the algorithms are designed keeping the input conditions in mind. Convex relaxations have an advantage that they are not specifically tailored to a particular distribution. In this chapter, we use this approach (sdp relaxation of the regularized k-means objective) to design an efficient and noise-robust clustering algorithm.

Our framework is the following. We are given an input dataset X made of two components. The first is the clusterable subset I which satisfies a niceness property. Namely, I is the union of k unit balls B_i each separated by a distance of at least δ. The second is the unstructured or noise component N. Note that the clustering algorithm only sees X and is not aware of I or N. The goal is to design an efficient clustering algorithm $\mathcal{A}$ such that the output of the algorithm $\mathcal{A}(X)$ when restricted to I, is able to detect and recover the structure of I (namely, the balls B_i). In this chapter, we consider two choices for $\mathcal{A}$. The first based on SDP relaxation of the regularized k-means objective and the second based on the LP relaxation. For the noiseless case, [Awasthi et al., 2015] showed that for $\delta > 2\sqrt{2}(1 + 1/\sqrt{d})$ (where d is the dimension of the euclidean space) the algorithm based on SDP-relaxation of the standard k-means objective recovers (with high probability) the structure of I if the balls B_i are generated by an isotropic distribution (stochastic ball model [Iguchi et al., 2015], [Iguchi et al., 2017], [Awasthi et al., 2015] and [Nellore and Ward, 2015]).

In this chapter, we improve over previous results and give success guarantees in the regime $\delta > 2(1 + \sqrt{k/d})$ which is near-optimal for large d. In the presence of noise, we prove that the algorithm based on the SDP-relaxation of the regularized objective recovers the clustering of I for $\delta > 2(1+\sqrt{\zeta + k/d})$. Here ζ is a term which depends on the ratio of number of noisy points and the number of points in the smallest cluster. We obtain similar results for the LP based algorithm as well. However, in that the case both separation requirement and the restrictions on noise are stronger.

Table 5.1: Known results for clustering under stochastic ball model. The columns show the separation under which recovery guarantees hold. All these results are for the noise-less case

	Separation condition	Reference
k-median LP	$\delta > 3.75$	Thm. 1 in [Nellore and Ward, 2015]
	$\delta > 2$	Thm. 7 in [Awasthi et al., 2015]
k-means LP	$\delta > 4$	Thm. 9 in [Nellore and Ward, 2015]
k-means SDP	$\delta > 2\sqrt{2}(1 + \frac{1}{\sqrt{d}})$	Thm. 11 in [Awasthi et al., 2015]
	$\delta > 2 + \frac{k^2}{d} cond(\gamma)$	Thm. 2 in [Iguchi et al., 2015]
	$\delta > 2 + \frac{k^2}{d}$	Thm. 9 in [Iguchi et al., 2017]
	$\delta > 2 + \sqrt{\frac{k}{d}}$	Thm. 5.7

We also conduct simulation studies where we examine the effect of λ, the number of noisy points m, the separation δ and other parameters on the performance of our regularised SDP-based algorithm. We also perform experiments on the MNIST dataset. We observed that the regularised version performed better than k-means++ when the dataset had noisy points. In the absence of noise, the performance of both these algorithms were similar.

5.1 Related Work

[Ben-David and Haghtalab, 2014] also studied the problem of robustifying a clustering objective function. They gave a generic procedure that can transform any clustering objective with metric d to an objective function with metric d', where d' is a truncated version of the original metric. More specifically, $d'(x, y) = \min(\lambda, d(x, y))$. Then they show that if there exists an algorithm which can solve the optimization problem under the truncated metric then such an algorithm is robust to noise. [Georgogiannis, 2016] also studies breakdown properties of the truncated objective function, that is the number of outliers which can cause significant change in the estimates of any one of the centers. However, both these papers do not make any comment about how one would solve the optimization problem for both the standard or the truncated versions. In this work, we propose an efficient algorithm based on convex relaxation of our regularized objective function and give robustness

guarantees for the same.

The problem of recovering the underlying cluster structure in the presence and absence of noise has been studied before both in distribution-free and distribution-based settings. In the distribution-free setting, the goal is to prove that if the data has some structure (is *clusterable*) then the (proposed or existing) clustering algorithm recovers that structure in the presence or absence of noise. Such works, make no assumption on the distribution that generated the data. Different works define different notions of 'clusterable' data. In the current work, the separation requirement on the clusters was global. That is, the each cluster was separated by at least δ times the maximum radius amongst all the clusters. Another popular notion of clusterability is α-center proximity [Awasthi et al., 2012] which requires that two clusters be separated relative to their radii. [Balcan and Liang, 2012] consider a dataset which has α-center proximity except for an ϵ-fraction of points. They propose an efficient algorithm which provides an $1 + O(\epsilon)$ approximation to the optimal k-median solution for $\alpha > 2 + \sqrt{7}$.

In this work, we assume that the input satisfies the stochastic ball assumption. This model was considered by [Nellore and Ward, 2015] in the context of LP relaxation of the k-median objective function. They showed that if the separation between the clusters $\delta > 3.75$, then the lp relaxation finds the desired clustering. This bound was later improved to $\delta > 2$ in [Awasthi et al., 2015]. They also considered the LP relaxation of k-means objective and gave recovery guarantees when $\delta > 4$. Along the same lines, [Awasthi et al., 2015] considered the sdp relaxation of the k-means objective. They showed similar recovery guarantees when $\delta > 2\sqrt{2}(1 + \sqrt{d})$. This bound was later improved to $\delta > 2 + k^2/d$ by [Iguchi et al., 2017]. Table 5.1 contains a summary of the various results under the stochastic ball model. These results however apply only to the noiseless case. In this chapter, we consider the same framework but prove recovery guarantees even in the presence of a small fraction of noisy points.

Another line of work, which has some similarities to the framework in this chapter is estimating the parameters of a distribution in the presence of noise. [Lai et al., 2016] considered the problem of mean estimation of a gaussian when ϵ fraction of the points are corrupted by noise and propose an efficient algorithm that can estimate it with almost linear samples. [Diakonikolas et al., 2016] acheive an error of $O(\epsilon\sqrt{\log \epsilon})$ for mean estimation which was later improved to $O(\sqrt{\epsilon})$ [Diakonikolas et al., 2017], [Steinhardt et al., 2018]. [Charikar et al., 2017] and [Diakonikolas et al., 2016] considered the problem of recovering a mixture of k gaussians with few adversaries. [Charikar et al., 2017] show that it is possible to recover the means when the separation between the means is $\tilde{\Omega}(\sigma\sqrt{k})$ where σ^2 is an upper bound on the variance of the k gaussians. [Diakonikolas et al., 2016] output a distribution which is close to the target distribution in total variation distance. However,

their run-time is exponential in k. In this chapter our focus is on understanding the robustness properties of convex-relaxation type algorithms (specifically those based on relaxation of our regularized objective). Hence, these results our orthogonal to the current discussion.

[Peng and Wei, 2007] formulated the k-means objective as a 0-1 SDP and proved equivalence between the two formulations. They then relaxed the 0-1 SDP to a standard SDP. In this work, we use a similar proof and relaxation technique but for the regularised k-means objective.

5.2 Preliminaries and definition

Let $(\mathbf{M}, d)$ be a metric space. Given a finite set $\mathcal{X} \subset \mathbf{M}$, a k-clustering $\mathcal{C}$ of $\mathcal{X}$ partitions the set into k disjoint subsets $\mathcal{C} = \{C_1, \ldots, C_k\}$. An objective-based clustering algorithm associates a cost with each possible partition of $\mathcal{X}$ and then tries to find the clustering with minimum cost. Throughout this section, f denotes a function on the nonnegative reals.

Definition 5.1 ((k, f)-objective algorithm)**.** *Given $\mathcal{X} \subset \mathbf{M}$ and a distance function d, a (k, f)-objective based algorithm $\mathcal{A}$ tries to find centers $\mu_1, \ldots, \mu_k \in \mathbf{M}$ so as to minimize the following function*

$$\mathit{Cost}(\mu_1, \ldots, \mu_k) = \sum_{x \in \mathcal{X}} f(d(x, \mu(x))), \quad \mu(x) = \operatorname*{arg\,min}_{\mu \in \{\mu_1, \ldots, \mu_k\}} d(x, \mu). \tag{5.1}$$

Note that algorithm $\mathcal{A}$ may not find the optimal solution because for many common functions f, solving the optimization is NP-hard. Thus, heuristics are used that can get stuck at a local minimum. For example, when $f(x) = x^2$, the above definition corresponds to the k-means objective, and the Lloyd's algorithm that is used to solve this objective can get stuck at a local minima.

Definition 5.2 ((k, f)-λ-regularised objective algorithm)**.** *Given $\mathcal{X} \subset \mathbf{M}$ and a distance function d, a (k, f)-λ-regularised objective based algorithm $\mathcal{A}'$ tries to find centers $\mu_1, \ldots, \mu_k \in \mathbf{M}$ and set $\mathcal{I} \subseteq \mathcal{X}$ so as to minimize the following function*

$$\mathit{Cost}(\mu_1, \ldots, \mu_k, \mathcal{I}) = \sum_{x \in \mathcal{I}} f(d(x, \mu(x))) + \lambda |\mathcal{X} \setminus \mathcal{I}|, \tag{5.2}$$

where $\mu(x) = \arg\min_{\mu_i} d(x, \mu_i)$.

The regularised objective allows discarding certain points into a "garbage" cluster at the expense of paying a constant penalty. The intuition is that this will help the algorithm better detect the structure of the remaining points. We will see in §5.3 that minimizing this objective function is NP-hard for all $k \geq 1$.

5.2.1 Robustification paradigms

Definition 5.3 (λ-Regularised Paradigm)**.** *The λ-regularised paradigm is a robustification paradigm which takes as input a (k, f)-objective algorithm $\mathcal{A}$ and returns a (k, f)-λ-regularised objective algorithm $\mathcal{A}'$.*

In this work, we focus on robustification of the k-means objective. Hence, it is useful to define the regularised k-means objective as we will refer to it many times in the remainder of the chapter.

5.2.2 Robustness measure

Given two clusterings $\mathcal{C}$ and $\mathcal{C}'$ of the same set $\mathcal{X}$, we define the distance between them, $\Delta(\mathcal{C}, \mathcal{C}')$, as the fraction of pairs of points which are clustered differently in $\mathcal{C}$ than in $\mathcal{C}'$. Given $\mathcal{I} \subseteq \mathcal{X}$, $\mathcal{C}|\mathcal{I}$ denotes the restriction of the clustering $\mathcal{C}$ to the set $\mathcal{I}$.

Definition 5.4 (γ-robust [Ben-David and Haghtalab, 2014])**.** *Given $\mathcal{X} \subset \mathbf{M}$ and clustering algorithm $\mathcal{A}$, let $\mathcal{A}'$ be its robustified version obtained using any robustification paradigm. Given $\mathcal{I} \subseteq \mathcal{X}$, we say that $\mathcal{I}$ is γ-robust w.r.t $\mathcal{X} \setminus \mathcal{I}$ and $\mathcal{A}'$ if*

$$\Delta(\mathcal{A}'(\mathcal{X})|\mathcal{I}, \mathcal{A}(\mathcal{I})) \leq \gamma \tag{5.3}$$

This measure tries to quantify the difference in the clustering of the set $\mathcal{I}$ after the addition of 'noisy' points $\mathcal{X} \setminus \mathcal{I}$. If $\mathcal{A}'$ is indeed robust to noisy points, then the clusterings should be similar.

5.2.3 Regularised k-means objective

Given a finite set $\mathcal{X} \subset \mathbf{R}^d$ and an integer k, the regularised k-means objective aims to partition the data into $k+1$ clusters $\mathcal{C} = \{C_1, \ldots, C_k, C_{k+1}\}$ so as to solve

$$\min_{\substack{C_1,\ldots,C_{k+1} \\ c_1,\ldots,c_k}} \sum_{i=1}^{k} \sum_{x \in C_i} \|x - c_i\|_2^2 + \lambda |C_{k+1}|. \tag{5.4}$$

Note that the first term of the objective depends on the l_2 norm, while the second term depends only on the cardinality of the "noise" cluster. In order to make our objective function invariant to scaling, the regularization constant λ is added to the cost function.

Let $m(\mathcal{X}) := \min_{x \neq y \in \mathcal{X}} \|x - y\|_2^2$ and $N = |\mathcal{X}|$. Then, it is easy to see when $\lambda \leq \frac{m(\mathcal{X})}{2}$, then (5.4) admits a trivial solution: each cluster C_i for $i \leq k$ has exactly one point and all the remaining points are in C_{k+1}, leading to an objective $\lambda(N-k)$. Indeed, for any other clustering with $|C_i| = n_i$ its objective is at least $\sum_{i=1}^{k}(n_i - 1)m(\mathcal{X})/2 + \lambda n_{k+1} = (n - n_{k+1} - k)m(\mathcal{X})/2 + \lambda n_{k+1}$, where we have used the simple fact:

$$\forall C \subset \mathcal{X}, \quad \min_{c} \sum_{x \in C} \|x - c\|_2^2 \;=\; \frac{1}{2|C|} \sum_{x,y \in C} \|x - y\|_2^2. \tag{5.5}$$

Comparing the objectives we see the solution is indeed trivial when $\lambda \leq m(\mathcal{X})/2$. Surprisingly, for the interesting case when $\lambda > m(\mathcal{X})/2$, the problem suddenly becomes NP-hard, as we prove below.

5.3 Hardness of regularised k-means

In this section, we present hardness results for the regularised k-means objective.

We consider the interesting case of the regularized 1-means problem. It is well-known that 1-means can be solved in linear time [Bellman, 1973]. However, we will show by reducing an instance of the MAX-CLIQUE problem to the regularised 1-means problem is NP-hard. We give a proof sketch here. The technical details can be found in the appendix.

Theorem 5.5. *For all λ there exists a clustering instance $X \subset \mathbf{R}^p$ such that $\lambda > \frac{m(X)}{2}$ but finding the optimal solution to the regularized 1-means objective is NP-hard where recall that $m(X) := \min_{x \neq y \in X} \|x - y\|^2$.*

Proof sketch. The proof has two parts. We first show that for fixed λ the problem is NP-hard. The proof works by reducing an instance of MAX-CLIQUE to the regularised 1-mean instance. The idea is to define the distance between any pair of vertices as 1 if there exists an edge between them. If not, then define the distance as $1 + \Delta$ for a suitably chosen Δ. This construction guarentees that the problem is NP-hard for at least one $\lambda > \frac{m(\mathcal{X})}{2}$. Our result then follows using a scaling argument. □

Note that the same argument (as in the appendix) infact works for $k \geq 2$ where we would reduce from the MAX-k-CLIQUE problem (cover maximum vertices by k cliques).

Algorithm 10: SDP-based regularised k-means algorithm

Input: $\mathcal{X} \subset R^d$, k, and hyperparameter λ.
Output: $\mathcal{C}' := \{C_1, \ldots, C_k, C_{k+1}\}$.

1 Compute the matrix $D_{ij} = \|x_i - x_j\|_2^2$.
2 Solve the SDP (Eqn. 5.6) using any standard SDP solver and obtain matrix Z and vector y.
3 Use the rounding procedure (Alg. 11) to obtain the partition $\mathcal{C}'$.

We leave this proof as a simple exercise for the reader. Thus, we show that regularised k-means is hard for all $k \geq 1$.

5.4 The regularised k-means SDP-based algorithm

In the previous section, we showed that the regularised k-means objective is NP-hard to optimize. Hence, we cannot hope to solve the problem exactly unless $P = NP$. In this section, we develop an algorithm based on semi-definite programming relaxation of the regularised objective.

[Peng and Wei, 2007] developed an algorithm $\mathcal{A}$ (Alg. 10 with $\lambda = \infty$) which tries to minimize the k-means objective. They obtained a convex relaxation of the k-means objective and solved it polynomially using standard solvers. In this section, we use the same technique to obtain and efficiently solve the convex relaxation of the regularised k-means objective. Our algorithm $\mathcal{A}'$ (Alg. 10) is the robustified version of $\mathcal{A}$ using the λ-regularised paradigm. In §5.4.1, we give the details of how we transform the regularised objective into an SDP. §5.4.2 has our main results where we give robustness guarentees for $\mathcal{A}'$.

5.4.1 The SDP-based algorithm

The SDP relaxation of the regularised k-means objective is obtained in two steps. Using similar technique to that of [Peng and Wei, 2007], we translate Eqn. 5.4 into a 0-1 SDP (Eqn. 5.6). We then prove that solving the 0-1 SDP exactly is equivalent to solving the regularised k-means problem exactly. Then, we relax some of the constraints of the 0-1 SDP to obtain a tractable SDP which we then solve using standard solvers. We then describe

the rounding procedure which uses the solution of the SDP to construct a clustering of the original dataset.

$$\textbf{0-1 SDP} \begin{cases} \min_{Z,y} & \mathrm{Tr}(DZ) + \lambda\langle \mathbf{1}, y\rangle \\ \text{s.t.} & \mathrm{Tr}(Z) = k \\ & Z \cdot \mathbf{1} + y = \mathbf{1} \\ & Z \geq 0, Z^2 = Z, Z^T = Z \\ & y \in \{0,1\}^n \end{cases} \xrightarrow{\text{relaxed}} \textbf{SDP} \begin{cases} \min_{Z,y} & \mathrm{Tr}(DZ) + \lambda\langle \mathbf{1}, y\rangle \\ \text{s.t.} & \mathrm{Tr}(Z) = k \\ & \left(\frac{Z+Z^T}{2}\right) \cdot \mathbf{1} + y = \mathbf{1} \\ & Z \geq 0, y \geq 0, Z \succeq 0 \end{cases} \tag{5.6}$$

Theorem 5.6. *Finding a solution to the 0-1 SDP (5.6) is equivalent to finding a solution to the regularised k-means objective (5.4).*

Equation 5.6 shows our 0-1 SDP formulation. The optimization is NP-hard as it is equivalent to the regularised k-means objective. Hence, we consider a convex relaxation of the same. First, we replace $Z^2 = Z$ with $Z \succeq 0$. In addition, we relax $y \in \{0,1\}^n$ to $y \geq 0$, as the constraint $y \leq 1$ is redundant. Using these relaxations, we obtain the SDP formulation for our objective function.

We solve the SDP using standard solvers [Yang et al., 2015] thereby obtaining Z, y. The proof of Thm. 5.6 showed that the optimal solution of 0-1 SDP is of the following form. Z is a $n \times n$ block diagonal matrix of the form $\mathrm{diag}(Z_{I_1}, \ldots, Z_{I_k}, 0)$, where $n = |\mathcal{X}|$ and $Z_{I_i} = \frac{1}{|C_i|}\mathbf{1}\ \mathbf{1}^T$. Thus, given Z, we can extract the set of cluster centers $C = ZX$ which is an $n \times d$ matrix. Each row C_i contains the cluster center to which data point x_i belongs. For the points x_j assigned to the noise cluster, the corresponding row C_j is zero and $y_j = 1$. The SDP solver does not always return the optimal solution, as the relaxation is not exact. However, we expect that it returns a near-optimal solution. Hence, given Z and y returned by the solver, we use Alg. 11 to extract a clustering of our original dataset. The *threshold* parameter indicates our confidence that a given point is noise. In our experiments, we have used a threshold of 0.5.

5.4.2 Robustness guarantees

Assume that we are given a set $\mathcal{I}$ of k well-separated balls in $\mathbf{R}^d$. That is, $\mathcal{I} := \cup_{i=1}^k B_i$ where each B_i is a ball of radius at most one and centered at μ_i such that $\|\mu_i - \mu_j\| \geq \delta$. On top of this structure, points are added from the set $\mathcal{N}$. Let $\mathcal{A}$ and $\mathcal{A}'$ be the SDP

Algorithm 11: Regularised k-means rounding procedure

Input: $Z \subset \mathbb{R}^{n\times n}$, $y \subset \mathbb{R}^n$, $\mathcal{X}$, and $threshold \in [0,1]$.
Output: $\mathcal{C}'$.

If $y_i > threshold$ then
  Delete z_i and z_i^T from Z. Put x_i in C_{k+1}.
  Delete x_i from X.
k-cluster the columns of X^TZ to obtain clusters $C_1, \ldots, C_k$.
Output $\mathcal{C}' = \{C_1, \ldots, C_k, C_{k+1}\}$.

based standard and regularised k-means algorithm respectively (as defined in the begining of §5.4). We will show that $\mathcal{I}$ is 0-robust w.r.t $\mathcal{A}'$ and $\mathcal{N}$ under certain conditions on δ and mildness properties of the set $\mathcal{N}$. To show this, we need to compare the clusterings $\mathcal{A}(\mathcal{I})$ and $\mathcal{A}'(\mathcal{X})|_{\mathcal{I}}$. We first prove recovery guarentees for $\mathcal{A}$ in the absense of noisy points.

Theorem 5.7. *Let $\mathcal{P}$ denote the isotropic distribution on the unit ball centered at origin in $\mathbf{R}^d$. Given points $\mu_1, \ldots, \mu_k$ such that $\|\mu_i - \mu_j\| > \delta > 2$. Let $\mathcal{P}_i$ be the measure $\mathcal{P}$ translated with respect to the center μ_i. Let each B_i is drawn i.i.d w.r.t the distribution $\mathcal{P}_i$.*

Given a clustering instance $\mathcal{I} \subset \mathbf{R}^{N\times d}$ and k where $\mathcal{I} := \cup_{i=1}^k B_i$. Define $n := \min_{i\in[k]} |B_i|$ and $\rho = \frac{N}{nk}$. If

$$\delta > 1 + \sqrt{1 + \frac{2\theta\rho k}{d}\Big(1 + \frac{1}{\log N}\Big)^2}$$

where $\theta = \mathbf{E}[\|x_{pi} - c_p\|^2] < 1$, then there exists a constant $c > 0$ such that with probability at least $1 - 2d\exp(\frac{-cN\theta}{d\log^2 N})$ the k-means SDP (Alg. 10 with $\lambda = \infty$) finds the intended cluster solution $\mathcal{C}^* = \{B_1, \ldots, B_k\}$.

Thm. 5.7 improves the result of Thm. 11 in [Awasthi et al., 2015]. Under the stochastic ball assumption, they showed that the k-means SDP finds the intended solution for $\delta > 2\sqrt{2}(1 + \frac{1}{\sqrt{d}})$. For $k \ll d$, which is the case in many situations our bounds are optimal in terms of the separation requirement of the clusters. [Iguchi et al., 2015] obtained similar results but for $\delta > 2 + \frac{k^2}{d}cond(\mathcal{I})$. However, the condition number (ratio of maximum distance between any two centers to the minimum distance between any two centers) can be arbitrarily large. [Iguchi et al., 2017] improve the separation requirement to $\delta > 2 + \frac{k^2}{d}$. Asymptotically, as d goes to ∞ their bound matches our result. We also have a better dependence on the number of clusters k while they have a better dependence on the dimension d.

Next, we analyse the recovery guarentees for $\mathcal{A}'$ in the presence of noisy points $\mathcal{N}$. We decompose the noisy points into two disjoint sets $\mathcal{N}_1$ and $\mathcal{N}_2$. The set $\mathcal{N}_2$ consists of all the points which are far from any of the points in $\mathcal{I}$. The set $\mathcal{N}_1$ consists of points which are close to at least one of the clusters. We also require that any point in $\mathcal{N}_1$ has an α-margin w.r.t to the centers of the balls $B_1, \ldots, B_k$. That is the difference of the distance between any point in $\mathcal{N}_1$ to a cluster center is at least α. Now, we will show that if $\mathcal{N}$ has the aforementioned properties then $\mathcal{I}$ is robust w.r.t the regularised SDP algorithm $\mathcal{A}'$.

Theorem 5.8. *Let $\mathcal{P}$ denote the isotropic distribution on the unit ball centered at origin in $\mathbf{R}^d$. Given centers $\mu_1, \ldots, \mu_k$ such that $\|\mu_i - \mu_j\| > \delta > 2$. Let $\mathcal{P}_i$ be the measure $\mathcal{P}$ translated with respect to the center μ_i. Let B_i is drawn i.i.d w.r.t the distribution $\mathcal{P}_i$.*

Given a clustering instance $\mathcal{X} \subset \mathbf{R}^{N \times d}$ and k. Let $\mathcal{X} := \mathcal{I} \cup \mathcal{N}$ where $\mathcal{I} := \cup_{i=1}^{k} B_i$. Let $\mathcal{N} = \mathcal{N}_1 \cup N_2$ have the following properties. For all $n \in N_1$ and for all i, j, we have that $|\|n-\mu_i\|^2 - \|n-\mu_j\|^2| \geq \alpha$. For all $n \in \mathcal{N}_2$ and for all $x \in \mathcal{I}, \|n-x\| \geq \nu \geq \sqrt{(\delta-1)^2+1}$. Note that $\mathcal{N}_1 \cap \mathcal{N}_2 = \phi$. Let $n = \min_i |B_i|$ and $\epsilon = \frac{|\mathcal{N}_1|}{n}$ and $\rho = \frac{|I|}{nk}$. If

- $\delta > 2 + \sqrt{O(\epsilon) + \frac{2\rho k\theta(1+1/\log(|I|))^2}{d}}$
- $\alpha \geq O(\epsilon) + \frac{2\rho k\theta(1+1/\log|I|)^2}{d}$
- $\frac{|\mathcal{N}_2|}{n} \leq \frac{\delta^2 - 2\delta - O(\epsilon)}{\lambda}$

then there exists a constant $c_2 > 0$ such that with probability at least $1 - 2d\exp(\frac{-c_2|I|\theta}{d\log^2|I|})$ the regularised k-means SDP (Alg. 10) finds the intended cluster solution $\mathcal{C}^* = \{C_1, \ldots, C_k, \mathcal{N}_2\}$ where $B_i \subseteq C_i$ when given $\mathcal{X}$ and $\delta^2 + 2\delta \geq \lambda \geq (\delta-1)^2 + 1$ as input.

The proof of both the Thms. 5.7 and 5.8 use the following ideas. We construct a dual for the SDP. We then show that when the conditions of our theorems are satisfied then there exists a feasible solution for the dual program. Moreover, the objective function value of primal and dual sdp program are the same. Hence, the solution found is indeed optimal. We use similar techniques as in the proof of Thm. 11 in [Awasthi et al., 2015]. However, our analysis is tighter which helps us to obtain better bounds. The details are in the appendix.

We have also developed a regularized version of the k-means LP based algorithm. The details and the robustness guarantees for the LP-based algorithm are in the appendix.

5.5 Experiments

We ran several experiments to analyse the performance of our regularised k-means algorithm. The first set of experiments were simulations done on synthetic data. The second set of experiments were done on real world datasets like MNIST where we compared the performance of our algorithm against other popular clustering algorithms like k-means++. All our experiments were run on Matlab. We solved the SDP formulation using the Matlab SDPNAL+ package [Yang et al., 2015]. To run k-means++ we used the standard implementation of the algorithm available on Matlab.

5.5.1 Simulation studies

The goal of these sets of experiments was to understand the effect of different parameters on the performance of the regularised SDP algorithm. Given the number of clusters k, the separation between the clusters δ, the dimension of the space d, the number of points in each cluster n and the number of noisy points m. We generate a clustering instance $\mathcal{X}$ in $\mathbf{R}^d$ as follows. We first pick k seed points $\mu_1, \ldots, \mu_k$ such that each of these points are separated by at least δ. Next we generate n points in the unit ball centered at each of the $\mu_i's$. Finally we add m points uniformly at random.

We analyse the performance of the regularised SDP algorithm as the parameters change. The most crucial amongst them is the separation between the clusters δ, the regularization constant λ and the dimension d. Fig. 5.1 shows the heatmap under different parameter values. For each setting of the parameters, we generated 50 random clustering instances. We then calculated the fraction of times the regularised sdp was able to recover the true clustering of the data. If the fraction is close to one, then its color on the plot is light. Darker colors represent values close to zero.

We see an interesting transition for λ. When λ is 'too small' then the probability of recovering the true clustering is also low. As λ increases the probability of success goes up which are represented by the light colors. However, if we increase λ to a very high value then the success probability again goes down. This shows that there is a 'right' range of λ as was also predicted by our theoretical analysis.

Another parameter of interest is the dimension of the space d. Note that from our theoretical analysis, we know that both the probability of success and the separation depend on d. Fig. 5.1 shows that for very low dimension, the regularised sdp fails to perfectly recover the underlying clustering. However, as the dimension grows so does the probability of success. For these two simulations, we fixed the number of points per cluster $n = 30$,

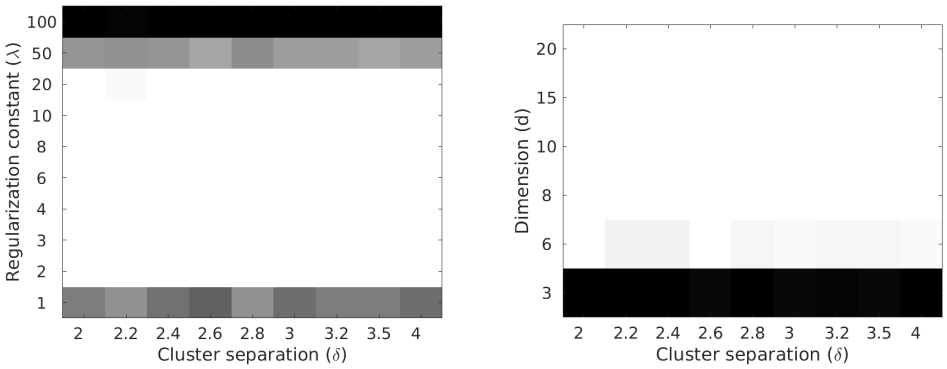


Figure 5.1: Heatmap showing the probability of success of the k-means regularised sdp algorithm. Lighter color indicates probability closer to one while darker indicates probability closer to zero.

$k = 8$ and the number of noisy points $m = 30$. We have similar plots for (δ, n) and (δ, k) and (δ, m) and (n, m). These plots very mostly light colored as long as the number of noisy points was not too large ($\frac{m}{n} \leq 5$). Hence, due to space constraints, we have included them only in the appendix.

5.5.2 Results on MNIST dataset

We compare our regularised SDP algorithm against k-means++ on the MNIST dataset. MNIST is a dataset of images of handwritten digits from zero to nine. It contains 60,000 training images and 10,000 test images. We choose $k = 4$ different classes and randomly sample a total of $N = 1,000$ images from these classes. We then run both our regularised SDP algorithm and the k-means++ algorithm on this dataset. We repeat this process for 10 different random samples of MNIST. We measure the performance of the two algorithms in terms of the precision and recall over the pairs of points in the same cluster. Given a clustering $\mathcal{C}$ and some target clustering C^*. Define the precision p of $\mathcal{C}$ as the fraction of pairs that were in the same cluster according to $\mathcal{C}^*$ given that they were in the same cluster according to $\mathcal{C}$. The recall r of $\mathcal{C}$ is the fraction of pairs that were in the same clustering according to $\mathcal{C}$ given that they were in the same cluster according to $\mathcal{C}^*$. We finally measure the f_1 score of the clustering $\mathcal{C}$ as the harmonic mean of its precision and recall. $f_1 = \frac{2pr}{p+r}$.

Note that the regularised algorithm outputs $k + 1$ clusters. Hence, to make a fair comparison, we finally assign each point in the noisy cluster (C_{k+1}) to one of the clusters

$C_1, \ldots, C_k$ depending upon the distance of the point to the clusters. Another point is that the f_1 measures are sensitive to the choice of the k digits or classes. For some choice of k classes, the f_1 measures for both the algorithms are higher than compared to other classes. This shows that some classes are more difficult to cluster than other classes. Hence, we only report the difference in performance of the two algorithms.

We report the performance on datasets with and without noisy points. The first is when there are no outliers or noisy points. In this case, the difference in the f_1 values was about 4.34% in favor of k-means++. We then added noisy points to the dataset. In the first case, we added images from different datasets like EMNIST (images of handwritten letters). In this case, the difference was 2.54% in favor of the regularised algorithm. In the second case, besides images from different datasets, we also added a few random noisy points to the MNIST dataset. In this case, the difference increased further to about 6.9% in favor of the regularised algorithm.

5.6 Conclusion

We introduced a regularisation paradigm which can transform any center-based clustering objective to one that is more robust to the addition of noisy points. We proved that regularised objective is NP-hard for common cost functions like k-means. We then obtained regularised versions of an existing clustering algorithm based on convex (sdp) relaxation of the k-means cost. We then proved noise robustness guarentees for the regularised algorithm. The proof improved existing bounds (in terms of cluster separation) for sdp-based standard (non-regularised) k-means algorithm. Our experiments showed that regularised sdp-based k-means performed better than existing algorithms like k-means++ on MNIST especially in the presence of noisy points.

References

[pub,] Bibliography of scientific publications. https://www13.hpi.uni-potsdam.de/fileadmin/user_upload/fachgebiete/naumann/projekte/dude/CORA.xml.

[pro,] E-commerce. https://dbs.uni-leipzig.de/en/research/projects/object_matching/fever/benchmark_datasets_for_entity_resolution.

[res,] Fodor's and zagat's restaurant guides. http://www.cs.utexas.edu/users/ml/riddle/data.html.

[A000108,] A000108, S. The on-line encyclopedia of integer sequences. *published electronically at https://oeis.org, 2010*.

[Achlioptas and McSherry, 2005] Achlioptas, D. and McSherry, F. (2005). On spectral learning of mixtures of distributions. In *International Conference on Computational Learning Theory*, pages 458–469. Springer.

[Ackerman and Ben-David, 2009] Ackerman, M. and Ben-David, S. (2009). Clusterability: A theoretical study. In *International Conference on Artificial Intelligence and Statistics*, pages 1–8.

[Ailon et al., 2018] Ailon, N., Bhattacharya, A., and Jaiswal, R. (2018). Approximate correlation clustering using same-cluster queries. In *Latin American Symposium on Theoretical Informatics*, pages 14–27. Springer.

[Ailon et al., 2017] Ailon, N., Bhattacharya, A., Jaiswal, R., and Kumar, A. (2017). Approximate clustering with same-cluster queries. *arXiv preprint arXiv:1704.01862*.

[Ashtiani and Ben-David, 2015] Ashtiani, H. and Ben-David, S. (2015). Representation learning for clustering: A statistical framework. *arXiv preprint arXiv:1506.05900*.

[Ashtiani and Ghodsi, 2015] Ashtiani, H. and Ghodsi, A. (2015). A dimension-independent generalization bound for kernel supervised principal component analysis. In *Proceedings of The 1st International Workshop on Feature Extraction: Modern Questions and Challenges, NIPS*, pages 19–29.

[Ashtiani et al., 2016] Ashtiani, H., Kushagra, S., and Ben-David, S. (2016). Clustering with same-cluster queries. In *Advances in neural information processing systems*, pages 3216–3224.

[Aslam et al., 2006] Aslam, J., Pelekhov, E., and Rus, D. (2006). *The Star Clustering Algorithm for Information Organization*, pages 1–23. Springer Berlin Heidelberg, Berlin, Heidelberg.

[Awasthi et al., 2015] Awasthi, P., Bandeira, A. S., Charikar, M., Krishnaswamy, R., Villar, S., and Ward, R. (2015). Relax, no need to round: Integrality of clustering formulations. In *Proceedings of the 2015 Conference on Innovations in Theoretical Computer Science*, pages 191–200. ACM.

[Awasthi et al., 2012] Awasthi, P., Blum, A., and Sheffet, O. (2012). Center-based clustering under perturbation stability. *Information Processing Letters*, 112(1):49–54.

[Balcan and Blum, 2008] Balcan, M.-F. and Blum, A. (2008). Clustering with interactive feedback. In *International Conference on Algorithmic Learning Theory*, pages 316–328. Springer.

[Balcan et al., 2012] Balcan, M.-F., Blum, A., Fine, S., and Mansour, Y. (2012). Distributed learning, communication complexity and privacy. *arXiv preprint arXiv:1204.3514*.

[Balcan et al., 2008] Balcan, M.-F., Blum, A., and Vempala, S. (2008). A discriminative framework for clustering via similarity functions. In *Proceedings of the fortieth annual ACM symposium on Theory of computing*, pages 671–680. ACM.

[Balcan and Liang, 2012] Balcan, M. F. and Liang, Y. (2012). Clustering under perturbation resilience. In *Automata, Languages, and Programming*, pages 63–74. Springer.

[Bansal et al., 2004] Bansal, N., Blum, A., and Chawla, S. (2004). Correlation clustering. *Machine Learning*, 56(1-3):89–113.

[Basu et al., 2002] Basu, S., Banerjee, A., and Mooney, R. (2002). Semi-supervised clustering by seeding. In *In Proceedings of 19th International Conference on Machine Learning (ICML-2002*. Citeseer.

[Basu et al., 2004] Basu, S., Bilenko, M., and Mooney, R. J. (2004). A probabilistic framework for semi-supervised clustering. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 59–68. ACM.

[Bellman, 1973] Bellman, R. (1973). A note on cluster analysis and dynamic programming. *Mathematical Biosciences*, 18:311–312.

[Ben-David, 2015] Ben-David, S. (2015). Computational feasibility of clustering under clusterability assumptions. *CoRR*, abs/1501.00437.

[Ben-David, 2015] Ben-David, S. (2015). Computational feasibility of clustering under clusterability assumptions. *arXiv preprint arXiv:1501.00437*.

[Ben-David and Haghtalab, 2014] Ben-David, S. and Haghtalab, N. (2014). Clustering in the presence of background noise. In *ICML*, pages 280–288.

[Ben-David and Reyzin, 2014] Ben-David, S. and Reyzin, L. (2014). Data stability in clustering: A closer look. *Theoretical Computer Science*, 558:51–61.

[Bilenko et al., 2003] Bilenko, M., Mooney, R., Cohen, W., Ravikumar, P., and Fienberg, S. (2003). Adaptive name matching in information integration. *IEEE Intelligent Systems*, 18(5):16–23.

[Bilu and Linial, 2012] Bilu, Y. and Linial, N. (2012). Are stable instances easy? *Combinatorics, Probability and Computing*, 21(05):643–660.

[Blumer et al., 1989] Blumer, A., Ehrenfeucht, A., Haussler, D., and Warmuth, M. K. (1989). Learnability and the vapnik-chervonenkis dimension. *Journal of the ACM (JACM)*, 36(4):929–965.

[Broder, 1997] Broder, A. Z. (1997). On the resemblance and containment of documents. In *Compression and Complexity of Sequences 1997. Proceedings*, pages 21–29. IEEE.

[Broder et al., 2000] Broder, A. Z., Charikar, M., Frieze, A. M., and Mitzenmacher, M. (2000). Min-wise independent permutations. *Journal of Computer and System Sciences*, 60(3):630–659.

[Brown, 2011] Brown, D. G. (2011). How i wasted too long finding a concentration inequality for sums of geometric variables. *Found at https://cs. uwaterloo. ca/˜ browndg/negbin. pdf*, 6.

[Charikar et al., 2005] Charikar, M., Guruswami, V., and Wirth, A. (2005). Clustering with qualitative information. *Journal of Computer and System Sciences*, 71(3):360–383.

[Charikar et al., 2017] Charikar, M., Steinhardt, J., and Valiant, G. (2017). Learning from untrusted data. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, pages 47–60. ACM.

[Charikar, 2002] Charikar, M. S. (2002). Similarity estimation techniques from rounding algorithms. In *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, pages 380–388. ACM.

[Cochinwala et al., 2001] Cochinwala, M., Kurien, V., Lalk, G., and Shasha, D. (2001). Efficient data reconciliation. *Information Sciences*, 137(1-4):1–15.

[Cohen, 1998] Cohen, W. W. (1998). Integration of heterogeneous databases without common domains using queries based on textual similarity. In *ACM SIGMOD Record*, volume 27, pages 201–212. ACM.

[Cormen et al., 2009] Cormen, T. H., Leiserson, C. E., Rivest, R. L., and Stein, C. (2009). *Introduction to Algorithms, Third Edition*. The MIT Press, 3rd edition.

[Cuesta-Albertos et al., 1997] Cuesta-Albertos, J., Gordaliza, A., Matrán, C., et al. (1997). Trimmed k-means: An attempt to robustify quantizers. *The Annals of Statistics*, 25(2):553–576.

[Dasgupta, 2008] Dasgupta, S. (2008). *The hardness of k-means clustering*. Department of Computer Science and Engineering, University of California, San Diego.

[Dave, 1993] Dave, R. N. (1993). Robust fuzzy clustering algorithms. In *Fuzzy Systems, 1993., Second IEEE International Conference on*, pages 1281–1286. IEEE.

[Demaine et al., 2006] Demaine, E. D., Emanuel, D., Fiat, A., and Immorlica, N. (2006). Correlation clustering in general weighted graphs. *Theoretical Computer Science*, 361(2-3):172–187.

[Diakonikolas et al., 2016] Diakonikolas, I., Kamath, G., Kane, D. M., Li, J., Moitra, A., and Stewart, A. (2016). Robust estimators in high dimensions without the computational intractability. In *IEEE 57th Annual Symposium on Foundations of Computer Science, FOCS 2016, 9-11 October 2016, Hyatt Regency, New Brunswick, New Jersey, USA*, pages 655–664.

[Diakonikolas et al., 2017] Diakonikolas, I., Kamath, G., Kane, D. M., Li, J., Moitra, A., and Stewart, A. (2017). Being robust (in high dimensions) can be practical. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, pages 999–1008.

[Elmagarmid et al., 2007] Elmagarmid, A. K., Ipeirotis, P. G., and Verykios, V. S. (2007). Duplicate record detection: A survey. *IEEE Transactions on knowledge and data engineering*, 19(1):1–16.

[García-Escudero et al., 2008] García-Escudero, L. A., Gordaliza, A., Matrán, C., and Mayo-Iscar, A. (2008). A general trimming approach to robust cluster analysis. *The Annals of Statistics*, pages 1324–1345.

[Garey and Johnson, 2002] Garey, M. R. and Johnson, D. S. (2002). *Computers and intractability*, volume 29. wh freeman New York.

[Georgogiannis, 2016] Georgogiannis, A. (2016). Robust k-means: a theoretical revisit. In *Advances in Neural Information Processing Systems*, pages 2891–2899.

[Gionis et al., 1999] Gionis, A., Indyk, P., Motwani, R., et al. (1999). Similarity search in high dimensions via hashing. In *Vldb*, volume 99, pages 518–529.

[Hassanzadeh et al., 2009] Hassanzadeh, O., Chiang, F., Lee, H. C., and Miller, R. J. (2009). Framework for evaluating clustering algorithms in duplicate detection. *Proc. VLDB Endow.*, 2(1):1282–1293.

[Hernández and Stolfo, 1995] Hernández, M. A. and Stolfo, S. J. (1995). The merge/purge problem for large databases. In *ACM Sigmod Record*, volume 24, pages 127–138. ACM.

[Hsu and Kakade, 2013] Hsu, D. and Kakade, S. M. (2013). Learning mixtures of spherical gaussians: moment methods and spectral decompositions. In *Proceedings of the 4th conference on Innovations in Theoretical Computer Science*, pages 11–20. ACM.

[Iguchi et al., 2015] Iguchi, T., Mixon, D. G., Peterson, J., and Villar, S. (2015). On the tightness of an sdp relaxation of k-means. *arXiv preprint arXiv:1505.04778*.

[Iguchi et al., 2017] Iguchi, T., Mixon, D. G., Peterson, J., and Villar, S. (2017). Probably certifiably correct k-means clustering. *Mathematical Programming*, 165(2):605–642.

[Indyk and Motwani, 1998] Indyk, P. and Motwani, R. (1998). Approximate nearest neighbors: towards removing the curse of dimensionality. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, pages 604–613. ACM.

[Jaro, 1980] Jaro, M. A. (1980). *UNIMATCH, a Record Linkage System: Users Manual*. Bureau of the Census.

[Kalai et al., 2010] Kalai, A. T., Moitra, A., and Valiant, G. (2010). Efficiently learning mixtures of two gaussians. In *Proceedings of the forty-second ACM symposium on Theory of computing*, pages 553–562. ACM.

[Kulis et al., 2009] Kulis, B., Basu, S., Dhillon, I., and Mooney, R. (2009). Semi-supervised graph clustering: a kernel approach. *Machine learning*, 74(1):1–22.

[Kushagra et al., 2019a] Kushagra, S., Ben-David, S., and Ilyas, I. (2019a). Semi-supervised clustering for deduplication. In *AISTATS*.

[Kushagra et al., 2016] Kushagra, S., Samadi, S., and Ben-David, S. (2016). Finding meaningful cluster structure amidst background noise. In *International Conference on Algorithmic Learning Theory*, pages 339–354. Springer.

[Kushagra et al., 2019b] Kushagra, S., Saxena, H., Ilyas, I., and Ben-David, S. (2019b). A semi-supervised framework of clustering selection for deduplication. In *ICDE*.

[Kushagra et al., 2017] Kushagra, S., Yu, Y., and Ben-David, S. (2017). Provably noise-robust, regularised k-means clustering. *arXiv preprint arXiv:1711.11247*.

[Lai et al., 2016] Lai, K. A., Rao, A. B., and Vempala, S. (2016). Agnostic estimation of mean and covariance. In *IEEE 57th Annual Symposium on Foundations of Computer Science, FOCS 2016, 9-11 October 2016, Hyatt Regency, New Brunswick, New Jersey, USA*, pages 665–674.

[Levenshtein, 1966] Levenshtein, V. I. (1966). Binary codes capable of correcting deletions, insertions, and reversals. In *Soviet physics doklady*, volume 10, pages 707–710.

[Lloyd, 1982] Lloyd, S. (1982). Least squares quantization in pcm. *IEEE transactions on information theory*, 28(2):129–137.

[Mahajan et al., 2009] Mahajan, M., Nimbhorkar, P., and Varadarajan, K. (2009). The planar k-means problem is np-hard. In *WALCOM: Algorithms and Computation*, pages 274–285. Springer.

[Maimon and Browarnik, 2009] Maimon, O. and Browarnik, A. (2009). Nhecd-nano health and environmental commented database. In *Data mining and knowledge discovery handbook*, pages 1221–1241. Springer.

[Mazumdar and Saha, 2017] Mazumdar, A. and Saha, B. (2017). Clustering with noisy queries. In *Advances in Neural Information Processing Systems*, pages 5788–5799.

[Megiddo and Supowit, 1984] Megiddo, N. and Supowit, K. J. (1984). On the complexity of some common geometric location problems. *SIAM journal on computing*, 13(1):182–196.

[Mitzenmacher and Upfal, 2005] Mitzenmacher, M. and Upfal, E. (2005). *Probability and computing: Randomized algorithms and probabilistic analysis*. Cambridge university press.

[Monge et al., 1996] Monge, A. E., Elkan, C., et al. (1996). The field matching problem: Algorithms and applications. In *KDD*, pages 267–270.

[Nellore and Ward, 2015] Nellore, A. and Ward, R. (2015). Recovery guarantees for exemplar-based clustering. *Information and Computation*, 245:165–180.

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

[Pennington et al., 2014] Pennington, J., Socher, R., and Manning, C. D. (2014). Glove: Global vectors for word representation. In *In EMNLP*.

[Reyzin, 2012] Reyzin, L. (2012). Data stability in clustering: A closer look. In *Algorithmic Learning Theory*, pages 184–198. Springer.

[Sanjeev and Kannan, 2001] Sanjeev, A. and Kannan, R. (2001). Learning mixtures of arbitrary gaussians. In *Proceedings of the thirty-third annual ACM symposium on Theory of computing*, pages 247–257. ACM.

[Shalev-Shwartz and Ben-David, 2014] Shalev-Shwartz, S. and Ben-David, S. (2014). *Understanding machine learning: From theory to algorithms*. Cambridge university press.

[Smith and Waterman, 1981] Smith, T. and Waterman, M. (1981). Identification of common molecular subsequence. *J Mol. Biol*, 147.

[Steinhardt et al., 2018] Steinhardt, J., Charikar, M., and Valiant, G. (2018). Resilience: A criterion for learning in the presence of arbitrary outliers. In *9th Innovations in Theoretical Computer Science Conference, ITCS 2018, January 11-14, 2018, Cambridge, MA, USA*, pages 45:1–45:21.

[van Dongen, 2000] van Dongen, S. (2000). *Graph Clustering by Flow Simulation*. PhD thesis, University of Utrecht.

[Vapnik and Chervonenkis, 2015] Vapnik, V. N. and Chervonenkis, A. Y. (2015). On the uniform convergence of relative frequencies of events to their probabilities. In *Measures of Complexity*, pages 11–30. Springer.

[Vattani, 2009] Vattani, A. (2009). The hardness of k-means clustering in the plane. *Manuscript, accessible at http://cseweb. ucsd. edu/avattani/papers/kmeans_hardness. pdf*, 617.

[Vershynin, 2010] Vershynin, R. (2010). Introduction to the non-asymptotic analysis of random matrices. *arXiv preprint arXiv:1011.3027*.

[Yang et al., 2015] Yang, L., Sun, D., and Toh, K.-C. (2015). Sdpnal+: a majorized semismooth newton-cg augmented lagrangian method for semidefinite programming with nonnegative constraints. *Mathematical Programming Computation*, 7(3):331–366.

APPENDICES

Appendix A

Clustering with same-cluster queries

A.1 Relationships Between Query Models

Proposition A.1. *Any clustering algorithm that uses only q same-cluster queries can be adjusted to use $2q$ cluster-assignment queries (and no same-cluster queries) with the same order of time complexity.*

Proof. We can replace each same-cluster query with two cluster-assignment queries as in $Q(x_1, x_2) = \mathbb{1}\{Q(x_1) = Q(x_2))\}$. □

Proposition A.2. *Any algorithm that uses only q cluster-assignment queries can be adjusted to use kq same-cluster queries (and no cluster-assignment queries) with at most a factor k increase in computational complexity, where k is the number of clusters.*

Proof. If the clustering algorithm has access to an instance from each of k clusters (say $x_i \in X_i$), then it can simply simulate the cluster-assignment query by making k same-cluster queries ($Q(x) = \arg\max_i \mathbb{1}\{Q(x, x_i)\}$). Otherwise, assume that at the time of querying $Q(x)$ it has only instances from $k' < k$ clusters. In this case, the algorithm can do the same with the k' instances and if it does not find the cluster, assign x to a new cluster index. This will work, because in the clustering task the output of the algorithm is a partition of the elements, and therefore the indices of the clusters do not matter. □

Table A.1: Known results for α-center proximity

	Euclidean	General Metric
Centers from data	Upper bound : $\sqrt{2}+1$ [Balcan and Liang, 2012] Lower bound : ?	Upper bound : $\sqrt{2}+1$ [Balcan and Liang, 2012] Lower bound : 2 [Ben-David and Reyzin, 2014]
Unrestricted Centers	Upper bound : $2+\sqrt{3}$ [Awasthi et al., 2012] Lower bound : ?	Upper bound : $2+\sqrt{3}$ [Awasthi et al., 2012] Lower bound : 3 [Awasthi et al., 2012]

A.2 Comparison of γ-Margin and α-Center Proximity

In this paper, we introduced the notion of γ-margin niceness property. We further showed upper and lower bounds on the computational complexity of clustering under this assumption. It is therefore important to compare this notion with other previously-studied clusterability notions.

An important notion of niceness of data for clustering is α-center proximity property.

Definition A.3 (α-center proximity [Awasthi et al., 2012])**.** *Let $(\mathcal{X}, d)$ be a clustering instance in some metric space M, and let k be the number of clusters. We say that a center-based clustering $\mathcal{C} = \{C_1, \ldots, C_k\}$ induced by centers $c_1, \ldots, c_k \in M$ satisfies the α-center proximity property (with respect to $\mathcal{X}$ and k) if the following holds*

$$\forall x \in C_i, i \neq j, \alpha d(x, c_i) < d(x, c_j)$$

This property has been considered in the past in various studies [Balcan and Liang, 2012, Awasthi et al., 2012]. In this appendix we will show some connections between γ-margin and α-center proximity properties.

It is important to note that throughout this paper we considered clustering in Euclidean spaces. Furthermore, the centers were not restricted to be selected from the data points. However, this is not necessarily the case in other studies.

An overview of the known results under α-center proximity is provided in Table A.1. The results are provided for the case that the centers are restricted to be selected from the training set, and also the unrestricted case (where the centers can be arbitrary points

Table A.2: Results for γ-margin

	Euclidean	General Metric
Centers from data	Upper bound : 2 (Thm. A.4) Lower bound : ?	Upper bound : 2 (Thm. A.4) Lower bound : 2 (Thm. A.5)
Unrestricted Centers	Upper bound : 3 (Thm. A.6) Lower bound : 1.84 (Thm. 2.8)	Upper bound : 3 (Thm. A.6) Lower bound : 3 (Thm. A.7)

from the metric space). Note that any upper bound that works for general metric spaces also works for the Euclidean space.

We will show that using the same techniques one can prove upper and lower bounds for γ-margin property. It is important to note that for γ-margin property, in some cases the upper and lower bounds match. Hence, there is no hope to further improve those bounds unless P=NP. A summary of our results is provided in A.2.

A.2.1 Centers from data

Theorem A.4. *Let (X, d) be a clustering instance and $\gamma \geq 2$. Then, Algorithm 1 in [Balcan and Liang, 2012] outputs a tree $\mathcal{T}$ with the following property:*

Any k-clustering $\mathcal{C}^* = \{C_1^*, \ldots, C_k^*\}$ which satisfies the γ-margin property and its cluster centers $\mu_1, \ldots, \mu_k$ are in X, is a pruning of the tree T. In other words, for every $1 \leq i \leq k$, there exists a node N_i in the tree T such that $C_i^* = N_i$.

Proof. Let $p, p' \in C_i^*$ and $q \in C_j^*$. [Balcan and Liang, 2012] prove the correctness of their algorithm for $\alpha > \sqrt{2} + 1$. Their proof relies only on the following three properties which are implied when $\alpha > \sqrt{2} + 1$. We will show that these properties are implied by $\gamma > 2$ instances as well.

- $d(p, \mu_i) < d(p, q)$
  $\gamma d(p, \mu_i) < d(q, \mu_i) < d(p, q) + d(p, \mu_i) \implies d(p, \mu_i) < \frac{1}{\gamma - 1} d(p, q)$.
- $d(p, \mu_i) < d(q, \mu_i)$
  This is trivially true since $\gamma > 2$.
- $d(p, \mu_i) < d(p', q)$
  Let $r = \max_{x \in C_i^*} d(x, \mu_i)$. Observe that $d(p, \mu_i) < r$. Also, $d(p', q) > d(q, \mu_i) - d(p', \mu_i) > \gamma r - r = (\gamma - 1) r$.

□

Theorem A.5. *Let $(\mathcal{X}, d)$ be a clustering instance and k be the number of clusters. For $\gamma < 2$, finding a k-clustering of X which satisfies the γ-margin property and where the corresponding centers $\mu_1, \ldots, \mu_k$ belong to $\mathcal{X}$ is NP-hard.*

Proof. For $\alpha < 2$, [Ben-David and Reyzin, 2014] proved that in general metric spaces, finding a clustering which satisfies the α-center proximity and where the centers $\mu_1, \ldots, \mu_k \in \mathcal{X}$ is NP-hard. Note that the reduced instance in their proof, also satisfies γ-margin for $\gamma < 2$. □

A.2.2 Centers from metric space

Theorem A.6. *Let (X, d) be a clustering instance and $\gamma \geq 3$. Then, the standard single-linkage algorithm outputs a tree $\mathcal{T}$ with the following property:*

Any k-clustering $\mathcal{C}^* = \{C_1^*, \ldots, C_k^*\}$ which satisfies the γ-margin property is a pruning of T. In other words, for every $1 \leq i \leq k$, there exists a node N_i in the tree T such that $C_i^* = N_i$.

Proof. [Balcan et al., 2008] showed that if a clustering C^* has the strong stability property, then single-linkage outputs a tree with the required property. It is simple to see that if $\gamma > 3$ then instances have strong-stability and the claim follows. □

Theorem A.7. *Let $(\mathcal{X}, d)$ be a clustering instance and $\gamma < 3$. Then, finding a k-clustering of X which satisfies the γ-margin is NP-hard.*

Proof. [Awasthi et al., 2012] proved the above claim but for $\alpha < 3$ instances. Note however that the construction in their proof satisfies γ-margin for $\gamma < 3$. □

Appendix B

Semi-supervised clustering for deduplication

B.1 Locality Sensitive Hashing

The technique of locality sensitive hashing was introduced by [Gionis et al., 1999] to solve the problem of approximate nearest neighbour search. The LSH-based hashing schemes try to put similar points into the same bucket. Hence, to search for points which are 'similar' to a given point $x \in X$, one only needs to search within the same hash bucket instead of searching within the whole set X. Next, we describe a generic hashing based algorithm.

Definition B.1 (LSH [Charikar, 2002])**.** *Given a set X and a similarity function $f : X \times X \to [0, 1]$. Given a class of hash functions $H = \{h_1, h_2, \ldots\}$. An LSH w.r.t X and f is a probability distribution over H such that for each $x, y \in \mathcal{X}$,*

$$\mathop{\mathbf{P}}_{h \in H} [h(x) = h(y)] = f(x, y)$$

Some common examples of an LSH schemes are minhash scheme w.r.t jaccard similarity measure [Broder et al., 2000, Broder, 1997], simhash scheme w.r.t hamming similarity measure [Charikar, 2002]. We describe a generic locality sensitive hashing procedure in Alg. 12. Observe that, Alg. 12 outputs s different partitions of X. Assume without loss of generality, that a point x lies in the blocks $B_{11} \in P_1, B_{21} \in P_2, \ldots, B_{s1} \in P_s$. Now, to search for points y which are similar to x, we only need to search within the blocks B_{i1}. We say that these points lie in the same 'bucket' as x. We say that $b(x, y) = 1$ if and only if x and y lie together in the same block in at least one of the partitions.

Algorithm 12: A generic LSH based hashing algorithm [Indyk and Motwani, 1998, Charikar, 2002]

Input: A set X, a similarity function f, a class of hash functions H over X, integers r, s and a perfect hash function p over the domain $\mathbf{N}^r$.

Output: Partitions $P_1, \ldots, P_s$ of the set X.

1. Let D be a distribution over H which satisfies Defn. B.1 and let $k = rs$.
2. Sample hash functions $h_1, \ldots, h_k$ iid using D.
3. Group the hash functions into s bands. Each band contains r hash functions.
4. For all x and $1 \leq i \leq s$, let $g_i(x) = (h_{(i-1)s+1}(x), \ldots, h_{ir}(x))$. That is, $g_i(x)$ represents the i^{th} signature of x.
5. For all $1 \leq i \leq s$, obtain partitions P_i of $\mathcal{X}$. That is, $P_i = \{B_{i1}, \ldots, B_{im_i}\}$ where each $B_{ij} = \{x : p(g_i(x)) = b_{ij}\}$ for some b_{ij}.
6. Output $\{P_1, \ldots, P_s\}$.

Hence, to search for points which are similar to x, we need to search over y such that $b(x, y) = 1$. Our sampling procedure will sample pairs from the set $\mathcal{Q} := \{(x, y) : b(x, y) = 1\}$ (details in the next section). Hence, a requirement from the hashing scheme is that we should be able to construct the set $\mathcal{Q}$ in linear time. Thm. B.2 shows that this is indeed the case.

Theorem B.2. *Given X, a similarity function f, a class of hash functions H, integers r, s and perfect hash function p. Alg. 12 runs in $O(|X| rs \max_{ij} |B_{ij}|)$.*

Proof. Let $n = |X|$. Sampling k different hash function takes rs time. Computing the signatures for all x takes nrs time. Obtaining the different partition takes ns time. Now, computing R_x for all x takes time $t = \sum_{i=1}^{b} \sum_{j=1}^{m_j} |B_{ij}|^2$. Now, we know that for all i, $\sum_{j=1}^{m_j} |B_{ij}| = n$. Hence, $\sum_{j=1}^{m_j} |B_{ij}|^2 \leq \max_j B_{ij} \sum_{j=1}^{m_j} |B_{ij}| = n \max_j B_{ij}$. Hence, $t \leq ns \max_{ij} |B_{ij}|$. □

We see that the running time is dependent upon the block sizes. If the maximum block size is a constant then the hashing scheme runs in linear time. Even when the block sizes are bounded by $\log n$, then the scheme runs in $O(n \log n)$. Next, we show that a if $d(x, y) \leq \lambda$ then $(x, y) \in \mathcal{Q}$ with very high probability. Also, if $d(x, y) > O(2\lambda)$ then with high probability $(x, y) \notin \mathcal{Q}$.

Theorem B.3. *Given a set X, a distance function $d : X \times X \to [0, 1]$, a class of hash functions H, threshold parameter λ and a parameter δ. Let the similarity function be*

$f(x,y) := 1 - d(x,y)$ and let $\mathcal{A}$ be a generic LSH based algorithm (Alg. 12) which outputs partitions $P_1, \ldots, P_s$. Let $b(x,y) = 1$ iff x, y are together in at least one of these partitions. Choose r, s such that $\frac{1}{2\lambda} < r < \frac{1}{-\ln(1-\lambda)}$ and $s = \lceil 2.2 \ln(\frac{1}{\delta}) \rceil$. Define $\delta' := s \ln(1+\delta)$. Then for $(x,y) \in \mathcal{X}^2$

- *If $d(x,y) \leq \lambda$ then*

$$\mathop{\mathbf{P}}_{h \in H} \left[\; b(x,y) = 1 \right] \;>\; 1 - \delta$$

- *If $d(x,y) > 2\lambda \ln\left(1 + \frac{1}{\delta}\right)$ then*

$$\mathop{\mathbf{P}}_{h \in H} \left[\; b(x,y) = 1 \right] \;<\; \delta'$$

Proof. Observe that

$$\begin{aligned}
\mathbf{P}[b(x,y) = 0] &= \mathbf{P}\ [\textstyle\bigcap_{i=1}^{s} g_i(x) \neq g_i(y)] \\
&= \prod_i \left(1 - \prod_{j=1}^{r} \mathbf{P}[h_{(i-1)r+j}(x) = h_{(i-1)r+j}(y)]\right) \\
&= \prod_{i=1}^{s} (1 - \prod_{j=1}^{r} f(x,y)) = (1 - f(x,y)^r)^s
\end{aligned}$$

Consider the case when $d(x,y) \leq \lambda$. From the choice of s, we know that $s \geq 2.2 \ln(1/\delta) \implies s \geq \frac{\ln(1/\delta)}{1-\ln(e-1)} \iff 1 - \frac{1}{e} \leq \delta^{1/s}$. From the choice of r, we know that $r < \frac{1}{-\ln(1-\lambda)} \iff r \ln(\frac{1}{1-\lambda}) < 1 \iff (1-\lambda)^r > \frac{1}{e}$. Hence, then we have that $\mathbf{P}[b(x,y) = 0] = (1 - (1 - d(x,y))^r)^s \leq (1 - (1-\lambda)^r)^s < \delta$. This proves the first part of the theorem.

For the second part, consider the case when $d(x,y) > \lambda'$ where λ' is such that $\ln(1 + 1/\delta) = \frac{\lambda'}{2\lambda}$.

$\frac{1}{2\lambda} < r \iff \frac{\ln(1+1/\delta)}{\lambda'} < r$. Now, $\lambda' \leq \ln\left(\frac{1}{1-\lambda'}\right)$. Hence,

$$\begin{aligned}
&\implies \frac{\ln(1+1/\delta)}{\ln(\frac{1}{1-\lambda'})} < r \iff r \ln(1-\lambda') < \ln(\frac{\delta}{1+\delta}) \\
&\iff 1 - (1-\lambda')^r > e^{-\ln(1+\delta)} = e^{-\delta'/s} > (1-\delta')^{1/s} \\
&\implies \mathbf{P}[b(x,y) = 0] > (1 - (1-\lambda')^r)^s > 1 - \delta'
\end{aligned}$$

Now, the only thing that remains to be shown is that we can choose an integer r satisfying the conditions of the theorem. Consider the function $f(x) = -\frac{1}{2x} - \frac{1}{\ln(1-x)}$. Using elementary analysis, we see that for $x \to 0, f(x) \to \infty$. Infact, for $x \leq 0.32, f(x) > 1$. Hence, for $\lambda \leq 0.32$, r satisfying the conditions of the theorem exists. □

B.2 Sample and query complexity of RCC using $\mathcal{P}_{12}$

Theorem B.4. *Given metric space (X, d), a class of clusterings $\mathcal{F}$ of X and a threshold parameter λ. Given $\epsilon, \delta \in (0, 1)$ and a C^*-oracle. Let d be (α, β)-informative w.r.t C^* and λ and let X satisfy γ-skewed property. Let $\mathcal{A}$ be the ERM-based approach as described in Alg. 5 (where the LSH-based scheme is ζ-tight) and $\hat{C}$ be the output of $\mathcal{A}$. If*

$$m_-, m_+ \geq a\frac{\text{VC-Dim}(\mathcal{F}) + \log(\frac{2}{\delta})}{\epsilon^2} \tag{B.1}$$

where a is a global constant then with probability at least $1 - \delta - \exp\left(\frac{-2\nu^2(1-\alpha)^2|X_2^+|^2}{49}\right)$ (over the randomness in $\mathcal{A}$), we have that

$$L_{C^*}(\hat{C}) \leq \min_{C \in \mathcal{F}} L_{C^*}(C) + 3\alpha + \zeta + \epsilon + \nu$$

Proof. Let T_0 be the distribution induced by $\mathcal{P}_0$ and T_1 be the distribution induced by $\mathcal{P}_1$. Using Thm. 1.2, we know that if $m_+ > a\frac{\text{VC-Dim}(\mathcal{F})+\log(\frac{1}{\delta})}{\epsilon^2}$ then with probability at least $1 - \delta - e^{-2\nu^2(1-\alpha)^2|X_2^+|^2}$, we have that for all C

$$\begin{aligned} &|\hat{P}(C) - \mathop{\mathbf{P}}_{(x,y)\sim T_1}[C(x, y) = 0]| \leq \epsilon \\ \implies &\hat{P}(C) \leq \epsilon + (1 + 2\nu)(1 + 2\alpha)L_{P^+}(C) \quad \text{and} \\ &L_{P^+}(C) - \epsilon - \alpha - \zeta - \nu \leq \hat{P}(C) \end{aligned} \tag{B.2}$$

Note that we obtain upper and lower bounds for $\mathop{\mathbf{P}}_{(x,y)\sim T_1}[h(x, y) = 0]$ using Thm. 3.20. Similarly, if $m_- > a\frac{\text{VC-Dim}(\mathcal{F})+\log(\frac{1}{\delta})}{\epsilon^2}$, then with probability at least $1 - \delta$, we have that for all h,

$$\begin{aligned} &|\hat{N}(C) - \mathop{\mathbf{P}}_{(x,y)\sim T_0}[C(x, y) = 1]| \leq \epsilon \\ \implies &\hat{N}(C) \leq \epsilon + L_{P^-}(C) \quad \text{and } L_{P^-}(C) - \epsilon \leq \hat{N}(C) \end{aligned} \tag{B.3}$$

Combining Eqns. B.2 and B.3, we get that with probability at least $1-2\delta-e^{-2\nu^2(1-\alpha)^2|X_2^+|^2}$, we have that for all $C \in \mathcal{F}$

$$\begin{aligned}
\hat{L}(C) &\leq \mu[\epsilon + (1+2\nu)(1+2\alpha)L_{P^+}(C)] \\
&\quad + (1-\mu)(\epsilon + L_{P^-}(C)) \\
&\leq L_{C^*}(C) + \epsilon + 2\alpha + 2\nu + 4\alpha\nu \\
\text{And } \hat{L}(C) &\geq \mu(L_{P^+}(h) - \epsilon - \alpha - \nu - \zeta) + (1-\mu)(L_{P^-}(h) - \epsilon) \\
&\geq L_{C^*}(C) - \epsilon - \alpha - \nu - \zeta
\end{aligned}$$

Now, let $\hat{C}$ be the output of $\mathcal{A}$ and let $\hat{C}^*$ be $\arg\min_{C\in\mathcal{F}} L_{C^*}(C)$. Then, we have that

$$\begin{aligned}
L_{C^*}(\hat{C}) &\leq \hat{L}(\hat{C}) + \alpha + \epsilon + \nu + \zeta \leq \hat{L}(\hat{C}^*) + \alpha + \epsilon + \nu + \zeta \\
&\leq L_{C^*}(\hat{C}^*) + 2\epsilon + 3\alpha + 3\nu + 4\alpha\nu + \zeta
\end{aligned}$$

Choosing $\epsilon = \frac{\epsilon}{2}$ and $\delta = \frac{\delta}{2}$ and $\nu = \frac{\nu}{7}$ throughout gives the result of the theorem, and this completes the proof of the theorem. □

Finally, we analyse the query complexity of our approach. That is the number of queries that our algorithm makes to the C^*-oracle. Our algorithm makes queries during the sampling procedure. We see that to sample m_- negative and m_+ positive pairs the number of queries is 'close' to $m_+ + m_-$ with very high probability. Thus, the number of 'wasted' queries is small.

Theorem B.5. *[Query Complexity] Let the framework be as in Thm. B.4. With probability at least* $1 - \exp\big(-\frac{\nu^2 m_-}{4}\big) - \exp\big(-\frac{\nu^2 m_+}{4}\big) - \exp(-\nu^2(1-\alpha)^2|\mathcal{X}_+^2|^2)$ *over the randomness in the sampling procedure, the number of same-cluster queries* q *made by* $\mathcal{A}$ *is*

$$q \leq (1+\nu)\left(\frac{m_-}{(1-\gamma)} + \frac{m_+}{\beta(1-\zeta-\nu)}\right)$$

Proof. The number of queries made to sample the set S_g is $q_g = m_g$. Let q_+ denote the number queries to sample the set S_+. Using Lemma 3.21, we know that $\mathbf{E}[q_+] \leq \frac{1}{\beta(1-\zeta-\nu)}$ with probability at least $1 - \exp(-\nu^2(1-\alpha)^2|\mathcal{X}_+^2|^2)$ over the randomness in the hashing procedure. Given that the expectation is bounded as above, using Thm. 1.3, we get that $q_+ \leq \frac{(1+\nu)m_+}{\beta(1-\zeta-\nu)}$ with probability at least $1 - \exp(\frac{-\nu^2 m_+}{4})$. Similarly, combining Lemma 3.16 with Thm. 1.3, we get that with probability at least $1 - \exp(\frac{-\nu^2 m_-}{4})$, $q_- \leq \frac{(1+\nu)m_-}{(1-\gamma)}$. □

Appendix C

Noise-robust k-means clustering

C.1 Hardness of regularized k-means

Theorem C.1. *Given a clustering instance $X \subset \mathbf{R}^p$, define $m(X) := \min_{x \neq y \in X} \|x - y\|_2^2$ and $n := |X|$. Finding the optimal solution to the regularized 1-means objective is NP-hard for $\frac{m(X)}{2} < \lambda < \frac{m(X)}{2} + \frac{1}{2n^2(n-1)}$.*

In particular, the optimization problem is NP-hard for $\lambda = \lambda_0(X) := \frac{m(X)}{2} + \frac{1}{4n^3}$.

Proof. The proof uses reduction from the clique problem. Given a graph $G = (V, E)$ and an integer q, the clique problem asks the following: does there exist a clique in G of size at least q?

Given an instance of the clique problem, we construct an instance of regularized 1-means as follows. For every $v \in V$, construct $x_v \in X$. Define the metric as

$$d^2(x_i, x_j) = \begin{cases} 1 & \text{if } (i,j) \in E \\ 1 + \Delta & \text{if } (i,j) \notin E \end{cases} \tag{C.1}$$

where $0 < n\Delta < 1$. Now, we will show that G has a clique of size $\geq q \iff X$ has a clustering of cost $\leq c = \frac{q-1}{2} + \lambda(n - q)$.

$\implies$ Assume G has a clique of size at least q. Assign all points in the clique to C_1 and the remaining points to C_2. This clustering has cost as desired.

$\impliedby$ Let $|C_1| = n'$. Now, there are three possibilities.

Case 1: $n' > q$. If all the distances in C_1 are 1, then the cost of the clustering is $\frac{n'-1}{2} + \lambda(n-n') \leq c$ as $\lambda > \frac{1}{2}$. Thus, the vertices corresponding to the points in C_1 form a clique of size $n' > q$. If at least one distance is $1+\Delta$ then the cost of the clustering is

$$\frac{n'-1}{2} + \lambda(n-n') + \frac{\Delta}{n'} \leq c \implies \lambda \geq \frac{1}{2} + \frac{\Delta}{n'(n'-q)}$$

This is a contradiction because of the choice of λ.

Case 2: $n' < q$. If all the distances in $C_1 = 1$, then the cost of the clustering is $\frac{n'-1}{2} + \lambda(n-n') > c$. If atleast one distance is $1+\Delta$ then the cost of the clustering is even greater as $\lambda > \frac{1}{2}$.

Case 3: $n' = q$. If atleast one distance is $1+\Delta$ then the cost of the clustering is $\frac{q-1}{2} + \lambda(n-q) + \frac{\Delta}{m} > c$. Hence, the only possibility remains that $|C_1| = q$ and all the distances in $C_1 = 1$. Hence, G has a clique of size q. □

Corollary C.2 ([Dasgupta, 2008] Cor. 7)**.** *An $N \times N$ symmetric matrix D can be embedded in $\mathbf{R}^N$ if and only if $u^T D u \leq 0$ for all $u \in R^N$ with $u.\mathbf{1} = 0$*

Lemma C.3. *Let d be as in Eqn. C.1. Then d can be embedded into $\mathbf{R}^{|X|}$.*

Proof. The proof of embedding is very similar to Thm. 8 in [Dasgupta, 2008]. Using Cor. C.2, we know that D can be embedded into $\mathbf{R}^{|X|}$ if and only if $u^T D u \leq 0$ for all $u^T 1 = 0$.

$$\begin{aligned} u^T D u &= \sum_{ij} u_i d_{ij} u_j = \sum_{ij} u_i u_j \Big(1 - \mathbf{1}(i=j) + \Delta \mathbf{1}(i \not\sim_E j)\Big) \\ &\leq \Big(\sum_i u_i\Big)^2 - \sum_i u_i^2 + \Delta \sum_{ij} |u_i||u_j| \leq -\|u\|^2 + |X|\Delta\|u\|^2 \\ &= -(1 - |X|\Delta)\|u\|^2, \end{aligned}$$

which completes our proof. □

We now use Thm. C.1 to show that regularized 1-means is Np-Hard for any $\lambda > m(X)/2$.

Theorem C.4. *For all λ there exists a clustering instance $X \subset \mathbf{R}^p$ such that $\lambda > \frac{m(X)}{2}$ but finding the optimal solution to the regularized 1-means objective is NP-hard.*

Proof. Let (X, d) be any clustering instance. Thm. C.1 showed that optimizing the problem

$$\min_{C \subseteq X} \sum_{x \in C} d^2(x, c) + \lambda_0(X) \ |X \setminus C| \qquad \text{(FRM)}$$

is NP-hard when $\lambda_0(X) := \frac{m(X)}{2} + \frac{1}{4n^3}$. Given $\lambda > 0$, we want to show that for all $\lambda > m(X)/2$, the following is also NP-hard to optimize:

$$\min_{C \subseteq X} \sum_{x \in C} d^2(x, c) + \lambda \ |X \setminus C| \qquad \text{(GRM)}$$

Let $\lambda' = \sqrt{n^3(4\lambda - 2m(X))} > 0$, and define a new Euclidean distance $d'(x, y) = \frac{d(x,y)}{\lambda'}$. Then

$$\begin{aligned}\sum_{x \in C} d^2(x, c) + \lambda \ |X \setminus C| &= \lambda'^2 \left(\sum_{C} d'^2(x, c) + \frac{|X \setminus C|}{\lambda'^2} \Big(\frac{m(X)}{2} + \frac{\lambda'^2}{4n^3} \Big) \right) \\ &= \lambda'^2 \Big(\sum_{C} d'^2(x, c) + |X \setminus C| \lambda_0(X') \Big).\end{aligned}$$

where X' is the clustering instance X but with distance function d' instead of d. Hence, we see that GRM is equivalent to FRM which is NP-hard. □

C.2 The regularized k-means algorithm

Equivalence of regularized k-means with $0, 1$-SDP

We follow a similar technique to that of [Peng and Wei, 2007] to translate equation 5.4 into a 0-1 SDP. We are given a set X with n data points. The goal is to partition the set into k clusters with the option of throwing some points into the garbage $k+1$ cluster. Let S be an assignment matrix of size $n \times k$ and y be an $n \times 1$ column vector that assigns points to the "garbage" cluster.

$$s_{ij} = \begin{cases} 1 \ \text{iff} \ x_i \in C_j, j \leq k \\ 0 \ \text{otherwise} \end{cases} \qquad y_i = \begin{cases} 1 \ \text{iff} \ x_i \in C_{k+1} \\ 0 \ \text{otherwise} \end{cases}$$

Provided that $C_i \neq \emptyset$, we have the following equality by expanding the mean $c_i = \frac{1}{|C_i|}\sum_{x \in C_i} x$:

$$\sum_{x \in C_i} \|x - c_i\|^2 = \frac{1}{2|C_i|} \sum_{x,y \in C_i} \|x - y\|^2 \tag{C.2}$$

$c_j = \frac{\sum_i s_{ij} x_i}{\sum_i s_{ij}}$ is the average of points in the j^{th} cluster. Hence,

$$\begin{aligned}
&\sum_j \sum_i s_{ij}(\langle c_j, c_j\rangle - 2\langle x_i, c_j\rangle) = \sum_j \langle \sum_i s_{ij} c_j,\ \ c_j\rangle - 2\sum_j \langle \sum_i s_{ij} x_i,\ \ c_j\rangle \\
&= \sum_j \langle c_j, \sum_i s_{ij} x_i\rangle - 2\sum_j \langle \sum_i s_{ij} x_i,\ \ c_j\rangle = -\sum_j \langle \frac{\sum_i s_{ij} x_i}{\sum_i s_{ij}}, \sum_i s_{ij} x_i\rangle = -\sum_j \frac{\|\sum_i s_{ij} x_i\|^2}{\sum_i s_{ij}}
\end{aligned} \tag{C.3}$$

Combining equations C.2 and C.3, we get that

$$\begin{aligned}
\sum_{j=1}^{k} \sum_{x \in C_i} \|x - c_i\|^2 &= \sum_{i=1}^{n} \|x_i\|^2 (1 - y_i) - \sum_{j=1}^{k} \frac{\|\sum_i s_{ij} x_i\|^2}{\sum_i s_{ij}} \\
&= \sum_{i=1}^{n} \|x_i\|^2 (1 - y_i) - \sum_{i=1}^{n}\sum_{j=1}^{n} \langle x_i, x_j\rangle Z_{ij}
\end{aligned} \tag{C.4}$$

where $Z = S(S^T S)^{-1} S^T$. Observe that if $x_i \notin C_{k+1}$ then $Z_{ij} = \frac{1}{|S_{c(i)}|}\langle s_i, s_j\rangle$ where $S_{c(i)}$ denotes the size of the i^{th} cluster. If $x_i \in C_{k+1}$ then $Z_{ij} = 0$. Thus,

$$Z_{ij} = \begin{cases} \dfrac{1}{|S_{c(i)}|}\langle s_i, s_j\rangle & \text{if } y_i = 0 \\ 0 & \text{otherwise} \end{cases}$$

Observe that $Z_{ij} = Z_{ji}$ and

$$\text{Tr}(Z) = \sum_i Z_{ii} = \sum_{x \notin C_{k+1}} \frac{1}{|S_{c(i)}|} = k.$$

Also, we have that and $\langle Z_i, \mathbf{1}\rangle = \sum_j Z_{ij}$. If $y_i = 0$ then $\sum_j Z_{ij} = 0$ else $\sum_j Z_{ij} = \frac{1}{|S_{c(i)}|}\sum_j \langle s_i, s_j\rangle = 1$. Hence, we get that $\langle Z_i, \mathbf{1}\rangle = \sum_j Z_{ij} = 1 - y_i$, or equivalently, $Z \cdot \mathbf{1} + y = \mathbf{1}$. Also, it is fairly easy to see that $Z^2 = Z$.

Let D be a matrix such that $D_{ij} = d^2(x_i, x_j)$. Using the above properties of Z, we get that

$$\mathrm{Tr}(DZ) = \sum_{ij} \langle x_i - x_j, x_i - x_j \rangle z_{ij} = \sum_i \|x_i\|^2 \sum_j z_{ij} + \sum_j \|x_j\|^2 \sum_i z_{ij} - 2 \sum_{ij} \langle x_i, x_j \rangle z_{ij}$$

$$= 2\Big(\sum_i \|x_i\|^2 (1 - y_i) - \sum_{ij} \langle x_i, x_j \rangle z_{ij} \Big) = 2 \sum_{i=1}^{k} \sum_{x \in C_i} \|x - c_i\|^2 \text{ (using Eqn. C.4)}$$

Finally, observe that $\lambda |C_{k+1}| = \lambda \langle \mathbf{1}, y \rangle$.

$$\textbf{0-1 SDP} \begin{cases} \min_{Z,y} & \mathrm{Tr}(DZ) + \lambda \langle \mathbf{1}, y \rangle \\ \text{s.t.} & \mathrm{Tr}(Z) = k \\ & Z \cdot \mathbf{1} + y = \mathbf{1} \\ & Z \geq 0, Z^2 = Z, Z^T = Z \\ & y \in \{0,1\}^n \end{cases} \xrightarrow{\text{relaxed}} \textbf{SDP} \begin{cases} \min_{Z,y} & \mathrm{Tr}(DZ) + \lambda \langle \mathbf{1}, y \rangle \\ \text{s.t.} & \mathrm{Tr}(Z) = k \\ & \left(\frac{Z + Z^T}{2} \right) \cdot \mathbf{1} + y = \mathbf{1} \\ & Z \geq 0, y \geq 0, Z \succeq 0 \end{cases} \tag{C.5}$$

Theorem C.5. *Finding a solution to the 0-1 SDP (5.6) is equivalent to finding a solution to the regularized k-means objective (5.4).*

Proof. We will use the same proof ideas as in the proof of Thm 2.2 in [Peng and Wei, 2007]. However, we need to modify the proof slightly according to our formulation. From the discussion in the previous subsection, we can see that any solution for (5.4) implies a solution for the 0-1 SDP (5.6) with same cost. Now, we will prove the other direction. Any solution for the 0-1 SDP implies a solution for for (5.4) with same cost.

Let e_i be a vector with all zeros except in the i^{th} index. Observe that $u_i^T Z u_i \geq 0$. Hence, $Z_{ii} \geq 0$. Similarly, $(e_i - e_j)^T Z (e_i - e_j) = Z_{ii} - 2Z_{ij} + Z_{jj}$. Hence, $Z_{ij} \leq \max(Z_{ii}, Z_{jj})$. Let $Z_{i^*i^*} = \max_i Z_{ii}$. Hence, we have that for all i, j, $Z_{ij} \leq Z_{i^*i^*}$.

Suppose $Z_{i^*i^*} = 0$. Then, $Z = \mathbf{0}$ and $y = \mathbf{1}$. This implies that all points are assigned to the $k+1$ cluster. The cost of both the solutions in this case is $\lambda |C_{k+1}|$.

Now, suppose $Z_{i^*i^*} > 0$. Let $I = \{j : Z_{i^*j} > 0\}$. Since, $Z^2 = Z$, we have that $Z_{i^*i^*} = \sum_{j=1}^n Z_{i^*j}^2 = \sum_{j \in I} Z_{i^*j}^2$. Hence, $\sum_{j \in I} \frac{Z_{i^*j}}{Z_{i^*i^*}} Z_{i^*,j} = 1$. Also, we have that $\sum_j Z_{i^*j} + y_{i^*} = 1$.

If $y_{i^*} = 1$, then $\sum_j Z_{i^*j} = 0$ which contradicts our assumption that $Z_{i^*i^*} > 0$. Hence, $y_{i^*} = 0$ and we have $\sum_j Z_{i^*j} = \sum_{j \in I} Z_{i^*j} = 1$. Since we have the following constraints,

$$\sum_{j \in I} Z_{i^*j} = 1 \qquad \text{and} \qquad \sum_{j \in I} \frac{Z_{i^*j}}{Z_{i^*i^*}} Z_{i^*j} = 1$$

$Z_{i^*j} = Z_{i^*i^*}$ for all $j \in I$. Hence, we see that the matrix Z and the vector y can be decomposed as

$$Z = \begin{bmatrix} Z_{II} & 0 \\ 0 & Z' \end{bmatrix} \quad y = \begin{bmatrix} 0 \\ y' \end{bmatrix}$$

where $Z_{II} = \frac{1}{|I|} \mathbf{1}_{|I|} \mathbf{1}_{|I|}^T$. Now, we can see that $\mathrm{Tr}(Z') = k - 1$ and

$$Z\mathbf{1} + a = \begin{bmatrix} 1 \\ Z'\mathbf{1} \end{bmatrix} + \begin{bmatrix} 0 \\ a' \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}.$$

This implies that $Z' \cdot \mathbf{1} + y' = \mathbf{1}$. Hence, the optimization problem now reduces to

$$\begin{cases} \min_{Z', y'} & \mathrm{Tr}(DZ') + \lambda \langle \cdot \mathbf{1}, y' \rangle \\ \text{subject to} & \mathrm{Tr}(Z') = k - 1 \\ & Z' \cdot \mathbf{1} + y' = \mathbf{1} \end{cases}$$

Repeating this process k times, we get that Z can be decomposed into k non-zero block diagonal matrices and one zero block diagonal matrix. Hence, using this we construct a solution for the original clustering problem as follows. For all i, if the row Z_i belongs to the j^{th} diagonal block then x_i is assigned to C_i. Given Z and a, the cost of the 0-1 SDP solution is

$$\frac{1}{2}\mathrm{Tr}(DZ) + \lambda \langle \mathbf{1}, y \rangle = \sum_{i=1}^{k} \sum_{x,y \in C_i} \frac{\|x_i - x_j\|^2}{2|C_i|} + \lambda |C_{k+1}| = \sum_{i=1}^{k} \sum_{x \in C_i} \|x - c_i\|^2 + \lambda |C_{k+1}|$$

which is the same as the cost of the regularized k-means objective. Hence, from a feasible solution of (5.4), we can obtain a feasible solution of the 0-1 SDP of same cost. □

C.3 Tightness of the SDP based algorithm

$$\textbf{0-1 SDP}\begin{cases}\min_Z & \mathrm{Tr}(DZ)\\ \text{s.t.} & \mathrm{Tr}(Z)=k\\ & Z\cdot\mathbf{1}=\mathbf{1}\\ & Z\geq 0, Z^2=Z, Z^T=Z\end{cases}\xrightarrow{\text{relaxed}}\textbf{SDP}\begin{cases}\min_Z & \mathrm{Tr}(DZ)\\ \text{s.t.} & \mathrm{Tr}(Z)=k\\ & \left(\frac{Z+Z^T}{2}\right)\cdot\mathbf{1}=\mathbf{1}\\ & Z\geq 0, Z\succeq 0\end{cases}\tag{C.6}$$

We are given a set $X\subset\mathbf{R}^d$. X can be covered by a set of k "well-separated" balls. That is, $X:=\cup_{i=1}^k B_i$ where B_i is a ball of radius at most r centered at μ_i and $\|\mu_i-\mu_j\|\geq\delta r$. Define $n_i:=|B_i|$ and $n:=\min_{i\in[k]}n_i$ and $N=\sum_i n_i$. D is an $N\times N$ matrix such that $D_{ij}=\|x_i-x_j\|^2$.

The goal is to output a clustering $\mathcal{C}^*$ of X such that $C^*=\{B_1,B_2,\ldots,B_k\}$. From the way we constructed the 0-1 SDP, this corresponds to

$$Z^*=\sum_{p=1}^{k}\frac{\mathbf{1}^p\mathbf{1}^{p^T}}{n_p}\tag{C.7}$$

where $\mathbf{1}^p$ is an N-dimensional indicator vector for the p^{th} cluster. That is, Z is a block diagonal matrix and consists of k non-zero diagonal blocks. Observe that Z consists of blocks $Z^{(p,q)}$. Also, $Z^{(p,p)}=\frac{1}{n_p}\mathbf{1}\mathbf{1}^T$ and for $p\neq q, Z^{(p,q)}=0$. To prove that our SDP (Eqn. C.6) finds this solution, we will adopt the following strategy. We first construct a dual for Eqn. C.6. We then show that under certain conditions on δ (well-separateness of the balls) the following happens. The primal objective value and the dual objective value are the same. Also, the corresponding Z satisfies Eqn. C.7.

Before, we describe the dual, lets introduce a bit of notation. We index every point as (p,i) where p denotes the ball (or cluster) to which it belongs and i denotes the index within that ball. Observe that the distance matrix D consists of blocks $D^{(p,q)}$ such that $D_{ij}^{(p,q)}=\|x_{(p,i)}-x_{(q,j)}\|_2^2$. Now, to construct the dual, we introduce variables $z,\alpha_{(p,i)},\beta_{(p,i)(q,j)}$

and Q for each of the constraints in the primal problem.

$$\textbf{SDP Dual} \begin{cases} \max & -zk - \sum_{p=1}^{k} \sum_{i=1}^{n_p} \alpha_{(p,i)} \\ \text{subject to} & \\ & Q = D + zI + \sum_p \sum_i \alpha_{(p,i)} A_{(p,i)} - \sum_{p,q} \sum_{i,j} \beta_{(p,i)(q,j)} E_{(p,i)(q,j)} \\ & \beta \geq 0 \\ & Q \succeq 0 \end{cases} \tag{C.8}$$

where, $A_{(p,i)} = \frac{1}{2}(e_{p,i}\mathbf{1}^T + \mathbf{1}e_{p,i}^T)$ and $E_{(p,i)(q,j)} = e_{(p,i)}e_{(q,j)}^T$. $\mathbf{1}$ is an N-dimensional vector of all ones while $e_{p,i}$ is the indicator vector with one in position (p,i) and zeros elsewhere. Now, we will examine the conditions under which the dual objective value matches the primal objective such that all the constraints of the dual are satisfied.

Complementary slackness

We know that $\beta_{(p,i)(q,j)} Z_{(p,i)(q,j)} = 0$. Now, $Z^{(p,q)} = 0$ and $Z^{(p,p)\neq 0}$. Hence, we get that $\beta^{(p,p)} = 0$. Also, if we have that $Q^{(p,p)}\mathbf{1} = 0$ then $\langle Q, Z \rangle = 0$. This ensures that the second complementary slackness condition is also satisfied.

Some properties of the Q matrix

Before we proceed, let's examine some properties of the dual matrix Q. Observe that for all $1 \leq p \neq q \leq k$,

$$\begin{aligned} Q^{(p,p)} &= D^{(p,p)} + zI_{n_p} + \frac{1}{2}\sum_{i=1}^{n_p} \alpha_{(p,i)}(e_i\mathbf{1}^T + \mathbf{1}e_i^T) \\ Q^{(p,q)} &= D^{(p,q)} + \frac{1}{2}\sum_{i=1}^{n_p} \alpha_{(p,i)} e_i\mathbf{1}^T + \frac{1}{2}\sum_{i=1}^{n_q} \alpha_{(q,i)}\mathbf{1}e_i^T - \beta^{(p,q)} \end{aligned} \tag{C.9}$$

Dual matches intended primal solution

Now, using the fact that $Q^{(p,p)}\mathbf{1} = 0$ implies that

$$\begin{aligned} 0 = e_r^T D^{(p,p)}\mathbf{1} + z + \frac{1}{2}\sum \alpha_{(p,i)}(e_r^T e_i \mathbf{1}^T\mathbf{1} + \mathbf{e}_r^T\mathbf{1}e_i^T\mathbf{1}) = e_r^T D^{(p,p)}\mathbf{1} + z \\ + \frac{1}{2}\sum \alpha_{(p,i)}(e_r^T e_i n_p + 1) = e_r^T D^{(p,p)}\mathbf{1} + z + \frac{1}{2}\sum \alpha_{(p,i)} + \frac{n_p \alpha_{(p,r)}}{2}. \end{aligned}$$

Summing over all r and then substituiting back, we get that for all $1 \leq p \leq k$ and for all i,

$$\begin{aligned}
\alpha_{(p,i)} &= \frac{1^T D^{(p,p)} 1}{n_p^2} - \frac{z}{n_p} - \frac{2e_i^T D^{(p,p)} 1}{n_p} \\
&= \frac{\sum_{i,j} \langle x_{pi} - x_{pj}, x_{pi} - x_{pj} \rangle - 2n_p \sum_j \langle x_{pi} - x_{pj}, x_{pi} - x_{pj} \rangle}{n_p^2} - \frac{z}{n_p} \\
&= \frac{-2n_p^2 \|x_{pi}\|^2 + 4n_p \langle x_{pi}, \sum_j x_{pj} \rangle - 2\langle \sum_i x_{pi}, \sum_j x_{pj} \rangle - zn_p}{n_p^2} \\
&= -2\|x_{pi} - x_p\|^2 - \frac{z}{n_p} \qquad (x_p \text{ denotes the center of the } p^th \text{ cluster}) \quad \text{(C.10)}
\end{aligned}$$

Now, we have the value of α for all $1 \leq p \leq k$ and for all i. Computing the objective function, we get that

$$\begin{aligned}
kz + \sum_p \sum_i \alpha_{p,i} &= kz + \sum_p \frac{1^T D^{(p,p)} 1}{n_p} - \sum_p z - 2\sum_p \frac{1^T D^{(p,p)} 1}{n_p} = -\sum_p \frac{\mathbf{1}^T D^{(a,a)} \mathbf{1}}{n_p} \\
&= -\langle D, Z \rangle = -\mathrm{Tr}(DZ)
\end{aligned}$$

Hence, we see that for the intended solution, the primal and dual values are the same. Hence, solution is optimal. Now, the main question is to find Q such that Q is positive semi-definite while simultaneously ensuring that $\beta \geq 0$.

Satisfying PSD for Q

We already know that $Q^{(p,p)}\mathbf{1} = 0$. We will now try to ensure that $Q^{(p,q)}\mathbf{1} = 0$. As we will see later, this will help us to prove the positive semi-definiteness property for Q. Now, $Q^{(p,q)}1 = 0$ implies that for all r, we have $e_r^T Q^{(p,q)} 1 = 0$.

$$0 = e_r^T Q^{(p,q)} 1 = \sum_s Q_{rs}^{(p,q)} = \sum_s D_{rs}^{(p,q)} + \frac{n_q \alpha_{(p,r)}}{2} + \frac{1}{2} \sum_{i=1}^{n_q} \alpha_{(q,i)} - \sum_s \beta_{rs}^{(p,q)}$$

It is always possible to satisfy the above equation by choosing $\beta_{rs}^{(p,q)}$ as long as it is greater than zero. That is we need that,

$$\begin{aligned}
\beta_{rs}^{(p,q)} &:= \frac{\sum_s D_{rs}^{(p,q)}}{n_q} + \frac{\alpha_{(p,r)}}{2} + \frac{\sum_{i=1}^{n_q} \alpha_{(q,i)}}{2n_q} \geq 0 \\
&\iff \frac{\sum_s \|x_{pr} - x_{qs}\|^2}{n_q} \geq \frac{\sum_s \|x_{qs} - x_q\|^2}{n_q} + \|x_{pr} - x_p\|^2 + \frac{z}{2n_p} + \frac{z}{2n_q} \qquad \text{(C.11)}
\end{aligned}$$

Before, we go further lets examine,

$$\sum_s \|x_{pr} - x_{qs}\|^2 = n_q\|x_{pr}\|^2 - 2n_q\langle x_{pr}, x_q\rangle + \sum_s \|x_{qs}\|^2$$
$$= n_q\|x_{pr} - x_q\|^2 + \sum_s \langle x_{qs}, x_{qs}\rangle - n_q\langle x_q, x_q\rangle = n_q\|x_{pr} - x_q\|^2 + \sum_s \|x_{qs} - x_q\|^2$$

Substituting this in Eqn. C.11, we get that it is always possible to satisfy $Q^{(p,q)}1 = 0$ as long as for all r, we have that

$$\|x_{pr} - x_q\|^2 - \|x_{pr} - x_p\|^2 \geq \frac{z}{2n_p} + \frac{z}{2n_q} \tag{C.12}$$

Also, note that from $\beta_{rs}^{(p,q)}$ as defined in Eqn. C.11, we get that

$$Q_{rs}^{(p,q)} = D_{r,s}^{(p,q)} + \frac{1}{2}\alpha_{q,s} - \frac{1}{n_q}\sum_j D_{rj}^{pq} - \frac{1}{2}\frac{\sum_j \alpha_{qj}}{n_q} \qquad \text{and from Eqn. C.9} \tag{C.13}$$

$$Q_{rs}^{(p,p)} = D_{r,s}^{(p,p)} + \frac{1}{2}\alpha_{p,r} + \frac{1}{2}\alpha_{p,s} + z\mathbf{1}_{[r=s]} \tag{C.14}$$

If Eqn.C.12 holds, then for all $1 \leq p, q \leq k$, we have that $Q^{(p,q)}\mathbf{1} = 0$. Let $\mathbf{1}_p$ denote the N-dimensional indicator vector for the p^{th} cluster. Then, we see that for all $1 \leq p \leq k$, we have that $Q\mathbf{1}_p = 0$. Let V be the subspace spanned by these vectors. That is, $V = span\{1_p\}_{p=1}^k$. Then, for all $v \in V$, $v^TQy = v^T\mathbf{0} = \mathbf{0}$. Hence, we need to only show that for all $v \perp V$, $v^TQv \geq 0$. Let $v = [v_1, \ldots, v_k]^T$. Since, $v \perp V$, we know that for all p, $\langle v_p, 1\rangle = 0 = \sum_r v_{pr}$. Now,

$$v^TQv = \sum_{pq}\sum_{rs} x_{pr}Q_{rs}^{(p,q)}v_{qs} = \sum_{p\neq q}\sum_{rs} v_{pr}Q_{rs}^{(p,q)}v_{qs} + \sum_p\sum_{rs} v_{pr}Q_{rs}^{(p,p)}v_{qs}$$

Now, we analyse the case when $p \neq q$. Then, we have that

$$\sum_{rs} v_{pr}Q_{rs}^{pq}v_{qs} = \sum_{rs} v_{pr}D_{rs}^{pq}v_{qs} + \frac{1}{2}\sum_{rs}\alpha_{qs}v_{qs}v_{pr} - \frac{1}{n_q}\sum_{rs}\sum_j v_{pr}v_{qs}D_{rj}^{pq}$$
$$- \frac{1}{2n_q}\sum_{rs}\sum_j v_{pr}v_{qs}\alpha_{qj}$$
$$= \sum_{rs} v_{pr}v_{qs}D_{rs}^{pq} + \frac{1}{2}\sum_s \alpha_{qs}v_{qs}\sum_r v_{pr} - \frac{1}{n_q}\sum_r\sum_j D_{rj}^{pq}v_{pr}\sum_s v_{qs}$$
$$- \frac{1}{2n_q}\sum_s\sum_j v_{qs}\alpha_{qj}\sum_r v_{pr} = \sum_{rs} v_{pr}v_{qs}D_{rs}^{pq}$$

Now for the other case, we have that

$$\begin{aligned}\sum_{rs} v_{pr} Q^{pp}_{rs} v_{ps} &= \sum_{rs} v_{pr} D^{pp}_{rs} v_{ps} + \frac{1}{2}\sum_{rs} \alpha_{pr} v_{pr} v_{ps} + \frac{1}{2}\sum_{rs} \alpha_{ps} v_{pr} v_{ps} + \sum_{r} z v_{pr} v_{pr} \\ &= \sum_{rs} v_{pr} D^{pp}_{rs} v_{ps} + \frac{1}{2}\sum_{r} \alpha_{pr} v_{pr} \sum_{s} v_{ps} + \frac{1}{2}\sum_{s} \alpha_{ps} v_{ps} \sum_{r} v_{pr} + \sum_{r} z v_{pr} v_{pr} \\ &= \sum_{rs} v_{pr} D^{pp}_{rs} v_{ps} + \sum_{r} z v_{pr} v_{pr}\end{aligned}$$

Combining the above two equations, we get that

$$v^T Q v = \sum_{pq} v_{pr} D^{pq} v_{qs} + z \sum_{p}\sum_{r} (v_{pr})^2 = v^T D v + z v^T v$$

Now, let X be the $N \times d$ dimensional input matrix. That is, the matrix X contains the N points in d dimensional euclidean space. Then, $D = W + W^T - 2XX^T$ where W is a rank one matrix such that its i^{th} contains $\|x_i\|^2$ in its i^{th} row. That is, $W = \sum_i \|x_i\|^2 e_i \mathbf{1}^T$. Now, $v \perp V$, hence we get that $v^T D v = -2 v^T X X^T v$. Thus, Q is positive semi-definite as long as we can find z such that

$$z > 2 \max_{v \perp V} \frac{v^T X X^T v}{v^T v}. \tag{C.15}$$

Putting it all together

Eqns. C.15 and C.12, show that as long as

$$\|x_{pr} - x_q\|^2 - \|x_{pr} - x_p\|^2 > \frac{2}{n}\Big(\max_{v \perp V} \frac{v^T X X^T v}{v^T v}\Big) \tag{C.16}$$

then we can find Q and β satisfying the constraints of the dual and there is no primal and dual gap. First observe that LHS of Eqn. C.16 has a minimum of $(\delta - 1)^2 - 1$. Now, we need to upper bound the RHS of Eqn. C.16. Note that $X = X' + X$ where C is a rank k matrix which contains the centers $\mu_1, \ldots, \mu_k$. Also, for any $v \perp V$, we have that $v^T C = 0$. Let $\sigma_{\max}$ denote the maximum eigenvalue of the matrix X. Hence,

$$\frac{2}{n}\Big(\max_{v \perp V} \frac{v^T X X^T v}{v^T v}\Big) = \frac{2}{n}\Big(\max_{v \perp V} \frac{v^T X' X'^T v}{v^T v}\Big) \leq \frac{2}{n}\sigma_{\max}\big(X'\big)^2 \tag{C.17}$$

The last inequality follows from the Defn. of $\sigma_{\max}$. (Eqn. 5.3 in [Vershynin, 2010]). In the current setting, $X := \cup_{i=1}^{k} B_i$ where each B_i is generated as follows. Let $\mathcal{P}$ denote the

isotropic distribution on the ball centered at origin of radius at most r, that is $B_1(r) \subset \mathbf{R}^d$. Let B_i be a set of n_i points drawn according to P_i, the measure $\mathcal{P}$ translated to μ_i. Also, $\|\mu_i - \mu_j\| > \delta r > 2r$.

Let $\Theta = E[\|x'_{pr}\|^2]$. Using Thm. C.9, we can bound the RHS of Eqn. C.16 by upper bounding the maximum eigenvalue of X' as

$$\mathbf{P}\left[\sigma_{\max}\left(\sqrt{\frac{d}{\theta}}X'\right) > \sqrt{N} + t\sqrt{\frac{d}{\theta}}\right] \leq 2d\exp(-ct^2) \tag{C.18}$$

Now, let $t\sqrt{\frac{d}{\theta}} = s\sqrt{N}$. Then, we get that with probability atleast $1 - 2d\exp(-\frac{c\theta N s^2}{d})$ we have that

$$\frac{2}{n}\sigma_{\max}(X')^2 \leq 2(1+s)^2\frac{N\theta}{nd} \leq 2\rho\theta(1+s)^2\frac{1}{d}$$

So we see that as long $(\delta-1)^2 - 1 > 2k\rho\theta(1+s)^2\frac{1}{d}$, the primal and dual objective value are the same with high probability. In other words $\delta > 1 + \sqrt{1 + 2\rho\theta(1+s)^2\frac{k}{d}}$ implies the desired conditions. Now, we are finally ready to state our result.

Theorem C.6. *Let $\mathcal{P}$ denote the isotropic distribution on the unit ball centered at origin in $\mathbf{R}^d$. Given points $\mu_1, \ldots, \mu_k$ such that $\|\mu_i - \mu_j\| > \delta > 2$. Let $\mathcal{P}_i$ be the measure $\mathcal{P}$ translated with respect to the center μ_i. Let each B_i is drawn i.i.d w.r.t the distribution $\mathcal{P}_i$.*

Given a clustering instance $\mathcal{I} \subset \mathbf{R}^{N\times d}$ and k where $\mathcal{I} := \cup_{i=1}^k B_i$. Define $n := \min_{i\in[k]} |B_i|$ and $\rho = \frac{N}{nk}$. If

$$\delta > 1 + \sqrt{1 + \frac{2\theta\rho k}{d}\left(1 + \frac{1}{\log N}\right)^2}$$

where $\theta = \mathbf{E}[\|x_{pi} - c_p\|^2] < 1$, then there exists a constant $c > 0$ such that with probability at least $1 - 2d\exp(\frac{-cN\theta}{d\log^2 N})$ the k-means SDP finds the intended cluster solution $\mathcal{C}^* = \{B_1, \ldots, B_k\}$.

C.4 Tightness of the regularized SDP based algorithm

$$\textbf{0-1 SDP}\begin{cases}\min_{Z,y} & \mathrm{Tr}(DZ)+\lambda\langle \mathbf{1}, y\rangle \\ \text{s.t.} & \mathrm{Tr}(Z)=k \\ & Z\cdot\mathbf{1}+y=\mathbf{1} \\ & Z\geq 0 \\ & Z^2=Z, Z^T=Z \\ & y\in\{0,1\}^n\end{cases}\xrightarrow{\text{relaxed}}\textbf{SDP}\begin{cases}\min_{Z} & \mathrm{Tr}(DZ)+\lambda\langle \mathbf{1}, y\rangle \\ \text{s.t.} & \mathrm{Tr}(Z)=k \\ & \left(\frac{Z+Z^T}{2}\right)\cdot\mathbf{1}+y=\mathbf{1} \\ & Z\geq 0, y\geq 0, Z\succeq 0\end{cases} \tag{C.19}$$

We are given a set $X\subset\mathbf{R}^d$. $X=\mathcal{I}\cup\mathcal{N}$ is such that $\mathcal{I}$ can be covered by a set of k "well-separated" balls. That is, $\mathcal{I}:=\cup_{i=1}^k B_i$ where B_i is a ball of radius at most r centered at μ_i and $\|\mu_i-\mu_j\|_2^2\geq\delta r$. Define $n_i:=|B_i|$ and $n:=\min_{i\in[k]}n_i$ and $m:=|\mathcal{N}|=n_{k+1}$ and $N=\sum_i n_i+m$. D is an $N\times N$ matrix such that $D_{ij}=\|x_i-x_j\|^2$.

Note that the clustering algorithm gets X as input and does not know about the sets B_i's or $\mathcal{I}$ or $\mathcal{N}$. The goal is to output a clustering $\mathcal{C}^*$ of X such that $C^*=\{B_1,B_2,\ldots,B_k,\mathcal{N}\}$. From the way we constructed the 0-1 SDP, this corresponds to

$$Z^*=\sum_{p=1}^{k}\frac{\mathbf{1}_p\mathbf{1}_p^T}{n_p}\quad\text{and}\quad y^*=\mathbf{1}_{k+1} \tag{C.20}$$

where $\mathbf{1}_p$ is an N-dimensional indicator vector for the p^{th} cluster. That is, Z is a block diagonal matrix and consists of k non-zero diagonal blocks. Observe that Z consists of blocks $Z^{(p,q)}$. Also, for $1\leq p\leq k$, we have that $Z^{(p,p)}=\frac{1}{n_p}\mathbf{1}\mathbf{1}^T$ and for all $p\neq q$, $Z^{(p,q)}=0$. Also, $Z^{(k+1,k+1)}=0$. To prove that the regularized SDP (Eqn. C.19) finds the desired solution, we will adopt the following strategy. We first construct a dual for Eqn. C.19. We then show that under certain conditions on δ (well-separateness of the balls) and m (the number of noisy points) the following happens. The primal objective value and the dual objective value are the same. Also, the corresponding Z satisfies Eqn. C.32.

Before, we describe the dual, lets introduce a bit of notation. We index every point as (p,i) where p denotes the ball (or cluster) to which it belongs and i denotes the index within that ball. Observe that the distance matrix D consists of blocks $D^{(p,q)}$ such that $D_{ij}^{(p,q)}=\|x_{(p,i)}-x_{(q,j)}\|_2^2$. Now, to construct the dual, we introduce variables $z,\alpha_{(p,i)},\beta_{(p,i)(q,j)},\gamma_{(p,i)}$

and Q for each of the constraints in the primal problem.

$$\textbf{SDP Dual} \begin{cases} \max & -zk - \sum_{(p,i)} \alpha_{(p,i)} \\ \text{subject to} & Q = D + zI + \sum_p \sum_i \alpha_{(p,i)} A_{(p,i)} - \sum_{p,q} \sum_{i,j} \beta_{(p,i)(q,j)} E_{(p,i)(q,j)} \\ & \sum_{(p,i)} (\gamma_{(p,i)} - \alpha_{(q,i)}) e_{(p,i)} = \lambda \mathbf{1} \\ & \beta \geq 0, \gamma \geq 0 \\ & Q \succeq 0 \end{cases} \tag{C.21}$$

where, $A_{(p,i)} = \frac{1}{2}(e_{p,i}\mathbf{1}^T + \mathbf{1}e_{p,i}^T)$ and $E_{(p,i)(q,j)} = e_{(p,i)}e_{(q,j)}^T$. $\mathbf{1}$ is an N-dimensional vector of all ones while $e_{(p,i)}$ is the indicator vector with one in position (p,i) and zeros elsewhere. Now, we will examine the conditions under which the dual objective value matches the primal objective such that all the constraints of the dual are satisfied.

Complementary slackness

We know that $\beta_{(p,i)(q,j)} Z_{(p,i)(q,j)} = 0$. Now, for all $1 \leq p \leq k$, $Z^{(p,p)\neq 0}$ and for all the other pairs (p,q) we have that $Z^{(p,q)} = 0$. Hence, we get that for all $1 \leq p \leq k$, $\beta^{(p,p)} = 0$. Also, we know that $\gamma_{(p,i)} y_{(p,i)} = 0$. Now, $y_{(k+1,i)} \neq 0$, hence $\gamma_{(k+1,i)} = 0$. Also, if we have that for all $1 \leq k \leq p$, $Q^{(p,p)}\mathbf{1} = 0$ then $\langle Q, Z \rangle = 0$. This ensures that the second complementary slackness condition is also satisfied.

Some properties of the Q matrix

Before we proceed, let's examine some properties of the dual matrix Q. Observe that for all $1 \leq p \neq q \leq k$,

$$Q^{(p,q)} = \begin{cases} D^{(p,p)} + z\mathbf{I} + \frac{1}{2}\sum_{i=1}^{n_p} \alpha_{(p,i)}(e_i\mathbf{1}^T + \mathbf{1}e_i^T) & \text{if } 1 \leq p = q \leq k \\ D^{(k+1,k+1)} + z\mathbf{I} - \lambda\mathbf{1}\mathbf{1}^T - \beta^{(p,q)} & \text{if } p = q = k+1 \\ D^{(p,q)} + \frac{1}{2}\sum_{i=1}^{n_p} \alpha_{(p,i)} e_i\mathbf{1}^T + \frac{1}{2}\sum_{i=1}^{n_q} \alpha_{(q,i)}\mathbf{1}e_i^T - \beta^{(p,q)} & \text{otherwise} \end{cases} \tag{C.22}$$

Dual matches intended primal solution

Now, using the fact that for all $1 \leq p \leq k$, $Q^{(p,p)}\mathbf{1} = 0$ implies that

$$\begin{aligned} 0 &= e_r^T D^{(p,p)}\mathbf{1} + z + \frac{1}{2}\sum \alpha_{(p,i)}(e_r^T e_i \mathbf{1}^T\mathbf{1} + \mathbf{e}_r^T\mathbf{1}e_i^T\mathbf{1}) \\ &= e_r^T D^{(p,p)}\mathbf{1} + z + \frac{1}{2}\sum \alpha_{(p,i)}(e_r^T e_i n_p + 1) = e_r^T D^{(p,p)}\mathbf{1} + z + \frac{1}{2}\sum \alpha_{(p,i)} + \frac{n_p\alpha_{(p,r)}}{2}. \end{aligned}$$

Summing over all r and then substituiting back, we get that for all $1 \leq p \leq k$ and for all i,

$$\begin{aligned} \alpha_{(p,i)} &= \frac{1^T D^{(p,p)}1}{n_p^2} - \frac{z}{n_p} - \frac{2e_i^T D^{(p,p)}1}{n_p} \\ &= \frac{-2n_p^2\|x_{pi}\|^2 + 4n_p\langle x_{pi}, \sum_j x_{pj}\rangle - 2\langle\sum_i x_{pi}, \sum_j x_{pj}\rangle - zn_p}{n_p^2} = -2\|x_{pi} - x_p\|^2 - \frac{z}{n_p} \end{aligned}$$

Again, using complementary slackness, we know that $\gamma^{(k+1)} = 0$. This implies that $\alpha_{(k+1,i)} = -\lambda$. Combining these, we get that

$$\begin{aligned} \alpha_{(p,i)} &= -2\|x_{pi} - x_p\|^2 - \frac{z}{n_p} \\ \alpha_{(k+1,i)} &= -\lambda \end{aligned} \tag{C.23}$$

Now, we have the value of α for all $1 \leq p \leq k$ and for all i. Computing the objective function, we get that

$$\begin{aligned} kz + \sum_p\sum_i \alpha_{p,i} &= kz + \sum_p \frac{1^T D^{(p,p)}1}{n_p} - \sum_p z - 2\sum_p \frac{1^T D^{(p,p)}1}{n_p} - \lambda m \\ &= -\sum_p \frac{\mathbf{1}^T D^{(a,a)}\mathbf{1}}{n_p} - \lambda\langle\mathbf{1}, y\rangle = -\langle D, Z\rangle - \lambda\langle\mathbf{1}, y\rangle = -\mathrm{Tr}(DZ) - \lambda\langle\mathbf{1}, y\rangle \end{aligned}$$

Satisfying the λ constraint of dual

This constraint implies for all $1 \leq p \leq k, \gamma_{(p,r)} = \alpha_{(p,r)} + \lambda$. This will be satisfied as long as for all p and for all r, $\lambda \geq -\alpha_{(p,r)}$. Choosing λ as below ensures that the constraint is satisfied for all p and all r.

$$\lambda \geq \frac{z}{n} + 2\|x_{p,r} - x_p\|^2 \tag{C.24}$$

where x_p denotes the center of the p^{th} cluster and $x_{p.r}$ denotes the r^{th} point in the p^{th} cluster. Hence, we see that for the intended solution, the primal and dual values are the same. Hence, solution is optimal. Now, the main question is to find Q such that Q is positive semi-definite while simultaneously ensuring that $\beta, \gamma \geq 0$.

Satisfying PSD for Q

Decompose Q as follows.

$$Q = \begin{bmatrix} Q' & B_1 \\ B_2 & Q^{(k+1,k+1)} \end{bmatrix}$$

If $B_1 = B_2 = 0$ and $Q' \succeq 0$ and $Q^{(k+1,k+1)} \succeq 0$ then we know that $Q \succeq 0$. Let $X_1 = \{C_1, \ldots, C_k\}$ be the set of all points which were assigned to the $1 \leq p \leq k$ clusters. From the proof of the noiseless case, we know that if

$$\|x_{pr} - x_q\|^2 - \|x_{pr} - x_p\|^2 \geq \frac{z}{n} \geq \frac{2}{n}\Big(\max_{v \perp V} \frac{v^T X_1 X_1^T v}{v^T v}\Big) \tag{C.25}$$

where $V = span\{1_p\}_{p=1}^k$ is the subspace spanned by $\mathbf{1}_p$ (the indicator vector for the p^{th} cluster) then Q' is positive semi-definite. We know that the RHS is upper bounded by the square of maximum eigenvalue of X_1, which gives the following

$$\|x_{pr} - x_q\|^2 - \|x_{pr} - x_p\|^2 \geq \frac{z}{n} \geq \frac{2}{n}\sigma^2_{\max}(X_1') \tag{C.26}$$

Hence, if either Eqn. C.25 or Eqn. C.26 can be satisfied then we Q' is positive semi-definite. Here, the matrix X_1' is such that $X_1 = X_1' + C$ where C is a rank k matrix which contains the centers $\mu_1, \ldots \mu_k$. Next, to ensure that $B_1 = 0$, we need that for all $1 \leq p \leq k, Q^{(p,k+1)} = 0$. Using Eqn. C.22,

$$0 = Q_{rs}^{(p,k+1)} = D_{rs}^{(p,k+1)} + \frac{1}{2}\sum_i \alpha_{(p,i)} e_r^T e_i \mathbf{1}^T e_s - \frac{\lambda}{2} e_r^T \mathbf{1}\mathbf{1}^T e_s - \beta_{rs}^{(p,k+1)}$$

$= D_{rs}^{(p,k+1)} - \|x_{pr} - x_p\|^2 - \frac{z}{2n_p} - \frac{\lambda}{2} - \beta_{rs}^{(p,k+1)}$. We need to choose $\beta_{rs}^{(p,k+1)} \geq 0$. Hence,

$$\implies \|x_{k+1,s} - x_{pr}\|^2 \geq \frac{\lambda}{2} + \frac{z}{2n_p} + \|x_{pr} - x_p\|^2$$

Thus, we see that if

$$\|x_{k+1,s} - x_{pr}\|^2 \geq \frac{\lambda}{2} + \frac{z}{2n} + \|x_{pr} - x_p\|^2 \tag{C.27}$$

then for all $1 \leq p \leq k$, we have that $Q^{(p,k+1)} = 0$. In other words, we have that $B_1 = 0$. Next, to ensure that $B_2 = 0$, we need that for all $1 \leq q \leq k, Q^{(k+1,q)} = 0$. Using Eqn. C.22,

$$0 = Q_{rs}^{(k+1,q)} = D_{rs}^{(k+1,q)} + \frac{1}{2}\sum_i \alpha_{(q,i)} e_r^T e_i \mathbf{1}^T e_s - \frac{\lambda}{2} e_r^T \mathbf{1}\mathbf{1}^T e_s - \beta_{rs}^{(k+1,q)}$$

Using the same analysis as before, we see that if $\|x_{k+1,r} - x_{qs}\|^2 \geq \frac{\lambda}{2} + \frac{z}{2n} + \|x_{qs} - x_q\|^2$ then for all $1 \leq q \leq k$, we have that $Q^{(p,k+1)} = 0$. Observe that this is the same condition as Eqn. C.27. Thus, this ensures that $B_2 = 0$. Next, we need to show positive semi-definiteness of the matrix $Q^{(k+1,k+1)}$. Again, using Eqn. C.22, we get that for any vector $v \in \mathbf{R}^m$

$$v^T Q^{(k+1,k+1)} v = v^T D^{(k+1,k+1)} v - v^T \beta^{(k+1,k+1)} v + z v^T v - \lambda (v^T \mathbf{1})^2$$

To show that $Q^{(k+1,k+1)}$ is positive semi-definite, we need to ensure that the above is ≥ 0 for all v. If we choose $\beta^{(k+1,k+1)} = D^{(k+1,k+1)}$ and

$$\frac{z}{m} > \lambda \tag{C.28}$$

$$\text{then, we have that} \quad v^T Q^{(k+1,k+1)} v = z\sum_i v_i^2 - \lambda(\sum_i v_i)^2 \geq (z - \lambda m)\sum_i v_i^2 \geq 0$$

Putting it all together

We are given $X := I \cup N$. Let $I = \cup B_i$ where each B_i is ball of radius at most one and centered at μ_i where μ_i is the average of points in B_i. Also, $d(\mu_i, \mu_j) \geq \delta$. Decompose $N = N_1 \cup N_2$ into two sets. Let $N_2 = \{n \in N : \text{for all } b \in I, \|n - b\| \geq \nu\}$ and $N_1 = N \setminus N_2$. Let N_1 be such that for all $n \in N_1$, $|\|n - \mu_i\|^2 - \|n - \mu_j\|^2| \geq \alpha$.

We will show that the regularized SDP outputs the clustering $\mathcal{C} = \{C_1, \ldots, C_k, N_2\}$, where each $B_i \subseteq C_i$. Hence, the clusters contain all the points from the balls B_i plus (maybe) points from the set N_1.

Consider the p^{th} cluster C_p. We know that $C_p = B_p \cup M_p$ where $M_p \subseteq N_1$. Now, $x_p = \frac{\sum_{x \in B_p} x + \sum_{n \in M_p} n}{|B_p| + |M_p|} = \frac{\mu_p |B_p| + |M_p| avg(M_p)}{|B_p| + |M_p|}$. Thus, we get that $\|x_p - \mu_p\| = \frac{|M_p|}{|B_p| + |N_p|}\|\mu_p - avg(M_p)\| \leq \frac{|N_1|}{n}(\nu + 1) =: a$. Thus, we have that for all $x_{pr} \in B_p$

$$\|x_{pr} - x_q\|^2 - \|x_{pr} - x_p\|^2 \geq (\|x_{pr} - \mu_q\| - \|x_q - \mu_q\|)^2 - (\|x_{pr} - \mu_p\| + \|x_p - \mu_p\|)^2$$
$$= \|x_{pr} - \mu_q\|^2 - \|x_{pr} - \mu_p\|^2 + \|x_q - \mu_q\|^2 - \|x_p - \mu_p\|^2 - 2\|x_{pr} - \mu_p\|\|x_p - \mu_p\|$$
$$- 2\|x_{pr} - \mu_q\|\|x_q - \mu_q\| \geq (\delta - 1)^2 - 1 - a^2 - 4a$$

and for $x_{pr} \in M_p$

$\|x_{pr} - x_q\|^2 - \|x_{pr} - x_p\|^2 \geq \alpha - a^2 - 4a$

Choosing $\frac{z}{n} = (\delta-1)^2 - 1 - a^2 - 4a > \frac{2}{n}\sigma^2_{\max}(X'_1)$ and $\alpha - a^2 - 4a > \frac{2}{n}\sigma^2_{\max}(X'_1)$ ensures that Eqn. C.26 is satisfied. Next, we see that if λ is such that

$$2\|x_{k+1,r} - x_{qs}\|^2 - (\delta-1)^2 - 1 \geq \lambda \geq (\delta-1)^2 + 1 \tag{C.29}$$

then Eqns. C.24 and Eqns. C.27 can be satisfied. Hence, if $\nu \geq \sqrt{1+(\delta-1)^2}$ then the above condition can be satisfied. Finally, combining Eqns. C.26 and C.29, we see that if

$$|N_2| \leq n\frac{(\delta-1)^2 - 1 - a^2 - 4a}{\lambda}$$

then Eqn. C.28 can also be satisfied.

Now, for X as generated by the isotropic distribution, we need to ensure that Eqn. C.25 can be satisfied.

We will now upper bound the we can bound the RHS of Eqn. C.25. Decompose $X_1 := A'_1 + C + N'_1$. Now, A'_1 contains points from the balls $B'_1, \ldots, B'_k$. C contains the centers $\mu_1, \ldots, \mu_k$ and N'_1 contains the points from the set $|N_1|$ but shifted by μ_p. Now,

$$\begin{aligned}&\frac{2}{n}\Big(\max_{v\perp V}\frac{v^T X_1 X_1^T v}{v^T v}\Big) = \frac{2}{n}\Big(\max_{v\perp V}\frac{v^T A'_1 A_1^{'T} v}{v^T v}\Big) + \frac{2}{n}\Big(\max_{v\perp V}\frac{v^T N_1 N_1^T v}{v^T v}\Big)\\ &\leq \sigma^2_{\max}(A'_1) + \sigma^2_{\max}(N'_1)\end{aligned}$$

Now, it's easy to see that $\sigma^2_{\max}(N'_1) \leq |N_1|(\nu+1)$. Let $\theta = E[\|a'_{pr}\|^2]$. Using Thm. C.9, we will upper bound the maximum eigenvalue of A'_1 as

$$\mathbf{P}\left[\sigma_{\max}\left(\sqrt{\frac{d}{\theta}}A'\right) > \sqrt{|I|} + t\sqrt{\frac{d}{\theta}}\right] \leq 2d\exp(-ct^2) \tag{C.30}$$

Now, let $t\sqrt{\frac{d}{\theta}} = s\sqrt{|I|}$. Then, we get that with probability atleast $1 - 2d\exp(-\frac{c\theta|I|s^2}{d})$ we have that

$$\frac{2}{n}\sigma^2_{\max}(A'_1) \leq 2(1+s)^2\frac{|I|\theta}{nd} \leq 2k\rho\theta(1+s)^2\frac{1}{d}$$

Thus, $(\delta-1)^2 - 1 - a^2 - 4a > 2a + 2\rho k\theta(1+s)^2\frac{1}{d}$ implies the desired conditions. Now, we are finally ready to state our result.

Theorem C.7. *Let $\mathcal{P}$ denote the isotropic distribution on the unit ball centered at origin in $\mathbf{R}^d$. Given centers $\mu_1, \ldots, \mu_k$ such that $\|\mu_i - \mu_j\| > \delta > 2$. Let $\mathcal{P}_i$ be the measure $\mathcal{P}$ translated with respect to the center μ_i. Let B_i is drawn i.i.d w.r.t the distribution $\mathcal{P}_i$.*

Given a clustering instance $X \subset \mathbf{R}^{N\times d}$ and k. Let $X := \mathcal{I} \cup \mathcal{N}$ where $\mathcal{I} := \cup_{i=1}^{k} B_i$. Let $\mathcal{N} = \mathcal{N}_1 \cup N_2$ have the following properties. For all $n \in N_1$ and for all i, j, we have that $|\|(n-\mu_i\|^2 - \|n-\mu_j\|^2| \geq \alpha$. For all $n \in \mathcal{N}_2$ and for all $x \in \mathcal{I}$, $\|n-x\| \geq \nu \geq \sqrt{(\delta-1)^2+1}$. Note that $\mathcal{N}_1 \cap \mathcal{N}_2 = \phi$. Let $n = \min_i |B_i|$ and $\epsilon = \frac{|\mathcal{N}_1|}{n}$ and $\rho = \frac{|I|}{nk}$. If

- $\delta > 2 + \sqrt{O(\epsilon) + \frac{2\rho k\theta(1+1/\log(|I|))^2}{d}}$
- $\alpha \geq O(\epsilon) + \frac{2\rho k\theta(1+1/\log|I|)^2}{d}$
- $\frac{|\mathcal{N}_2|}{n} \leq \frac{\delta^2 - 2\delta - O(\epsilon)}{\lambda}$

then there exists a constant $c_2 > 0$ such that with probability at least $1 - 2d\exp(\frac{-c_2|I|\theta}{d\log^2|I|})$ the regularized k-means SDP finds the intended cluster solution $\mathcal{C}^* = \{C_1, \ldots, C_k, \mathcal{N}_2\}$ where $B_i \subseteq C_i$ when given X and $\delta^2 + 2\delta \geq \lambda \geq (\delta-1)^2 + 1$ as input.

C.5 Tightness of the regularized LP based algorithm

$$\textbf{IP}\begin{cases}\min_{Z,y} & \sum_{pq} d^2(p,q) z_{pq} + \lambda \sum_p y_p \\ \text{s.t.} & \sum_p z_{pp} = k \\ & \sum_q z_{pq} + y_p = 1 \\ & z_{pq} = z_{qp} \\ & z_{pq} \in \{0, z_{pp}\}, y \in \{0,1\}^n \end{cases} \xrightarrow{\text{relaxed}} \textbf{LP}\begin{cases}\min_{Z,y} & \sum_{pq} d^2(p,q) z_{pq} + \lambda \sum_p y_p \\ \text{s.t.} & \sum_q z_{pq} + y_p = 1 \\ & z_{pq} \leq z_{pp} \\ & \sum_p z_{pp} = k \\ & z_{pq} \geq 0, y_p \geq 0 \end{cases} \tag{C.31}$$

We are given a set $X \subset \mathbf{R}^d$. $X = \mathcal{I} \cup \mathcal{N}$ is such that $\mathcal{I}$ can be covered by a set of k "well-separated" balls. That is, $\mathcal{I} := \cup_{i=1}^{k} B_i$ where B_i is a ball of radius at most r centered at μ_i and $\|\mu_i - \mu_j\|_2^2 \geq \delta r$. Let $\mathcal{P}$ denote the isotropic distribution in the unit ball centered at origin $B_1(r)$ in $\mathbf{R}^d$. The ball B_i is drawn from the isotropic distribution $\mathcal{P}_i$, that is, the measure $\mathcal{P}$ translated with respect to the center μ_i. Define $n_i := |B_i|$ and $n := \min_{i\in[k]} n_i$ and $m := |\mathcal{N}| = n_{k+1}$ and $N = \sum_i n_i + m$. D is an $N \times N$ matrix such that $D_{ij} = \|x_i - x_j\|^2$.

Note that the clustering algorithm gets X as input and does not know about the sets B_i's or $\mathcal{I}$ or $\mathcal{N}$. The goal is to output a clustering $\mathcal{C}^*$ of X such that $C^* = \{B_1, B_2, \ldots, B_k, \mathcal{N}\}$. From the way we constructed the Integer program, this corresponds to

$$Z^* = \sum_{p=1}^{k} \frac{\mathbf{1}_p \mathbf{1}_p^T}{n_p} \quad \text{and} \quad y^* = \mathbf{1}_{k+1} \tag{C.32}$$

where $\mathbf{1}_p$ is an N-dimensional indicator vector for the p^{th} cluster. That is, Z is a block diagonal matrix and consists of k non-zero diagonal blocks. Observe that Z consists of blocks $Z^{(p,q)}$. Also, for $1 \leq p \leq k$, we have that $Z^{(p,p)} = \frac{1}{n_p}\mathbf{1}\mathbf{1}^T$ and for all $p \neq q, Z^{(p,q)} = 0$. Also, $Z^{(k+1,k+1)} = 0$. To prove that the regularized LP (Eqn. C.31) finds the desired solution, we will adopt the following strategy. We first construct a dual for Eqn. C.31. We then show that under certain conditions on δ (well-separateness of the balls) and m (the number of noisy points) the following happens. The primal objective value and the dual objective value are the same. Also, the corresponding Z and y satisfy Eqn. C.32.

Now to construct the dual, we introduce variables $\alpha_p, \beta_{pq}, \gamma$, μ_{pq} and η_p for each of the constraints in the primal problem.

$$\textbf{LP Dual} \begin{cases} \max & \sum_p \alpha_p - \gamma k \\ \text{subject to} & \alpha_p + \mu_{pq} = \beta_{pq} + d^2(p,q) \\ & \gamma = \sum_q \beta_{pq} \\ & \lambda = \alpha_p + \eta_p \\ & \mu_{pq} \geq 0, \eta_p \geq 0 \end{cases} \tag{C.33}$$

Now, we will examine the conditions under which the dual objective value matches the primal objective such that all the constraints of the dual are satisfied. Before we go into more details, lets introduce the following notation. We will refer to points using symbols a, a', b, b', c and c'. a, b denotes two points in different clusters. a, a' and b, b' will refer to a pair of points in the same cluster. c and c' refers to the points in the noisy cluster $k+1$.

Complementary slackness

- $\mu_{pq} z_{pq} = 0$. We get that

$$\mu_{aa'} = \mu_{bb'} = 0 \tag{C.34}$$

- $\eta_p y_p = 0$. We get that

$$\eta_c = 0 \tag{C.35}$$

- $\beta_{pq}(z_{pq} - z_{pp}) = 0$. We get that

$$\beta_{ab} = \beta_{ac} = \beta_{ba} = \beta_{bc} = 0 \tag{C.36}$$

Dual matches intended primal solution

Let $a \in C_i$ where $1 \leq i \leq k$. Hence, we get that

$$\begin{aligned} \sum_{a' \in C_i} \alpha_a + \sum_{a' \in C_i} \mu_{aa'} &= \sum_{a' \in C_i} \beta_{aa'} + \sum_{a' \in C_i} d^2(a, a') \\ \implies \alpha_a &= \frac{\gamma}{n_i} + \frac{\sum_{a' \in C_i} d^2(a, a')}{n_i} \end{aligned} \tag{C.37}$$

For $c \in C_{k+1}$, we have that

$$\alpha_c + \eta_c = \lambda \implies \alpha_c = \lambda.$$

Using these two equations, we get that

$$\begin{aligned} \sum_p \alpha_p - k\gamma = \sum_{i=1}^{k} \sum_{a \in C_i} \alpha_i + \sum_{c \in C_{k+1}} \alpha_c - k\gamma &= \sum_{i=1}^{k} \sum_{a \in C_i} \frac{\gamma}{n_i} + \frac{\sum_{a' \in C_i} d^2(a, a')}{n_i} + \lambda \langle \mathbf{1}, y \rangle - k\gamma \\ = \sum_{i=1}^{k} \gamma + \sum_{i=1}^{k} \frac{1}{n_i} \sum_{a, a' \in C_i} d^2(a, a') + \lambda \langle \mathbf{1}, y \rangle - k\gamma &= \sum_{pq} d^2(p, q) z^*_{pq} + \lambda \sum_p y^*_p \end{aligned}$$

Satisfying the λ constraint of dual

We have already seen that α_c should be equal to λ. Hence, if $\lambda \geq \alpha_a$ for all $a \in C_1, \ldots, C_k$ then this constraint can be satisfied. Thus, we get that if

$$\alpha_a \leq \lambda \quad \text{and} \quad \alpha_c = \lambda \tag{C.38}$$

then the λ constraint of the dual can be satisfied.

Satisfying the α_p constraint

Again observe that for $a \in C_i$ and $b \in C_j \neq C_i$ and $c \in C_{k+1}$

$$\begin{aligned} \alpha_a + \mu_{ab} = d^2(a,b) &\implies \alpha_a \leq d^2(a,b) \\ \alpha_a + \mu_{ac} = d^2(a,c) &\implies \alpha_a \leq d^2(a,c) \end{aligned} \tag{C.39}$$

To satisfy the constraint for α_c, we need that

$$\alpha_c + \mu_{cq} = \beta_{cq} + d^2(c,q) \implies \alpha_c |X| + \sum_q \mu_{cq} = \gamma + \sum_q d^2(c,q)$$

This can be satisfied as long as

$$\lambda |X| \leq \gamma + \sum_q d^2(c,q) \tag{C.40}$$

Putting it all together

From Eqns. C.39, C.38, C.37 and C.40, we see that the following constraints need to be satisfied. Let $a, a' \in C_a$, $b \notin C_a \cup C_{k+1}$ and $c \in C_{k+1}$.

$$d^2(a,a') \leq \frac{\gamma}{n_i} + \frac{\sum_{a_1 \in C_i} d^2(a,a_1)}{n_i} \leq d^2(a,b) \tag{C.41}$$

$$d^2(a,a') \leq \frac{\gamma}{n_i} + \frac{\sum_{a_1 \in C_i} d^2(a,a_1)}{n_i} \leq d^2(a,c) \tag{C.42}$$

$$\frac{\gamma}{n_i} + \frac{\sum_{a_1 \in C_i} d^2(a,a_1)}{n_i} \leq \lambda \leq \frac{\gamma}{|X|} + \sum_q \frac{d^2(c,q)}{|X|} \tag{C.43}$$

Following the exact same analysis as in [Awasthi et al., 2015], we know that Eqn. C.41 can be satisfied with high probability (as the points in the balls B_i are generated by an isotropic distribution). Let ν denote the minimum distance between any point in C_{k+1} to any other point in $C_1, \ldots, C_k$. Choosing $\nu \geq (\delta - 2)$ ensures that Eqn. C.42 is satisfied. Furthermore, if the number of noisy points $m \leq N(1 - \frac{4}{\nu^2})$ then Eqn. C.43 can be satisfied.

Theorem C.8. *Let $\mathcal{P}$ denote the isotropic distribution on the unit ball centered at origin in $\mathbf{R}^d$. Given centers $\mu_1, \ldots, \mu_k$ such that $\|\mu_i - \mu_j\| > \delta > 2$. Let $\mathcal{P}_i$ be the measure $\mathcal{P}$ translated with respect to the center μ_i. Let B_i is drawn i.i.d w.r.t the distribution $\mathcal{P}_i$.*

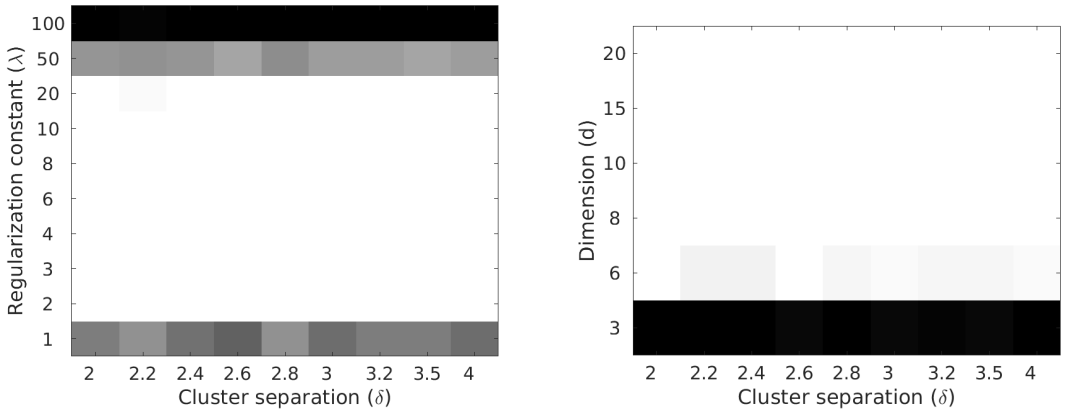


Figure C.1: Heatmap showing the probability of success of the k-means regularized sdp algorithm. Lighter color indicates probability closer to one while darker indicates probability closer to zero.

Given a clustering instance $X \subset \mathbf{R}^d$ and k. Let $X := \mathcal{I} \cup \mathcal{N}$. Let $\mathcal{I} := \cup_{i=1}^{k} B_i$ and $\mathcal{N}$ have the following property. Each $p_n \in \mathcal{N}$ is such that $\min_{p_i \in \mathcal{I}} \|p_n - p_i\| \geq \nu r$. Let $|\mathcal{N}| =: m$. If

- $\delta > 4$ *and* $\nu > \delta - 2$
- $m \leq |\mathcal{I}|\left(\frac{\nu^2}{(\delta-2)^2} - 1\right)$

then the regularized k-means LP finds the intended cluster solution $\mathcal{C}^* = \{B_1, \ldots, B_k, \mathcal{N}\}$ when given X and $(\delta-2)^2 r^2 \leq \lambda \leq \nu^2(1 - \frac{m}{N})$ as input.

C.6 Additional experimental results

Now, we provide some additional experiments on simulated data for the regularized sdp-based algorithm. Fig. C.1 shows the performance of the algorithm as the cluster separation and regularization constant are varied. It shows that for very high and very low values of λ, the performance of our algorithm is bad. There exists a 'correct' range of lambda when it is able to correctly recover the target clustering. Similarly, Figs. C.2 and C.3 shows the performance of the algorithm as other parameters are varied.

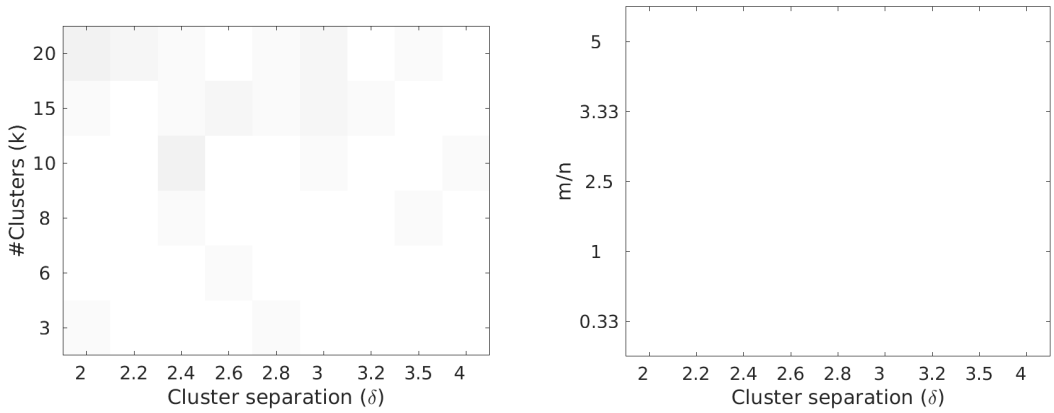


Figure C.2: Heatmap showing the probability of success of the k-means regularized sdp algorithm. Lighter color indicates probability closer to one while darker indicates probability closer to zero.m denotes the number of noisy points while n denotes the number of points in the smallest cluster.

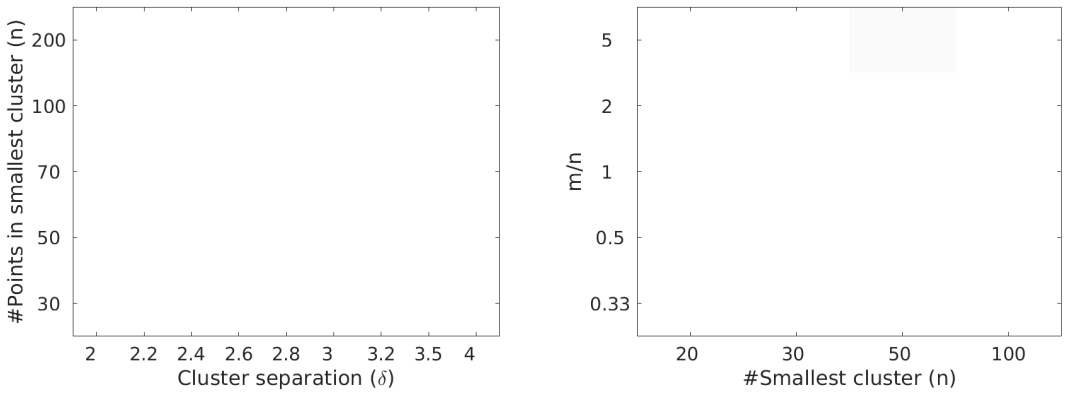


Figure C.3: Heatmap showing the probability of success of the k-means regularized sdp algorithm. Lighter color indicates probability closer to one while darker indicates probability closer to zero. m denotes the number of noisy points while n denotes the number of points in the smallest cluster.

C.7 Technical lemma

Theorem C.9 (Thm. 5.41 in [Vershynin, 2010])**.** *Let A be an $N \times d$ matrix whose rows A_i are independent isotropic random vectors in $\mathbf{R}^d$. Let m be a number such that $\|A_i\| \leq \sqrt{m}$ almost surely for all i. Then for every t, one has*

$$\sqrt{N} - t\sqrt{m} \leq \sigma_{\min}(A) \leq \sigma_{\max}(A) \leq \sqrt{N} + t\sqrt{m}$$

with probability atleast $1 - 2d\exp(-ct^2)$, where c is an absolute constant. $\sigma_{\min}$ and $\sigma_{\max}$ are the spectral norms or the minimum and maximum eigenvalues respectively of the matrix A.