Università degli Studi di Milano

Doctoral Thesis

Algorithms for Clustering and Robust Unsupervised Learning Problems

Author:
Andrea Paudice

Supervisors:
Nicolò Cesa-Bianchi
Massimiliano Pontil

PhD Coordinator:
Paolo Boldi

PhD Program in Computer Science
(XXXIV cycle)

A thesis submitted in fulfillment of the requirements for the degree of Doctor of Philosophy

Academic Year 2020-2021

“*Basically, I’m not interested in doing research and I never have been. I’m interested in understanding, which is quite a different thing. And often to understand something you have to work it out yourself because no one else has done it.*”

David Blackwell

Dedication

In loving memory of Chicco, Lucky I, Jack, Lula, Mottino and Knuth.

Statement of Contribution

All, but the material in Chapter 1, is based on published papers. Specifically, Chapter 2-3-4 are based on a joint work with Marco Bressan, Nicolò Cesa-Bianchi and Silvio Lattanzi (NeurIPS 2020, COLT 2021, NeurIPS 2021). Chapter 5 is based on a joint work with Marco Bressan, Nicolò Cesa-Bianchi and Fabio Vitale (NeurIPS 2019). Finally, Chapter 6 is based on a joint work with Andreas Maurer, Daniela Angela Parletta and Massimiliano Pontil (ICML 2021).

Abstract

This thesis is dedicated to the study of algorithms for clustering and robust unsupervised learning. Specifically, a significant part of this thesis is dedicated to the problem of *semi-supervised cluster recovery*. Given a finite set of n points partitioned according to an unknown clustering, the goal is to recover the unknown clustering using as few *oracle queries* as possible. We mostly focus on the framework of the *same-cluster queries*: given a pair of points x and y, a same cluster query returns a binary answer saying *yes* if x and y belongs to the same cluster, and *no* otherwise. In this framework, we seek for algorithms that by actively asking queries reconstruct exactly the underlying clustering. The target *query complexity* (i.e. the number of queries asked by the algorithm) if $O(\log n)$.

We also consider a more general (and weaker) notion of supervision offered by the *similarity queries*. Given a pair of points x and y, the oracle answer *yes* if the points are *similar* and *no* otherwise. In this context, the similarity may, for example, represent a notion of metric-closeness between points or also the membership to the same-cluster. In the former case, similarity queries may be seen as same-cluster queries with errors since points that are close may be well into different clusters. This type of feedback is common for clustering problems on graphs, where the presence of an edge between two vertices do not need to establish the membership to the same cluster. In this framework, we consider the case of *Correlation Clustering* and design a simple and efficient query-based approximation algorithm for it. We prove also that the same approximation algorithm features some cluster recovery properties.

Finally a third contribution goes into the area of *robust unsupervised learning*. In this framework, data comes from a mixture distribution $\lambda\mu + (1 - \lambda)\nu$ where ν is noise, and the goal is to identify a model with a small risk w.r.t. the target distribution μ. We show that the minimization of a certain *L-statistic* of the training data, provably leads to a model with small risk w.r.t. μ; even in the high noise regime. Our method works in the general *reconstruction error minimization* framework which encompass k-means and k-median clustering, principal subspace analysis, sparse coding and non-negative matrix factorization. Since the proposed method is NP-Hard to compute, we complement the results with an efficient Lloyd-like descent algorithm.

Acknowledgements

I would like to thank my advisors Nicolò Cesa-Bianchi and Massimiliano Pontil for their supervision and all the fun we had during these years. Thanks to Shai Ben-David, Varun Kanade and Sergei Vassilvitskii who kindly accepted to review this thesis.

Contents

Chapter 1

Introduction

This chapter is an overview of the contributions given by this thesis. The objective is to detail the questions investigated in the subsequent chapters, describe the major technical contributions and present the structure of the thesis.

1.1 Research Problems

In this section we details the problems investigated in the subsequent chapters. The research problems faced in this thesis are arranged around three main themes: cluster recovery, approximation-algorithms for correlation clustering, and robustness in unsupervised learning.

1.1.1 Semi-Supervised Cluster Recovery

Problem Formulation. Clustering is one of the most common learning task and can be (roughly) formulated as the problem of identifying groups in a set X of points from some given space $\mathcal{X}$. Such groups are called clusters and the induced partitions of X are called clusterings. There are at least two different ways in which clustering can be formulated. One approach consists in defining a cost function over possible clusterings of the data; once the cost is defined, clustering became an optimization problem that seeks for the clustering with the smallest cost. This approach lead to the so-called optimization-based clustering methods. Examples of such methods include the well known k-center methods, most spectral methods and correlation clustering.

The implicit assumption behind optimization-based methods is that the ground truth clusterings in the data are well approximated by the minima of the cost function. While this assumption holds on many datasets, there are also cases where the underlying clustering is not well approximated by the solutions of any clustering-based optimization. In these cases, it is more convenient to formulate the clustering task as the problem of identifying the ground-truth clustering within a prescribed family of clustering. In the stronger version of this problem, one typically asks for the exact reconstruction of the underlying clustering (modulo label permutations), a problem we refer to as **exact cluster recovery**. This problem can also be relaxed either by allowing for some error and/or by focusing only on the large clusters.

Notice that, unless strong assumptions on the data are made, the problem of exact recovery cannot be solved. As a result, one may relax the problem to allow some form of expert supervision. One popular form of supervision is offered by the so-called **same-cluster-queries** where the algorithm is allowed to ask to a human-expert *do x and y belong to the same cluster?*. The expert answers such questions without making mistakes. These queries are natural in many contexts, such as crowd-sourcing (e.g. the labellers are asked whether two images depict the same city).

We can now define the following search problem problem.

Definition 1.1 (Semi-Supervised Exact Recovery)**.** *Let $X \subseteq \mathcal{X}$ a size n set and $\mathcal{C}$ an unknown partition of X into k-clusters and let* SCQ *be the same-cluster query oracle. The semi-supervised exact cluster recovery problem is defined as the problem of identifying $\mathcal{C}$ by possibly querying* SQC.

Obviously, it is always possible to trivially recover $\mathcal{C}$ by asking all $^{n(n-1)}/_{2}$ queries. However, queries are costly and one is usually interested in using as few queries as possible. We refer to the number of queried asked by the algorithm in the worst case as its *query complexity.* Since we will consider algorithms that can interactively select the oracle queries - as opposed to algorithms that receive a batch of oracle answers at the beginning - the hope is to get an exponential improvement in the query complexity (similarly to what happens in some binary classification problems where one observe a transition from $\mathcal{O}(n)$ to $\mathcal{O}(\log(n))$). Thus the specific research problem we consider is that of establishing conditions on X and the class of clusterings, that allow for the exact recovery of $\mathcal{C}$ using $\mathcal{O}(\log n)$.

Contributions In this thesis we identify conditions and algorithms for the exact recovery of clusterings with $\mathcal{O}(\log n)$ queries in a range of settings. The main results are list below.

1. For convex clusterings in $\mathbb{R}^m$ we provide an algorithm that is capable of recovering any convex clustering satisfying a certain notion of γ-margin, we name *convex-hull margin*, with $\mathcal{O}(k^2 m^5 (1+1/\gamma)^m \log(1+1/\gamma)\log n)$. On the negative side, we show that a dependence on $(1+1/\gamma)^{m/2}$ is necessary to any algorithm.
2. For the case of arbitrary pseudo-metric spaces, we introduce the *one-vs-all* margin, a generalization of the *convex-hull* margin. We design an algorithm that recovers any (possibly non-convex) clustering satisfying the *one-vs-all* margin with $\mathcal{O}(M(\gamma)k\log k \log n)$ queries, where $M(\gamma)$ is a quantity related to the packing number of $\mathcal{X}$. We show that this notion of margin is implied by many popular notions of *data stability* such as SVM margin, α-center proximity and ϵ-perturbation stability. On the negative side, we show that - to achieve exact recovery - any algorithm has to make at least $\Omega(M(\gamma))$ queries in expectation. As a result, the proposed algorithm is essentially optimal as for the dependence on M.
3. For clustering that are realized by binary hypothesis from some class $\mathcal{H}$, we introduce a combinatorial parameter of $\mathcal{H}$ we name *coslicing dimension* denoted by $\textsc{cosl}(\mathcal{H})(\mathcal{H})$. We show that exact recovery with $\mathcal{O}(\log n)$ queries is possible if and only if $\textsc{cosl}(\mathcal{H}) < \infty$. Specifically we show that when the coslicing dimension is bounded, there exist an algorithm for exact recovery that makes $\mathcal{O}(\textsc{cosl}(\mathcal{H})k\log k\log n)$ queries. On the negative side, we show that if $\textsc{cosl}(\mathcal{H})$ is infinite, then any algorithm needs $\Omega(n)$ queries in expectation to achieve exact recovery.
4. Finally, we consider non-convex clusterings in general pseudo-metric spaces satisfying a certain notion of margin-based graph-theoretic convexity named *geodesic convexity* with margin. This notion, intuitively requires that: (i) each cluster C_i is connect at same scale ϵ_i; (ii) growing a small ball of radius at most $\beta\epsilon_i$ around a point $x \in C_i$ does not hit any points from different clusters, (iii) any small detour of length $\gamma\ell$ from the shortest path (whose length is ℓ) among two points from the same cluster does not contain points from other clusters. Under this assumption, we design algorithms that, if given a point for each cluster, recover the underlying (β,γ)-geodesically convex clustering with $\mathcal{O}(k^2\log n + k^2(6/(\gamma\beta))^{D(\mathcal{X})})$ queries, where $D(\mathcal{X})$ is a generalization of the doubling dimension of $\mathcal{X}$. On the lower bound side, we show that no algorithm can recover (γ,β)-geodesically convex clusters with less than $\Omega(n)$ queries if a point from each cluster is not provided as input.

In the first three settings, the provided algorithms have 0-error probability but the query complexity bounds hold with high-probability. In the last setting instead, the algorithm is completely deterministic.

1.1.2 Semi-Supervised Correlation Clustering

Problem Formulation. Correlation Clustering (CC) is a well known optimization-based clustering paradigm which found many applications despite its computational hardness: at best, only constant factor approximation algorithms are admitted. Its success is mostly due to its modelling flexibility (being based on arbitrary similarity graphs) and the fact that it does not requires the user to specify the number of clusters to adopt. In its basic formulation, CC requires the input to be completely specified in the form of a similarity graph, i.e. a clique over the points with $\pm$ signs over the edges. The output is the clustering with the smallest *disagreement coefficient* which we denote with OPT. Given a clustering, its disagreement coefficient is defined as the number of $-$ signs within-clusters plus the number of $+$ signs between clusters. However, as most of modern real world data-sets are huge, obtaining accurate similarities for the entire date is extremely costly. Then, it is natural to extend CC to a query based setting where the algorithm does not get the whole graph beforehand, but can ask the sign of the edges to an oracle. Within this context, a natural research problem is that of designing algorithms which given a query budget Q, obtain a small approximation error.

Contributions The main contributions of this thesis to the query-based extension of CC are the following.

1. We provide a simple query based 3-factor approximation algorithm for CC. The algorithm guarantees an expected cost of at most $3\mathrm{OPT}+\mathcal{O}(n^3/Q)$ using at most Q queries. We complement this result, by showing that with Q queries any algorithm has a cost of at least of $\mathrm{OPT}+\Omega(n^3/Q)$.
2. We show that a slight modification of the algorithm is capable of recovering all *almost clique* clusters; i.e. clusters where only a small fraction of edges between their nodes has been added or removed (even adversarially).
3. We design a fully additive approximation algorithm that with Q queries attains an error of $\mathrm{OPT}+\mathcal{O}(n^{5/2}/\sqrt{Q})$. If $\mathrm{OPT}=0$, the same algorithm attains an error of $\mathcal{O}(n^3/Q)$. We complement this results with a lower bound of $\Omega(n^2/\sqrt{Q})$.

1.1.3 Robust Unsupervised Learning

Problem Formulation. Most real-world data-sets are affected by noise which impacts negatively the performances of many common machine learning algorithms. As an example, it is not hard to show that the standard k-means algorithm can be broken by a single isolated data point. For this reason, it is important to design algorithms that can tolerate some amount of noise. In this context, one popular noise model that has received substantial consideration is the so called *Huber contamination model* which assumes the training data to be generated by a mixture distribution $\nu=(1-\lambda)\mu+\lambda\nu^*$, where μ is the so-called target distribution, ν^* is the so-called perturbing distribution which generate the noise and $\lambda\in[0,1]$ is a scalar controlling the amount of noise. The objective is to learn a model with good performances w.r.t μ from data generated by the mixture ν.

One general formulation of unsupervised learning is the following. Let $\mathcal{S}$ be a family of subsets of $\mathbb{R}^d$ and $S\in\mathcal{S}$, the reconstruction error of S on a point $z\in\mathbb{R}^d$ is defined as $d_S(z)=\inf_{y\in S}\|y-z\|_2^2$. The standard setting where the learner get an i.i.d. data-set $X\sim\mu^n$ is well understood and empirical risk minimization works well in this setting. Indeed, the empirical risk minimization formulation of this general problem includes many popular unsupervised learning problems. For example, if we let $\mathcal{S}$ to be the family of all size-k sets we recover the classical k-means problem. Similarly, if we let $\mathcal{S}$ to be the family of all k-dimensional sub-spaces of $\mathbb{R}^d$ we recover the classical Principal Component Analysis. The research question we investigate in this thesis is the following.

Definition 1.2 (Robust Unsupervised Learning)**.** *Under the Huber contamination model and possibly additional assumptions, is it possible to design methods that provably output a model with small risk with respect to the target distribution μ?*

Contributions In the context of this problem the main contributions of this thesis are the following.

1. We formalize a natural notation of outliers, withing the Huber contamination model, that allows for the design of algorithms that tolerate large fraction of noise, i.e. $\lambda>0.5$. The proposed notion focuses on the model class $\mathcal{S}$: outliers are defined as points that

do not fit any model in the given class. More formally, we require the distribution ν^* to be not as much as concentrated (up to same critical scale) as the target μ on any model in $\mathcal{S}$.

2. We propose a method based on the minimization of an L-statistic of the data and show that under the prescribed assumption, it is able of asymptotically recovering a model with good performance w.r.t. μ, even when $\lambda > 0.5$.

3. We show that the same approach also enjoys finite-sample guarantees in terms of uniform convergence. To this end, we prove 3 novel uniform bounds for the proposed statistical functional. Two bounds are dimension-free and one is variance sensitive. Overall, we show that uniform convergence holds with a rate of about $\mathcal{O}(\log n/n)$.

4. To overcame the computational hardness of the proposed algorithm, we develop a simple and efficient descent method and assess its performance on real-world data-sets.

1.2 How to Read This Thesis

This manuscript is contains 5 chapters in addition to this introduction. At high level, the thesis consists of three parts that can be read independently and that correspond to the research problems described in the previous section.

- The first part is devoted to the exact cluster recovery problem and consists of the 3 chapters. The first chapter serves mostly as a warm-up and presents the case of exact recovery of ellipsoidal shaped clusters. Most results in this chapter are subsumed by the results presented in Chapter 3. The lower bound presented in this chapter holds also in the more general setting of convex clustering in Euclidean spaces. In addition, this chapter introduces a novel algorithmic technique, we name *monocromatic tessellation*, which may be of independent interest.
- The second part consists of Chapter 5 and is about Correlation Clustering with Queries.
- The third part and the final part, consisting of Chapter 6, is devoted to the problem of robust unsupervised learning.

Each chapter is self-contained and can thus be read independently from the others. To this end, each chapter defines its own notation and presents its related work section and appendices.

Chapter 2

Exact recovery of Ellipsoidal Shaped Clusters

The scope of this chapter is mostly to serve as a warm-up for the subsequent two chapters. The main result of this chapter, namely an upper bound on the query complexity for recovering ellipsoidal clusters, is indeed improved in the next chapter. The lower bounds, although simple, are instead state-of-art. The algorithm we present in this chapter is significantly different from those presented in Chapter 3: while less efficient it is based on a novel algorithmic technique, named *monocromatic tessellation*, that we believe may be of independent interest.

2.1 Introduction

Clustering is a central problem of unsupervised learning with a wide range of applications in machine learning and data science. The goal of clustering is to partition a set of points in different groups, so that similar points are assigned to the same group and dissimilar points are assigned to different groups. A basic formulation is the k-clustering problem, in which the input points must be partitioned into k disjoint subsets. A typical example is center-based k-clustering, where the points lie in a metric space and one is interested in recovering k clusters that minimize the distance between the points and the cluster centers. Different variants of this problem, captured by the classic k-center, k-median, and k-means problems, have been extensively studied for several decades Ahmadian et al. [2020]; Gonzalez [1985]; Li and Svensson [2016].

In this chapter we investigate the problem of recovering a latent clustering in the popular semi-supervised active clustering model of Ashtiani et al. Ashtiani et al. [2016]. In this model, we are given a set X of n input points in $\mathbb{R}^d$ and access to an oracle. The oracle answers same-cluster queries (SCQs) with respect to a fixed but unknown k-clustering and tells whether any two given points in X belong to the same cluster or not. The goal is to design efficient algorithms that recover the latent clustering while asking as few oracle queries as possible. Because SCQ queries are natural in crowd-sourcing systems, this model has been extensively studied both in theory Ailon et al. [2018b,c]; Gamlath et al. [2018]; Huleihel et al. [2019]; Mazumdar and Pal [2017]; Mazumdar and Saha [2017b,a]; Saha and Subramanian [2019b]; Vitale et al. [2019] and in practice Firmani et al. [2018]; Gruenheid et al. [2015]; Verroios and Garcia-Molina [2015]; Verroios et al. [2017] — see also Emamjomeh-Zadeh and Kempe [2018] for other types of queries. In their work Ashtiani et al. [2016], Ashtiani et al. showed that by using $\mathcal{O}(\ln n)$ same-cluster queries one can recover the optimal k-means clustering of X in polynomial time, whereas doing so without the queries would be computationally hard. Unfortunately, Ashtiani et al. [2016] relies crucially on a strong separation assumption, called γ-margin condition: for every cluster C there must exist a sphere S_C, centered in the centroid μ_C of C, such that C lies entirely inside S_C and every point not in C is at distance $(1+\gamma)r_C$ from μ_C, where r_C is the radius of S_C. Thus, although Ashtiani et al. [2016] achieves cluster recovery with $\mathcal{O}(\ln n)$ queries, it does so only for a very narrow class of clusterings.

In this chapter we significantly enlarge the class of clusterings that can be efficiently recovered. We do so by relaxing the γ-margin condition of Ashtiani et al. [2016] in two ways (see Section 2.2 for a formal definition). First, we assume that every cluster C has γ-margin

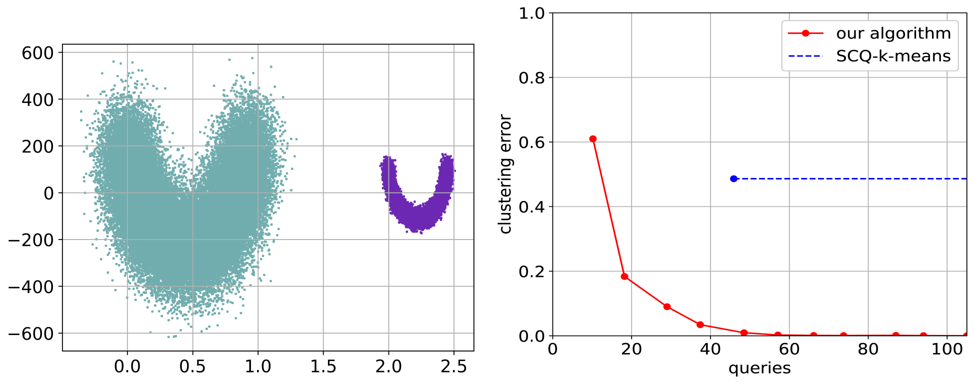


FIGURE 2.1: A toy instance on 10^5 points that we solve exactly with 105 queries, while the SCQ-k-means algorithm of Ashtiani et al. [2016] is no better than random labeling.

in some latent space, obtained by linearly transforming all points according to some unknown positive semi-definite matrix W_C. This is equivalent to assume that C is bounded by an ellipsoid (possibly degenerate) rather than by a sphere (which corresponds to $W_C = I$). This is useful because in many real-world applications the features are on different scales, and so each cluster tends to be distorted along specific directions causing ellipsoids to fit the data better than spheres Dzogang et al. [2012]; Kameyama and Kosugi [1999]; Marica [2014]; Moshtaghi et al. [2011]; Shuhua et al. [2013]. Second, we allow the center of the ellipsoid to lie anywhere in space — in the centroid of C or anywhere else, even outside the convex hull of C. This includes as special cases clusterings in the latent space which are solutions to k-medians, k-centers, or one of their variants. It is not hard to see that this setting captures much more general and challenging scenarios. For example, the latent clustering can be an optimal solution of k-centers where some points have been adversarially deleted and the features adversarially rescaled before the input points are handed to us. In fact, the latent clustering need not be the solution to an optimization problem, and in particular need not be center-based: it can be literally *any* clustering, as long as it respects the margin condition just described.

Our main result is that, even in this significantly more general setting, it is still possible to recover the latent clustering *exactly*, in polynomial time, and using only $\mathcal{O}(\ln n)$ same-cluster queries. The price to pay for this generality is an exponential dependence of the number of queries on the dimension d of the input space; this dependence is however unavoidable, as we show via rigorous lower bounds. Our algorithm is radically different from the one in Ashtiani et al. [2016], which we call SCQ-k-means here. The reason is that SCQ-k-means uses same-cluster queries to estimate the clusters' centroids and find their spherical boundaries via binary search. Under our more general setting, however, the clusters are not separated by spheres centered in their centroids, and thus SCQ-k-means fails, as shown in Figure 2.1 (see Section 2.8 for more experiments). Instead of binary search, we develop a geometric technique, based on careful tessellations of minimum-volume enclosing ellipsoids (MVEEs). The key idea is that MVEEs combine a low VC-dimension, which makes learning easy, with a small volume, which can be decomposed in easily classifiable elements. While MVEEs are not guaranteed to be consistent with the cluster samples, our results can be also proven using consistent ellipsoids that are close to the convex hull of the samples. This notion of low-stretch consistent ellipsoid is new, and may be interesting in its own right.

2.2 Preliminaries and definitions

All missing statements and proofs can be found in the appendix. The input to our problem is a triple (X, k, γ) where $X \subset \mathbb{R}^d$ is a set of n arbitrary points, $k \geq 2$ is an integer, and $\gamma \in \mathbb{R}_{>0}$ is the margin (see below). We assume there exists a latent clustering $\mathcal{C} = \{C_1, \ldots, C_k\}$ over the input set X, which we do not know and want to compute. To this end, we are given access to an oracle answering *same-cluster queries*: a query SCQ$(\boldsymbol{x}, \boldsymbol{x}')$ is answered by $+1$ if

$\boldsymbol{x}, \boldsymbol{x}'$ are in the same cluster of $\mathcal{C}$, and by -1 otherwise. Our goal is to recover $\mathcal{C}$ while using as few queries as possible. Note that, given any subset $S \subseteq X$, with at most $k|S|$ queries one can always learn the label (cluster) of each $\boldsymbol{x} \in S$ up to a relabeling of $\mathcal{C}$, see Ashtiani et al. [2016].

It is immediate to see that if $\mathcal{C}$ is completely arbitrary, then no algorithm can reconstruct $\mathcal{C}$ with less than n queries. Here, we assume some structure by requiring each cluster to satisfy a certain *margin* condition, as follows. Let $W \in \mathbb{R}^{d \times d}$ be some positive semidefinite matrix (possibly different for each cluster). Then W induces the seminorm $\|\boldsymbol{x}\|_W = \sqrt{\boldsymbol{x}^\top W \boldsymbol{x}}$, which in turn induces the pseudo-metric $d_W(\boldsymbol{x}, \boldsymbol{y}) = \|\boldsymbol{x} - \boldsymbol{y}\|_W$. The same notation applies to any other PSD matrix, and when the matrix is clear from the context, we drop the subscript and write simply $d(\cdot, \cdot)$. The margin condition that we assume is the following:

Definition 2.1 (Clustering margin)**.** *A cluster C has margin $\gamma > 0$ if there exist a PSD matrix $W = W(C)$ and a point $\boldsymbol{c} \in \mathbb{R}^d$ such that for all $\boldsymbol{y} \notin C$ and all $\boldsymbol{x} \in C$ we have $d_W(\boldsymbol{y}, \boldsymbol{c}) > \sqrt{1+\gamma}\, d_W(\boldsymbol{x}, \boldsymbol{c})$. If this holds for all clusters, then we say that the clustering $\mathcal{C}$ has margin γ.*

This is our only assumption. In particular, we do not assume the cluster sizes are balanced, or that $\mathcal{C}$ is the solution to an optimization problem, or that points in a cluster C are closer to the center of C than to the centers of other clusters. Note that the matrices W and the points $\boldsymbol{c}$ are unknown to us. The spherical k-means setting of Ashtiani et al. [2016] corresponds to the special case where for every C we have $W = rI$ for some $r = r(C) > 0$ and $\boldsymbol{c} = \boldsymbol{\mu}(C) = \frac{1}{|C|}\sum_{\boldsymbol{x} \in C} \boldsymbol{x}$.

We denote a clustering returned by our algorithm by $\hat{\mathcal{C}} = \{\hat{C}_1, \dots, \hat{C}_k\}$. The quality of $\hat{\mathcal{C}}$ is measured by the disagreement with $\mathcal{C}$ under the best possible relabeling of the clusters, that is, $\triangle(\hat{\mathcal{C}}, \mathcal{C}) = \min_{\sigma \in S_k} \frac{1}{2n} \sum_{i=1}^k |\hat{C}_{\sigma(i)} \Delta C_i|$, where S_k is the set of all permutations of $[k]$. Our goal is to minimize $\triangle(\hat{\mathcal{C}}, \mathcal{C})$ using as few queries as possible. In particular, we characterize the query complexity of exact reconstruction, corresponding to $\triangle(\hat{\mathcal{C}}, \mathcal{C}) = 0$. The *rank* of a cluster C, denoted by $\operatorname{rank}(C)$, is the rank of the subspace spanned by its points.

Finally we recall few definitions and results from PAC learning.

Definition 2.2 (PAC learning)**.** *Let $\mathcal{X}$ an arbitrary set and $\mathcal{H} \subset 2^{\mathcal{X}}$ a family of binary hypothesis on $\mathcal{X}$. $\mathcal{H}$ is said to be PAC learnable if there exists and algorithm A such that for every distribution D on X, every $h^* \in \mathcal{H}$ and every $\epsilon, \delta \in (0,1)$, upon receiving an i.i.d. sample S of size $m = m(\epsilon, \delta)$ from D labelled by h^*, outputs an hypothesis function $h_S \in 2^{\mathcal{X}}$ that satisfies*

$$\mathbb{P}_{S \sim D^m}\left(R_D(h_S) \leq \epsilon\right) \geq 1 - \delta \tag{2.1}$$

where $R_D(h_S) = \mathbb{P}_{x \sim D}\left(x \in h_S \Delta h^*\right)$ is the risk of h_S and m is called query complexity of A.

It is well known that PAC learnability is determined by the VC dimension of $\mathcal{H}$ defined as follows. Given a sample S, the projection of $\mathcal{H}$ on S is defined as $\mathcal{H}_S = \{h \cap S | h \in \mathcal{H}\}$; S is shattered by $\mathcal{H}$ if $\mathcal{H}_S = 2^S$. The VC dimension of $\mathcal{H}$ is defined as the size of the largest subset S of $\mathcal{X}$ that is shattered by $\mathcal{H}$ if it is finite and as ∞ otherwise. The following classical result holds (e.g. [Shalev-Shwartz and Ben-David, 2014c, Theorem 6.8]).

Theorem 2.3. *If $\mathcal{H}$ has finite VC dimension $\hat{d}$, then the algorithm that returns a subset h_S of $\mathcal{H}$ consistent with S, i.e. $h_S \cap S = h^* \cap S$, is a PAC learner for $\mathcal{H}$ and its sample complexity is $\lceil C \frac{\hat{d} + \log(1/\delta)}{\epsilon} \rceil$ for an absolute constant $C > 0$. Such algorithm is called Empirical Risk Minimizer (ERM).*

2.3 Contributions

Our main contribution is an efficient active clustering algorithm, named RECUR, to recover the latent clustering exactly. We show the following.

Theorem 2.4. *Consider any instance (X, k, γ) whose clustering $\mathcal{C}$ has margin γ. Let $n = |X|$, let $r \leq d$ be the maximum rank of a cluster in $\mathcal{C}$, and let $f(r, \gamma) = \max\left\{2^r, \mathcal{O}\big(\frac{r}{\gamma} \ln \frac{r}{\gamma}\big)^r\right\}$. Given (X, k, γ), RECUR with probability 1 outputs $\mathcal{C}$ (up to a relabeling), and with high probability runs in time $\mathcal{O}((k \ln n)(n + k^2 \ln k))$ using $\mathcal{O}\big((k \ln n)\,(k^2 d^2 \ln k + f(r, \gamma))\big)$ same-cluster queries.*

More in general, RECUR clusters correctly $(1-\epsilon)n$ points using $\mathcal{O}\big((k \ln {}^1\!/\!_\epsilon)\,(k^2 d^2 \ln k + f(r,\gamma))\big)$ queries in expectation. Note that the query complexity depends on r rather than on d, which is desirable as real-world data often exhibits a low rank (i.e., every point can be expressed as a linear combination of at most r other points in the same cluster, for some $r \ll d$). In addition, unlike the algorithm of Ashtiani et al. [2016], which is Monte Carlo and thus can fail, RECUR is Las Vegas: it returns the correct clustering with probability 1, and the randomness is only over the number of queries and the running time. Notice that in this setting, the standard transformation from Monte Carlo to Las Vegas is not applicable as it requires the existence of an efficient verifier, i.e. a routine that can assess the correctness of the proposed clustering. Moreover, RECUR is simple to understand and easy to implement. It works by recovering a constant fraction of some cluster at each round, as follows (see Section 2.5 and Section 2.6):

1. **Sampling.** We draw points uniformly at random from X until, for some cluster C, we obtain a sample S_C of size $\simeq d^2$. We can show that with good probability $|C| \simeq \frac{1}{k}|X|$, and that, by standard PAC bounds, any ellipsoid E containing S_C contains at least half of C.
2. **Computing the MVEE.** We compute the MVEE (minimum-volume enclosing ellipsoid) $E = E_{\mathrm{J}}(S_C)$ of S_C. As said, by PAC bounds, E contains at least half of C. If we were lucky, E would not contain any point from other clusters, and $E \cap X$ would be our large subset of C. Unfortunately, E can contain an arbitrarily large number of points from $X \setminus C$. Our goal is to find them and recover $C \cap E$.
3. **Tessellating the MVEE.** To recover $C \cap E$, we partition E into roughly $({}^d\!/\!_\gamma)^d$ hyperrectangles, each one with the property of being monochromatic: its points are either all in C or all in $X \setminus C$. Thanks to this special tessellation, with roughly $({}^d\!/\!_\gamma)^d$ queries we can find all hyperrectangles containing only points of C, and thus compute $C \cap E$.

Our second contribution is to show a family of instances where every algorithm needs roughly $({}^1\!/\!_\gamma)^r$ same-cluster queries to return the correct clustering. This holds even if the algorithm is allowed to fail with constant probability. Together with Theorem 2.4, this gives an approximate characterization of the query complexity of the problem as a function of γ and r. That is, for ellipsoidal clusters, a margin of γ is necessary and sufficient to achieve a query complexity that grows roughly as $({}^1\!/\!_\gamma)^r$. This lower bound also implies that our algorithm is nearly optimal, even compared to algorithms that can fail. The result is given formally in Section 2.7.

Our final contribution is a set of experiments on large synthetic datasets. They show that our algorithm RECUR achieves exact cluster reconstruction efficiently, see Section 2.8.

2.4 Related work.

The semi-supervised active clustering (SSAC) framework was introduced in Ashtiani et al. [2016], together with the SCQ-k-means algorithm that recovers $\mathcal{C}$ using $O(k^2 \ln k + k \ln n)$ same-cluster queries. This is achieved via binary search under assumptions much stronger than ours (see above). In our setting, SCQ-k-means works only when every point $\boldsymbol{c}$ is close to the cluster centroid and the condition number of W is small (see the appendix); indeed, our experiments show that SCQ-k-means fails even when $W \simeq I$. Interestingly, even if binary search and its generalizations are at the core of many active learning techniques Nowak [2011], here they do not seem to help. We remark that we depend on γ in the same way as Ashtiani et al. [2016]: if γ is a lower bound on the actual margin of $\mathcal{C}$, then the correctness is guaranteed, otherwise we may return any clustering. Clustering with same-cluster queries is also studied in Mazumdar and Pal [2017], but they assume stochastic similarities between points that do not necessarily define a metric. Same-cluster queries for center-based clustering in metric spaces were also considered by Sanyal and Das [2019], under α-center proximity Awasthi et al. [2012b] instead of γ-margin (see [Ashtiani et al., 2016, Appendix B] for a comparison between the two notions). Finally, Ailon et al. [2018c] used same-cluster queries to obtain a PTAS for k-means. Unfortunately, this gives no guarantee on the clustering error: a good k-means value can be achieved by a clustering very different from the optimal one, and vice versa. From a more theoretical viewpoint, the problem has been extensively studied for clusters generated by a latent mixture of Gaussians Dasgupta [1999]; Kalai et al. [2010]; Hardt and Price [2015].

As same-cluster queries can be used to label the points, one can also learn the clusters using standard pool-based active learning tools. For example, using quadratic feature expansion, our ellipsoidal clusters can be learned as hyperplanes. Unfortunately, the worst-case label complexity of actively learning hyperplanes with margin $\gamma < 1/2$ is still $\Omega\big((R/\gamma)^d\big)$, where R is the radius of the smallest ball enclosing the points Gonen et al. [2013]. Some approaches that bypass this lower bound have been proposed. In Gonen et al. [2013] they prove an approximation result, showing that $\mathrm{OPT} \times \mathcal{O}\big(d \ln \frac{R}{\gamma}\big)$ queries are sufficient to learn any hyperplane with margin γ, where OPT is the number of queries made by the optimal active learning algorithm. Moreover, under distributional assumptions, linear separators can be learned efficiently with roughly $\mathcal{O}(d \ln n)$ label queries Balcan et al. [2007b]; Balcan and Long [2013b]; Dasgupta [2005]. In a different line of work, Kane et al. [2017b] show that $\mathcal{O}\big((d \ln n) \ln \frac{R}{\gamma}\big)$ queries suffice for linear separators with margin γ when the algorithm can also make comparison queries: for any two pairs of points $(\boldsymbol{x}, \boldsymbol{x}')$ and $(\boldsymbol{y}, \boldsymbol{y}')$ from X, a comparison query returns 1 iff $d_W(\boldsymbol{x}, \boldsymbol{x}') \leq d_W(\boldsymbol{y}, \boldsymbol{y}')$. As we show, comparison queries do not help learning the latent metric d_W using metric learning techniques Kulis [2013] (see the appendix). In general, the query complexity of pool-based active learning is characterized by the star dimension of the family of sets Hanneke and Yang [2015]. This implies that, if we allow for a non-zero probability of failure, then $\mathcal{O}(\mathfrak{s} \ln n)$ queries are sufficient for reconstructing a single cluster, where $\mathfrak{s}$ is the star dimension of ellipsoids with margin γ. To the best of our knowledge, this quantity is not known for ellipsoids with margin (not even for halfspaces with margin), and our results seem to suggest a value of order $(d/\gamma)^d$. If true, this would imply then the general algorithms of Hanneke and Yang [2015] could be used to solve our problem with a number of queries comparable to ours. However, note that our reconstructions are exact with probability one, and are achieved by simple algorithms that work well in practice.

2.5 Recovery of a single cluster with one-sided error

This section describes the core of our cluster recovery algorithm. The main idea is to show that, given any subset $S_C \subseteq C$ of some cluster C, if we compute a small ellipsoid E containing S_C, then we can compute $C \cap E$ deterministically with a small number of queries.

Consider a subset $S_C \subseteq C$, and let $\mathrm{conv}(S_C)$ be its convex hull. The *minimum-volume enclosing ellipsoid* (MVEE) of S_C, also known as Löwner-John ellipsoid and denoted by $E_J(S_C)$, is the volume-minimizing ellipsoid E such that $S_C \subset E$ (see, e.g., Todd [2016]). The main result of this section is that $C \cap E_J(S_C)$ is easy to learn. Formally, we prove:

Theorem 2.5. *Suppose we are given a subset $S_C \subseteq C$, where C is any unknown cluster. Then we can learn $C \cap E_J(S_C)$ using $\max\big\{2^r, \mathcal{O}\big(\frac{r}{\gamma} \ln \frac{r}{\gamma}\big)^r\big\}$ same-cluster queries, where $r = \mathrm{rank}(C)$ and $E_J(S_C)$ is the minimum-volume enclosing ellipsoid of S_C.*

In the rest of the section we show how to learn $C \cap E_J(S_C)$ and sketch the proof of the theorem.

The MVEE. The first idea is to compute an ellipsoid E that is "close" to $\mathrm{conv}(S_C)$. A d-rounding of S_C is any ellipsoid E satisfying the following (we assume the center of E is the origin):

$$\frac{1}{d} E \subseteq \mathrm{conv}(S_C) \subseteq E \tag{2.2}$$

In particular, by a classical theorem by John Khachiyan [1996a], the MVEE $E_J(S_C)$ is a d-rounding of S_C. We therefore let $E = E_J(S_C)$. Note however that *any* d-rounding ellipsoid E can be chosen instead, as the only property we exploit in our proofs is (2.2).

It should be noted that the ambient space dimensionality d can be replaced by $r = \mathrm{rank}(S_C)$. To this end, before computing $E = E_J(S_C)$, we compute the span V of S_C and a canonical basis for it using a standard algorithm (e.g., Gram-Schmidt). We then use V as new ambient space, and search for $E_J(S_C)$ in V. This works since $E_J(S_C) \subset V$, and lowers the dimensionality from d to $r \leq d$. From this point onward we still use d in our notation,

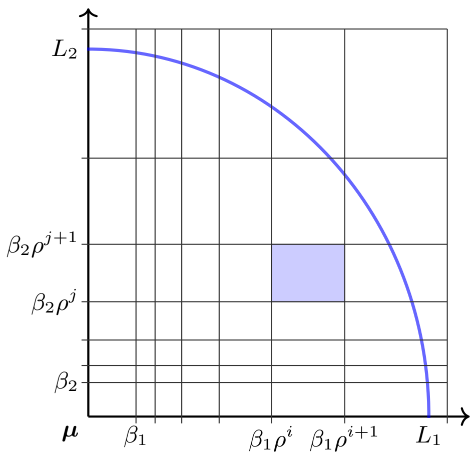


FIGURE 2.2: The tessellation $\mathcal{R}$ of $E\cap\mathbb{R}^d_+$. Every hyperrectangle R (shaded) is such that $R \cap E$ is monochromatic, i.e. contains only points of C or of $X \setminus C$.

but all our constructions and claims hold unchanged if instead one uses r, coherently with the bounds of Theorem 2.5.

The monochromatic tessellation. We now show that, by exploiting the γ-margin condition, we can learn $C\cap E_{\mathrm{J}}(S_C)$ with a small number of queries. We do so by discretizing $E_{\mathrm{J}}(S_C)$ into hyperrectangles so that, for each hyperrectangle, we need only one query to decide if it lies in C or not. The crux is to show that there exists such a discretization, which we call *monochromatic tessellation*, consisting of relatively few hyperrectangles, roughly $(\frac{d}{\gamma}\ln\frac{d}{\gamma})^d$.

Let $E = E_{\mathrm{J}}(S_C)$. To describe the monochromatic tessellation, we first define the notion of monochromatic subset:

Definition 2.6. *A set $B \subset \mathbb{R}^d$ is* monochromatic *with respect to a cluster C if it does not contain two points $\boldsymbol{x}, \boldsymbol{y}$ with $\boldsymbol{x} \in C$ and $\boldsymbol{y} \notin C$.*

Fix a hyperrectangle $R \subset \mathbb{R}^d$. The above definition implies that, if $B = R\cap E$ is monochromatic, then we learn the label of all points in B with a single query. Indeed, if we take any $\boldsymbol{y} \in B$ and any $\boldsymbol{x} \in S_C$, the query $\textsc{scq}(\boldsymbol{y}, \boldsymbol{x})$ tells us whether $\boldsymbol{y} \in C$ or $\boldsymbol{y} \notin C$ simultaneously for all $\boldsymbol{y} \in B$. Therefore, if we can cover E with m monochromatic hyperrectangles, then we can learn $C \cap E$ with m queries. Our goal is to show that we can do so with $m \simeq (\frac{d}{\gamma}\ln\frac{d}{\gamma})^d$.

We now describe the construction in more detail; see also Figure 2.2. The first observation is that, if any two points $\boldsymbol{x}, \boldsymbol{y} \in X$ are such that $\boldsymbol{x} \in C$ and $\boldsymbol{y} \notin C$, then $|x_i - y_i| \gtrsim \gamma/d$ for some i. Indeed, if this was not the case then $\boldsymbol{x}, \boldsymbol{y}$ would be too close and would violate the γ-margin condition. This implies that, for $\rho \simeq 1 + \gamma/d$, any hyperrectangle whose sides have the form $[\beta_i, \beta_i\rho]$ is monochromatic. We can exploit this observation to construct the tessellation. Let the semiaxes of E be the canonical basis for $\mathbb{R}^d$ and its center $\boldsymbol{\mu}$ be the origin. For simplicity, we only consider the positive orthant, the argument being identical for every other orthant. Let L_i be the length of the i-th semiaxis of E. The goal is to cover the interval $[0, L_i]$ along the i-th semiaxis of E with roughly $\log_\rho(L_i/\beta_i)$ intervals of length increasing geometrically with ρ. More precisely, we let $T_i = \big\{[0, \beta_i], (\beta_i, \beta_i\rho], \dots, (\beta_i\rho^{b-1}, \beta_i\rho^b]\big\}$, where $\beta_i > 0$, $\rho > 1$, and $b \geq 0$ are functions of γ and d. Then our tessellation is the cartesian product of all the T_i:

Definition 2.7. *Let $\mathbb{R}^d_+$ be the positive orthant of $\mathbb{R}^d$. The tessellation $\mathcal{R}$ of $E\cap\mathbb{R}^d_+$ is the set of $(b+1)^d$ hyperrectangles expressed in the canonical basis $\{\boldsymbol{u}_1, \dots, \boldsymbol{u}_d\}$ of E: $\mathcal{R} = T_1\times\ldots\times T_d$.*

We now come to the central fact. Loosely speaking, if $\beta_i \simeq \frac{\gamma}{d}L_i$ then the point $(\beta_1, \dots, \beta_d)$ lies "well inside" $\mathrm{conv}(S_C)$, because (2.2) tells us E itself is close to $\mathrm{conv}(S_C)$. By setting ρ, b adequately, then, we can guarantee the intervals of T_i of the form $(\beta_i\rho^{j-1}, \beta_i\rho^j]$ cover all the space between $\mathrm{conv}(S_C)$ and E. More formally we show that, for a suitable choice of β_i, ρ, b, the tessellation $\mathcal{R}$ satisfies the following properties (see the appendix):

(1) $|\mathcal{R}| \le \max\left\{1, \mathcal{O}\left(\frac{d}{\gamma}\ln\frac{d}{\gamma}\right)^d\right\}$

(2) $E \cap \mathbb{R}^d_+ \subseteq \bigcup_{R\in\mathcal{R}} R$

(3) For every $R \in \mathcal{R}$, the set $R \cap E$ is monochromatic w.r.t. C

Once the three properties are established, Theorem 2.5 immediately derives from the discussion above.

Pseudocode. We list below our algorithm that learns $C \cap E$ subject to the bounds of Theorem 2.5. We start by computing $E = E_J(S_C)$ and selecting the subset $E_X = X \cap E$. We then proceed with the tessellation, but without constructing $\mathcal{R}$ explicitly. Note indeed that, for every $\boldsymbol{y} \in E_X$, the hyperrectangle $R(\boldsymbol{y})$ containing $\boldsymbol{y}$ is determined uniquely by $|y_i|/\beta_i$ for all $i \in [d]$. In fact, we can manage all orthants at once by simply looking at y_i/β_i. After grouping all points $\boldsymbol{y}$ by their $R(\boldsymbol{y})$, we repeatedly take a yet-unlabeled R and label it as C or not C. Finally, we return all points in the hyperrectangles labeled as C.

Algorithm 1 TessellationLearn(X, S_C, γ)

```
1: compute $E \leftarrow E_J(S_C)$ or any other $r$-rounding of $S_C$
2: compute $E_X \leftarrow X \cap E$
3: compute $\beta_i, \rho, b$ as a function of $r, \gamma$          ▷ see Figure 2.2
4: for every $\boldsymbol{y} \in E_X$ do
5:     map $\boldsymbol{y}$ to $R(\boldsymbol{y})$
6: $\boldsymbol{x}_C \leftarrow$ any point in $S_C$
7: while there is some unlabeled $R$ do
8:     label$(R) \leftarrow$ SCQ$(\boldsymbol{x}_C, \boldsymbol{y})$, where $\boldsymbol{y}$ is any point s.t. $R(\boldsymbol{y}) = R$
9: return all $\boldsymbol{y}$ mapped to $R$ such that label$(R) = +1$
```

Low-stretch separators. We conclude this section with a technical note. Although the MVEEs enable exact cluster reconstruction, they do not give PAC guarantees since they do not ensure consistency. Indeed, if we draw a sample S from X and let $S_C = S \cap C$, there is no guarantee that $E = E_J(S_C)$ separates S_C from $S \setminus S_C$. On the other hand, any ellipsoid E separating S_C from $S \setminus S_C$ is a good classifier in the PAC sense, but there is no guarantee it will be close to conv(S_C), thus breaking down our algorithm. Interestingly, in the appendix we show that it is possible to compute an ellipsoid that is simultaneously a good PAC classifier *and* close to conv(S_C), yielding essentially the same bounds as Theorem 2.5. Formally, we have:

Definition 2.8. *Given any finite set X in $\mathbb{R}^d$ and a subset $S \subset X$, a Φ-stretch separator for S is any ellipsoid E separating S from $X \setminus S$ and such that $E \subseteq \Phi E_J(S)$.*

Theorem 2.9. *Suppose C has margin $\gamma > 0$ w.r.t. to some $\boldsymbol{z} \in \mathbb{R}^d$ and fix any subset $S_C \subseteq C$. There exists a Φ-stretch separator for S_C with $\Phi = 64\sqrt{2}d^2 \max\left\{125, 1/\gamma^3\right\}$.*

2.6 Exact recovery of all clusters

In this section we conclude the construction of our algorithm RECUR (listed below), and we bound its query complexity and running time. RECUR proceeds in rounds. At each round, it draws samples uniformly at random from X until, for some sufficiently large $b > 0$, it obtains a sample S_C of size $bd^2 \ln k$ from some cluster C. At this point, by concentration and PAC bounds, we know that any ellipsoid E containing S_C satisfies $|C \cap E| \ge \frac{1}{4k}|X|$ with probability at least $1/2$. RECUR uses the routine TessellationLearn() from Section 2.5 to compute such a subset $C \cap E$ efficiently (see Theorem 2.5). RECUR then deletes $C \cap E$ from X and repeats the process on the remaining points. This continues until a fraction $(1-\epsilon)$ of points have been clustered. In particular, when $\epsilon < 1/n$, RECUR clusters all the points of X.

Regarding the correctness of RECUR, we have:

Algorithm 2 RECUR(X, k, γ, ϵ)

```
1: $\hat{C}_1, \dots, \hat{C}_k \leftarrow \emptyset$
2: while $|X| > \epsilon n$ do
3:     draw samples with replacement from $X$ until $|S_C| \geq bd^2 \ln k$ for some $C$
4:     $C_E \leftarrow$ TessellationLearn$(X, S_C, \gamma)$
5:     add $C_E$ to the corresponding $\hat{C}_i$
6:     $X \leftarrow X \setminus C_E$
7: return $\hat{\mathcal{C}} = \{\hat{C}_1, \dots, \hat{C}_k\}$
```

Lemma 2.10. *The clustering $\hat{\mathcal{C}}$ returned by* RECUR(X, k, γ, ϵ) *satisfies $\triangle(\hat{\mathcal{C}}, \mathcal{C}) \leq \epsilon$. In particular, for $\epsilon < 1/n$ we have $\triangle(\hat{\mathcal{C}}, \mathcal{C}) = 0$.*

This holds because $\triangle(\hat{\mathcal{C}}, \mathcal{C})$ is bounded by the fraction of points that are still in X when RECUR returns; and this fraction is at most ϵ by construction. Regarding the cost of RECUR, we have:

Lemma 2.11. RECUR(X, k, γ, ϵ) *makes $\mathcal{O}(k^3 \ln k \ln(1/\epsilon))$ same-cluster queries in expectation, and for all fixed $a \geq 1$,* RECUR$(X, k, \gamma, 0)$ *with probability at least $1 - n^{-a}$ makes $\mathcal{O}(k^3 \ln k \ln n)$ same-cluster queries and runs in time $\mathcal{O}((k \ln n)(n + k^2 \ln k)) = \widetilde{\mathcal{O}}(kn + k^3)$.*

In the rest of the section we sketch the proof of Lemma 2.11. We start by bounding the number of rounds performed by RECUR. Recall that, at each round, with probability at least $1/2$ a fraction at least $1/4k$ of points are labeled and removed. Thus, at each round, the size of X drops by $(1 - 1/8k)$ in expectation. Hence, we need roughly $8k \ln(1/\epsilon)$ rounds before the size of X drops below ϵn. Indeed, we prove:

Lemma 2.12. RECUR*(X, k, γ, ϵ) makes at most $8k \ln(1/\epsilon)$ rounds in expectation, and for all fixed $a \geq 1$,* RECUR*$(X, k, \gamma, 0)$ with probability at least $1 - n^{-a}$ performs at most $(8k + 6a\sqrt{k}) \ln n$ rounds.*

We can now bound the query cost and running time of RECUR, by counting the work done at each round and using Lemma 2.12. To simplify the discussion we treat d, r, γ as constants, but fine-grained bounds can be derived immediately from the discussion itself.

Query cost of RECUR**.** The algorithm makes queries at line 3 and line 4. At line 3, RECUR draws at most $bkd^2 \ln k = \mathcal{O}(k \ln k)$ samples. This holds since there are at most k clusters, so after $bkd^2 \ln k$ samples, the condition $|S_C| \geq bd^2 \ln k$ will hold for some C. Since learning the label of each sample requires at most k queries, line 3 makes $\mathcal{O}(k^2 \ln k)$ queries in total. At line 4, RECUR makes $f(d, \gamma) = \mathcal{O}(1)$ queries by Theorem 2.5. Together with Lemma 2.12, this implies that RECUR with probability at least $1 - n^{-a}$ makes at most $\mathcal{O}(k \ln n) \times \mathcal{O}(k^2 \ln k) = \mathcal{O}(k^3 \ln k \ln n)$ queries.

Running time of RECUR**.** Line 3 takes time $\mathcal{O}(k^2 \ln k)$, see above. The rest of each round is dominated by the invocation of TessellationLearn at line 4. Recall then the pseudocode of TessellationLearn from Section 2.5. At line 1, computing $E = E_J(S_C)$ or any r-rounding of S_C takes time $\mathcal{O}(|S_C|^{3.5} \ln |S_C|)$, see Khachiyan [1996a].[1] This is in $\widetilde{\mathcal{O}}(1)$ since by construction $|S_C| = \mathcal{O}(d^2 \ln k) = \widetilde{\mathcal{O}}(1)$. Computing $E_X = X \cap E$ takes time $\mathcal{O}(|X| \operatorname{poly}(d)) = \mathcal{O}(n)$. For the index (line 4), we can build in time $\mathcal{O}(|X \cap E|)$ a dictionary that maps every $R \in \mathcal{R}$ to the set $R \cap E_X$. The classification part (line 7) takes time $|\mathcal{R}| = \mathcal{O}(1)$. Finally, enumerating all positive R and concatenating the list of their points takes again time $\mathcal{O}(|X \cap E| \operatorname{poly}(d))$. By the rounds bound of Lemma 2.12, then, RECUR with probability at least $1 - n^{-a}$ runs in time $\mathcal{O}((k \ln n)(n + k^2 \ln k))$.

2.7 Lower bounds

We show that any algorithm achieving exact cluster reconstruction must, in the worst case, perform a number of same-cluster queries that is exponential in d (the well-known "curse of dimensionality"). Formally, in the appendix we prove:

[1] More precisely, for a set S an ellipsoid E such that $\frac{1}{(1+\epsilon)d} E \subset \operatorname{conv}(S) \subset E$ can be computed in $\mathcal{O}(|S|^{3.5} \ln(|S|/\epsilon))$ operations in the real number model of computation, see Khachiyan [1996a].

Theorem 2.13. *Choose any possibly randomized learning algorithm. There exist:*

1. *for all $\gamma \in (0, 1/7)$ and $d \geq 2$, an instance on $n = \Omega\big((\frac{1+\gamma}{8\gamma})^{\frac{d-1}{2}}\big)$ points and 3 clusters*
2. *for all $\gamma > 0$ and $d \geq 48(1+\gamma)^2$, an instance on $n = \Omega\big(e^{\frac{d}{48(1+\gamma)^2}}\big)$ points and 2 clusters*

such that (i) the latent clustering $\mathcal{C}$ has margin γ, and (ii) to return with probability $2/3$ a $\hat{\mathcal{C}}$ such that $\triangle(\hat{\mathcal{C}}, \mathcal{C}) = 0$, the algorithm must make $\Omega(n)$ same-cluster queries in expectation.

The lower bound uses two different constructions, each one giving a specific instance distribution where any algorithm must perform $\Omega(n)$ queries in expectation, where n is exponential in d as in the statement of the theorem. The first construction is similar to the one shown in Gonen et al. [2013]. The input set X is a packing of $\simeq (1/\gamma)^d$ points on the d-dimensional sphere, at distance $\simeq \sqrt{\gamma}$ from each other. We show that, for $\boldsymbol{x} = (x_1, \dots, x_d) \in X$ drawn uniformly at random, setting $W = (1+\gamma)\,\mathrm{diag}(x_1^2, \dots, x_d^2)$ makes $\boldsymbol{x}$ an outlier. That is, $X \setminus \{\boldsymbol{x}\}$ forms a first cluster C_1, and $\{\boldsymbol{x}\}$ forms a second cluster C_2, and both clusters satisfy the margin condition. In order to output the correct clustering, any algorithm must find $\boldsymbol{x}$, which requires $\Omega(n)$ queries in expectation. In the second construction, X is a random sample of $n \simeq \exp(d/(1+\gamma)^2)$ points from the d-dimensional hypercube $\{0,1\}^d$ such that each coordinate is independently 1 with probability $\simeq \frac{1}{1+\gamma}$. Similarly to the first construction we show that, for $\boldsymbol{x} \in X$ drawn uniformly at random, setting $W = (1+\gamma)\,\mathrm{diag}(x_1, \dots, x_d)$ makes $\boldsymbol{x}$ an outlier, and any algorithm needs $\Omega(n)$ queries to find it.

2.8 Experiments

We implemented our algorithm RECUR and compared it against SCQ-k-means Ashtiani et al. [2016]. To this end, we generated four synthetic instances on $n = 10^5$ points with increasing dimension $d = 2, 4, 6, 8$. The latent clusterings consist of $k = 5$ ellipsoidal clusters of equal size, each one with margin $\gamma = 1$ w.r.t. a random center and a random PSD matrix with condition number $\kappa = 100$, making each cluster stretched by $10\times$ in a random direction. To account for an imperfect knowledge of the data, we fed RECUR with a value of $\gamma = 10$ (thus, it could in principle output a wrong clustering). We also adopted for RECUR the batch sampling of SCQ-k-means, i.e., we draw $k \cdot 10$ samples in each round; this makes RECUR slightly less efficient than with its original sampling scheme (see line 3).

To further improve the performance of RECUR, we use a simple "greedy hull expansion" heuristic that can increase the number of points recovered at each round without performing additional queries. Immediately after taking the sample S_C, we repeatedly expand its convex hull $\mathrm{conv}(S_C)$ by a factor $\simeq (1 + \gamma/d)$, and add all the points that fall inside it to S_C. If C is sufficiently dense, a substantial fraction of it will be added to S_C; while, by the margin assumption, no point outside C will ever be added to S_C (see the proof of the tessellation). This greedy hull expansion is repeated until no new points are found, in which case we proceed to compute the MVEE and the tessellation.

Figure 2.3 shows for both algorithms the clustering error $\triangle$ versus the number of queries, round by round, averaged over 10 independent runs (SCQ-k-means has a single measurement since it runs "in one shot"). The run variance is negligible and we do not report it. Observe that the error of SCQ-k-means is always in the range 20%–40%. In contrast, the error of RECUR decreases exponentially with the rounds until the latent clustering is exactly recovered, as predicted by our theoretical results. To achieve $\triangle \leq .05$, RECUR uses less than 3% of the queries needed by a brute force labeling, which is $kn = 5\times10^5$. Note that, except when clusters are aligned as in Figure 2.1, SCQ-k-means continues to perform poorly even after whitening the input data to compensate for skewness. Finally, note how the number of queries issued by RECUR increases with the dimensionality d, in line with Theorem 2.13.

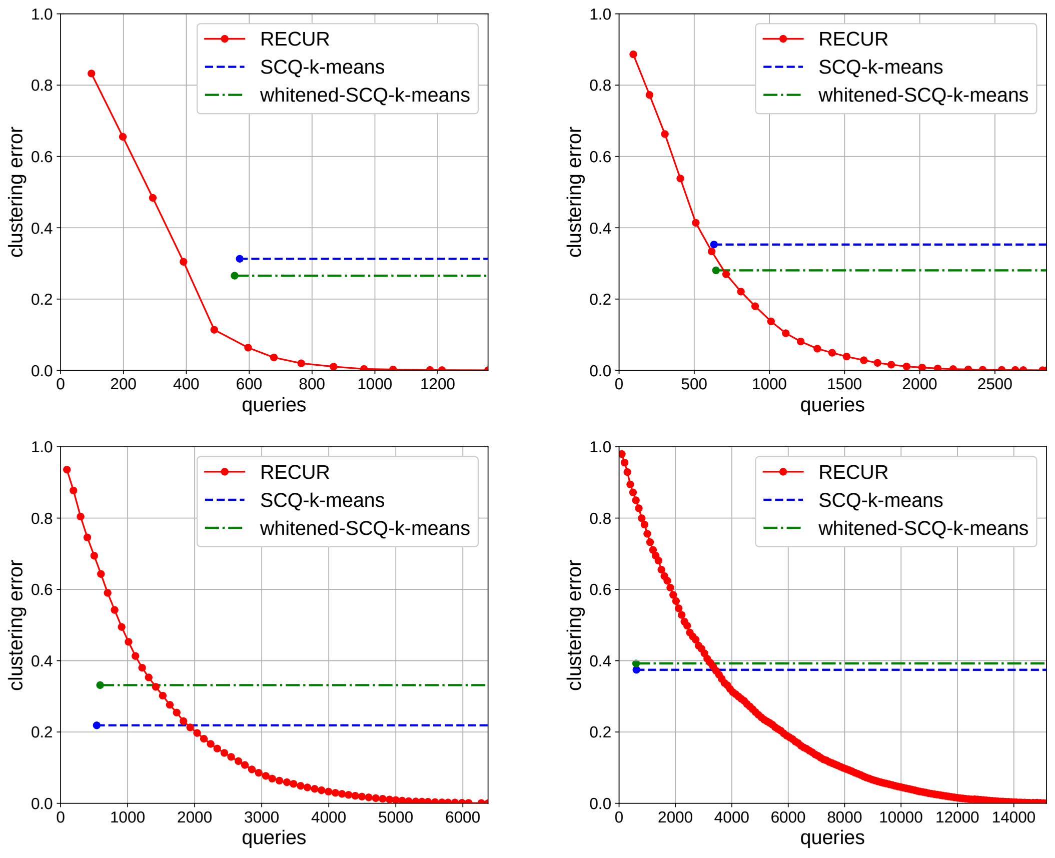


FIGURE 2.3: Clustering error vs. number of queries for $k = 5$ and $d = 2, 4, 6, 8$ (left to right, top to bottom). While SCQ-k-means performs rather poorly, RECUR always achieves exact reconstruction.

Appendix

2.A Ancillary results

2.A.1 VC-dimension of ellipsoids

For any PSD matrix M, we denote by $E_M = \{\boldsymbol{x} \in \mathbb{R}^d : d_M(\boldsymbol{x}, \boldsymbol{\mu}) \leq 1\}$ the $\boldsymbol{\mu}$-centered ellipsoid with semiaxes of length $\lambda_1^{-1/2}, \dots, \lambda_d^{-1/2}$, where $\lambda_1, \dots, \lambda_d \geq 0$ are the eigenvalues of M. We recall the following classical VC-dimension bound (see, e.g., Fournier and Teytaud [2011]).

Theorem 2.14. *The VC-dimension of the class $\mathcal{H} = \{E_M : M \in \mathbb{R}^d, M \succeq 0\}$ of (possibly degenerate) ellipsoids in $\mathbb{R}^d$ is $\frac{d^2+3d}{2}$.*

2.A.2 Generalization error bounds

The next result is a simple adaptation of the classical VC bound for the realizable case (see, e.g., [Shalev-Shwartz and Ben-David, 2014c, Theorem 6.8]).

Theorem 2.15. *There exists a universal constant $c > 0$ such that for any family $\mathcal{H}$ of measurable sets $E \subset \mathbb{R}^d$ of VC-dimension $d < \infty$, any probability distribution $\mathcal{D}$ on $\mathbb{R}^d$, and any $\epsilon, \delta \in (0,1)$, if S is a sample of $m \geq c\frac{d\ln(1/\epsilon)+\ln(1/\delta)}{\epsilon}$ points drawn i.i.d. from $\mathcal{D}$, then for any $E^* \in \mathcal{H}$ we have:*

$$\mathcal{D}\big(E \Delta E^*\big) \leq \epsilon \qquad \text{and} \qquad \mathcal{D}\big(E' \setminus E^*\big) \leq \epsilon$$

with probability at least $1-\delta$ with respect to the random draw of S, where E is any element of $\mathcal{H}$ such that $E \cap S = E^* \cap S$, and E' is any element of $\mathcal{H}$ such that $E^* \cap S \subseteq E' \cap S$.

The first inequality is the classical PAC bound for the zero-one loss, which uses the fact that the VC dimension of $\{E \Delta E^* : E \in \mathcal{H}\}$ is the same as the VC dimension of $\mathcal{H}$. The second inequality follows immediately from the same proof by noting that, for any $E^* \in \mathcal{H}$ the VC dimension of $\{E \setminus E^* : E \in \mathcal{H}\}$ is not larger than the VC dimension of $\mathcal{H}$ because, for any sample S and for any $F, G \in \mathcal{H}$, $(F \setminus E^*) \cap S \neq (G \setminus E^*) \cap S$ implies $F \cap S \neq G \cap S$.

2.A.3 Concentration bounds

We recall standard concentration bounds for non-positively correlated binary random variables, see Dubhashi and Panconesi [2009a]. Let $X_1, \dots, X_n$ be binary random variables. We say that $X_1, \dots, X_n$ are non-positively correlated if for all $I \subseteq \{1, \dots, n\}$ we have:

$$\mathbb{P}\big(\forall i \in I : X_i = 0\big) \leq \prod_{i \in I} \mathbb{P}(X_i = 0) \quad \text{and} \quad \mathbb{P}\big(\forall i \in I : X_i = 1\big) \leq \prod_{i \in I} \mathbb{P}(X_i = 1) \tag{2.3}$$

Lemma 2.16 (Chernoff bounds)**.** *Let $X_1, \dots, X_n$ be non-positively correlated binary random variables. Let $a_1, \dots, a_n \in [0,1]$ and $X = \sum_{i=1}^n a_i X_i$. Then, for any $\epsilon > 0$, we have:*

$$\mathbb{P}\big(X < (1-\epsilon)\mathbb{E}[X]\big) < e^{-\frac{\epsilon^2}{2}\mathbb{E}[X]} \tag{2.4}$$

$$\mathbb{P}\big(X > (1+\epsilon)\mathbb{E}[X]\big) < e^{-\frac{\epsilon^2}{2+\epsilon}\mathbb{E}[X]} \tag{2.5}$$

2.A.4 Yao's minimax principle

We recall Yao's minimax principle for Monte Carlo algorithms. Let $\mathcal{A}$ be a finite family of deterministic algorithms and $\mathcal{I}$ a finite family of problem instances. Fix any two distributions $\boldsymbol{p}$ over $\mathcal{I}$ and $\boldsymbol{q}$ over $\mathcal{A}$, and any $\delta \in [0, 1/2]$. Let $\min_{A\in\mathcal{A}} \mathbb{E}_{I\sim\boldsymbol{p}}[C_\delta(I,A)]$ be the minimum, over every algorithm A that fails with probability at most δ over the input distribution $\boldsymbol{p}$, of the expect cost of A over the input distribution itself. Similarly, let $\max_{I\in\mathcal{I}} \mathbb{E}_{A\sim\boldsymbol{q}}[C_\delta(I,A)]$ be the expected cost of the randomized algorithm defined by $\boldsymbol{q}$ under its worst input from $\mathcal{I}$, assuming it fails with probability at most δ. Then (see Motwani and Raghavan [1995], Proposition 2.6):

$$\max_{I\in\mathcal{I}} \mathbb{E}_{\boldsymbol{q}}[C_\delta(I,A)] \geq \frac{1}{2} \min_{A\in\mathcal{A}} \mathbb{E}_{\boldsymbol{p}}[C_{2\delta}(I,A)] \tag{2.6}$$

2.B Supplementary material for Section 2.5

2.B.1 Monochromatic Tessellation

We give a formal version of the claim about the monochromatic tessellation of Section 2.5:

Theorem 2.17. *Suppose we are given an ellipsoid E such that $\frac{1}{d\Phi}E \subset \mathrm{conv}(S_C) \subset E$ for some stretch factor $\Phi > 0$. Then for a suitable choice of β_i, ρ, b, the tessellation $\mathcal{R}$ of the positive orthant of E (Definition 2.7) satisfies:*

(1) $|\mathcal{R}| \leq \max\left\{1, O\left(\frac{d\Phi}{\gamma}\ln\frac{d\Phi}{\gamma}\right)^d\right\}$

(2) $E \cap \mathbb{R}^d_+ \subseteq \cup_{R\in\mathcal{R}} R$

(3) for every $R \in \mathcal{R}$, the set $R \cap E$ is monochromatic

In order to prove Theorem 2.17, we define the tessellation and prove properties (1-3) for $\gamma \leq 1/2$. For $\gamma > \frac{1}{2}$ the tessellation is defined as for $\gamma = \frac{1}{2}$, and one can check all properties still hold. In the proof we use a constant $c = \sqrt{5}$ and assume $\gamma < c^2 - 2c$, which is satisfied since $c^2 - 2c = 5 - 2\sqrt{5} > 1/2$.

First of all, we define the intervals T_i. The base i-th coordinate is:

$$\beta_i = \frac{\gamma}{c\sqrt{2d}} \frac{L_i}{\Phi d} \tag{2.7}$$

Note that, for all i,

$$\frac{L_i}{\beta_i} = \frac{\Phi c d \sqrt{2d}}{\gamma} \tag{2.8}$$

Define:

$$\alpha = \frac{\gamma}{c\sqrt{2}\Phi d} \tag{2.9}$$

and let:

$$b = \max\left(0, \left\lceil \log_{1+\alpha}\left(\frac{c\Phi d\sqrt{2d}}{\gamma}\right)\right\rceil\right) \tag{2.10}$$

(The parameter ρ of the informal description of Section 2.5 is exactly $1+\alpha$). Finally, define the interval set along the i-th axis as:

$$T_i = \begin{cases} \left\{[0,\beta_i]\right\} & \text{if } b = 0 \\ \left\{[0,\beta_i], \left(\beta_i, \beta_i(1+\alpha)\right], \ldots, \left(\beta_i(1+\alpha)^{b-1}, \beta_i(1+\alpha)^b\right]\right\} & \text{if } b \geq 1 \end{cases} \tag{2.11}$$

Proof of (1). By construction, $|T_i| = b+1$. Thus, $|\mathcal{R}| = \prod_{i\in[d]} |T_i| = (b+1)^d$. Thus, if $b = 0$ then $|\mathcal{R}| = 1$, else by (2.10) and 2.8,

$$\begin{aligned} b &= \left\lceil \frac{\ln\left(\frac{c\Phi d\sqrt{2d}}{\gamma}\right)}{\ln(1+\alpha)} \right\rceil && (2.12)\\ &\le \left\lceil \frac{2}{\alpha} \ln\left(\frac{c\Phi d\sqrt{2d}}{\gamma}\right) \right\rceil && \text{since } \ln(1+\alpha) \ge {}^{\alpha}/_{2} \text{ as } \alpha \le 1 \quad (2.13)\\ &= \left\lceil \frac{2\sqrt{2}c\Phi d}{\gamma} \ln \frac{c\Phi d\sqrt{2d}}{\gamma} \right\rceil && \text{definition of } \alpha \quad (2.14)\\ &= O\left(\frac{d\Phi}{\gamma} \ln \frac{d\Phi}{\gamma}\right) && \text{since } d\Phi \ge 1, \gamma \le {}^{1}/_{2} \quad (2.15) \end{aligned}$$

in which case $|\mathcal{R}| = O\left(\frac{d\Phi}{\gamma} \ln \frac{d\Phi}{\gamma}\right)^d$. Taking the maximum over the two cases proves the claim.

Proof of (2). We show for any $\boldsymbol{x} \in E \cap \mathbb{R}^d_+$ there exists $R \in \mathcal{R}$ containing $\boldsymbol{x}$. Clearly, if $\boldsymbol{x} \in E \cap \mathbb{R}^d_+$, then $\langle \boldsymbol{x}, \boldsymbol{u}_i \rangle \in [0, L_i]$ for all $i \in [d]$. But T_i covers, along the i-th direction $\boldsymbol{u}_i$, the interval from 0 to

$$\beta_i(1+\alpha)^b = \beta_i(1+\alpha)^{\max(0,\lceil \log_{1+\alpha}(L_i/\beta_i)\rceil)} \ge \beta_i(1+\alpha)^{\lceil \log_{1+\alpha}(L_i/\beta_i)\rceil} \ge L_i \tag{2.16}$$

Therefore some $R \in \mathcal{R}$ contains $\boldsymbol{x}$.

Proof of (3). Given any hyperrectangle $R \in \mathcal{R}$, we show that the existence of $\boldsymbol{x}, \boldsymbol{y} \in R \cap E$ with $\boldsymbol{x} \in C$ and $\boldsymbol{y} \notin C$ leads to a contradiction. For the sake of the analysis we conventionally set the origin at the center $\boldsymbol{\mu}$ of E, i.e. we assume $\boldsymbol{\mu} = \boldsymbol{0}$.

We define $E_{\text{in}} = \frac{1}{\Phi d} E$ and let $M = U \Lambda U^\top$ be its PSD matrix, where $U = [\boldsymbol{u}_1, \dots, \boldsymbol{u}_d]$ and $\Lambda = \operatorname{diag}(\lambda_1, \dots, \lambda_d)$. Note that $\lambda_i = \frac{1}{\ell_i^2} = \frac{\Phi^2 d^2}{L_i^2}$ where $\ell_i = \frac{L_i}{\Phi d}$ is the length of the i-th semiaxis of E_{in}. For any $R \in \mathcal{R}$, let R_i be the projection of R on $\boldsymbol{u}_i$ (i.e. R_i is one of the intervals of T_i defined in (2.11)). Let $D = D(R) = \{i \in [d] : 0 \notin R_i\}$. We let U_D and $U_{\neg D}$ be the matrices obtained by zeroing out the columns of U corresponding to the indices in $[d] \setminus D$ and D, respectively. Observe that if $\boldsymbol{x}, \boldsymbol{y} \in R \cap E$ then:

$$\begin{aligned} \langle \boldsymbol{x} - \boldsymbol{y}, \boldsymbol{u}_i \rangle^2 &< \alpha^2 \langle \boldsymbol{x}, \boldsymbol{u}_i \rangle^2 && \forall i \in D && (2.17)\\ \langle \boldsymbol{x} - \boldsymbol{y}, \boldsymbol{u}_i \rangle^2 &\le \beta_i^2 && \forall i \notin D && (2.18) \end{aligned}$$

Now suppose C has margin at least γ for some $\gamma \in (0, c^2 - 2c]$, and suppose $\boldsymbol{x}, \boldsymbol{y} \in R \cap E$ with $\boldsymbol{x} \in C$ and $\boldsymbol{y} \notin C$. Through a set of ancillary lemmata proven below, this leads to the absurd:

$$\begin{aligned} \frac{\gamma^2}{c^2} &< d_W(\boldsymbol{y}, \boldsymbol{x})^2 && \text{Lemma 2.18} && (2.19)\\ &\le d_M(\boldsymbol{y}, \boldsymbol{x})^2 && \text{Lemma 2.19} && (2.20)\\ &< \alpha^2 d_M(\boldsymbol{x}, \boldsymbol{\mu})^2 + \frac{\gamma^2}{2c^2} && \text{Lemma 2.20} && (2.21)\\ &\le \frac{\gamma^2}{2c^2} + \frac{\gamma^2}{2c^2} && \text{Lemma 2.21} && (2.22) \end{aligned}$$

In the rest of the proof we prove the four lemmata.

Lemma 2.18. $\frac{\gamma}{c} < d_W(\boldsymbol{y}, \boldsymbol{x})$.

Proof. Let $\boldsymbol{z}$ be the point w.r.t. which the margin of C holds. By the margin assumption,

$$d_W(\boldsymbol{y}, \boldsymbol{z}) > \sqrt{1+\gamma} \qquad \text{and} \qquad d_W(\boldsymbol{x}, \boldsymbol{z}) \le 1 \tag{2.23}$$

By the triangle inequality then,

$$d_W(\boldsymbol{y}, \boldsymbol{x}) \ge d_W(\boldsymbol{y}, \boldsymbol{z}) - d_W(\boldsymbol{x}, \boldsymbol{z}) > \sqrt{1+\gamma} - 1 \tag{2.24}$$

One can check that for $\gamma \leq c^2 - 2c$ we have $1 + \gamma \geq (1 + \frac{\gamma}{c})^2$. Therefore

$$d_W(\boldsymbol{y}, \boldsymbol{x}) > \sqrt{(1 + \gamma/c)^2} - 1 = \frac{\gamma}{c} \tag{2.25}$$

as desired. $\square$

Lemma 2.19. $d_W(\cdot) \leq d_M(\cdot)$.

Proof. By the assumptions of the theorem, $E_{\text{in}} \subseteq \text{conv}_{\boldsymbol{\mu}}(C)$. Moreover, by the assumptions on $d_W(\cdot)$, the unit ball of $d_W(\cdot)$ contains $\text{conv}(C)$. Thus, the unit ball of $d_W(\cdot)$ contains the unit ball of $d_M(\cdot)$. This implies $W \preceq M$, thus $\|\cdot\|_W \leq \|\cdot\|_M$ and $d_W(\cdot) \leq d_M(\cdot)$. $\square$

Lemma 2.20. $d_M(\boldsymbol{y}, \boldsymbol{x})^2 < \alpha^2 d_M(\boldsymbol{x}, \boldsymbol{\mu})^2 + \frac{\gamma^2}{2c^2}$.

Proof. We decompose $d_M(\boldsymbol{y}, \boldsymbol{x})^2$ along the colspaces of U_D and $U_{\neg D}$:

$$\begin{aligned} d_M(\boldsymbol{y}, \boldsymbol{x})^2 &= \|M^{1/2}(\boldsymbol{y} - \boldsymbol{x})\|_2^2 && (2.26) \\ &= \|M^{1/2}(\boldsymbol{y} - \boldsymbol{x})\|^2_{U_D U_D^\top} + \|M^{1/2}(\boldsymbol{y} - \boldsymbol{x})\|^2_{U_{\neg D} U_{\neg D}^\top} && (2.27) \end{aligned}$$

Next, we bound the two terms of (2.27). To this end, we need to show that for all $D \subseteq [d]$ and $\boldsymbol{v} \in \mathbb{R}^d$:

$$\|M^{1/2}\boldsymbol{v}\|^2_{U_D U_D{}^\intercal} = \sum_{i \in D} \lambda_i \langle \boldsymbol{v}, \boldsymbol{u}_i \rangle^2 \tag{2.28}$$

Let indeed $J_D = \text{diag}(\mathbf{1}_D)$ be the selection matrix corresponding to the indices of D. Then $U_D = UJ_D$, and so $U^\intercal U_D = U^\intercal U J_D = J_D$. This gives:

$$\begin{aligned} \|M^{1/2}\boldsymbol{v}\|^2_{U_D U_D{}^\intercal} &= \boldsymbol{v}^\intercal (U\Lambda^{1/2}U^\intercal) U_D U_D{}^\intercal (U\Lambda^{1/2}U^\intercal)\boldsymbol{v} && \text{definition of } M \text{ and } \|\cdot\|. && (2.29) \\ &= \boldsymbol{v}^\intercal U\Lambda^{1/2} J_D J_D \Lambda^{1/2} U^\intercal \boldsymbol{v} && \text{since } U^\intercal U_D = J_D && (2.30) \\ &= \boldsymbol{v}^\intercal U J_D \Lambda^{1/2}\Lambda^{1/2} J_D U^\intercal \boldsymbol{v} && \text{since } \Lambda, J_D \text{ are diagonal} && (2.31) \\ &= \boldsymbol{v}^\intercal U_D \Lambda U_D{}^\intercal \boldsymbol{v} && \text{since } UJ_D = U_D && (2.32) \\ &= \|U_D{}^\intercal \boldsymbol{v}\|^2_\Lambda && \text{by definition} && (2.33) \\ &= \sum_{i \in D} \lambda_i \langle \boldsymbol{v}, \boldsymbol{u}_i \rangle^2 && && (2.34) \end{aligned}$$

Now we can bound the first term of (2.27):

$$\begin{aligned} \|M^{1/2}(\boldsymbol{y} - \boldsymbol{x})\|^2_{U_D U_D^\top} &= \sum_{i \in D} \lambda_i \langle \boldsymbol{y} - \boldsymbol{x}, \boldsymbol{u}_i \rangle^2 && \text{by (2.34)} && (2.35) \\ &< \alpha^2 \sum_{i \in D} \lambda_i \langle \boldsymbol{x}, \boldsymbol{u}_i \rangle^2 && \text{by (2.17)} && (2.36) \\ &= \alpha^2 \|M^{1/2}\boldsymbol{x}\|^2_{U_D U_D^\top} && \text{by (2.34)} && (2.37) \\ &\leq \alpha^2 \|M^{1/2}\boldsymbol{x}\|^2_{UU^\top} && && (2.38) \\ &= \alpha^2 \|M^{1/2}\boldsymbol{x}\|_2^2 && \text{since } UU^\top = I && (2.39) \\ &= \alpha^2 d_M^2(\boldsymbol{x}, \boldsymbol{\mu}) && \text{since } \boldsymbol{\mu} = \mathbf{0} && (2.40) \end{aligned}$$

And for the second term of (2.27), we have:

$$\begin{aligned} \|M^{1/2}(\boldsymbol{y} - \boldsymbol{x})\|^2_{U_{\neg D} U_{\neg D}^\top} &= \sum_{i \notin D} \lambda_i \langle \boldsymbol{y} - \boldsymbol{x}, \boldsymbol{u}_i \rangle^2 && \text{by (2.34)} && (2.41) \\ &\leq \sum_{i \notin D} \lambda_i \beta_i^2 && \text{by (2.18)} && (2.42) \\ &= \sum_{i \notin D} \frac{\Phi^2 d^2}{L_i^2} \left(\frac{\gamma}{c\sqrt{2d}} \frac{L_i}{\Phi d} \right)^2 && \text{by definition of } \lambda_i \text{ and } \beta_i && (2.43) \end{aligned}$$

$$= \sum_{i \notin D} \frac{\gamma^2}{2dc^2} \tag{2.44}$$

$$\leq \frac{\gamma^2}{2c^2} \tag{2.45}$$

Summing the bounds on the two terms shows that $d_M(\boldsymbol{y}, \boldsymbol{x})^2 < \alpha^2 d_M(\boldsymbol{x}, \boldsymbol{\mu})^2 + \frac{\gamma^2}{2c^2}$, as claimed. □

Lemma 2.21. $\alpha^2 d_M(\boldsymbol{x}, \boldsymbol{\mu})^2 \leq \frac{\gamma^2}{2c^2}$.

Proof. By construction we have $\boldsymbol{x} \in E$ and $E = \Phi d \cdot E_{\text{in}}$. Therefore $\frac{1}{\Phi d}\boldsymbol{x} \in E_{\text{in}}$, that is:

$$1 \geq d_M\left(\frac{1}{\Phi d}\boldsymbol{x}, \boldsymbol{\mu}\right)^2 = \frac{1}{\Phi^2 d^2} d_M(\boldsymbol{x}, \boldsymbol{\mu})^2 \tag{2.46}$$

where we used the fact that $d_M(\cdot, \boldsymbol{\mu})^2 = \|\cdot\|_M^2$ since $\boldsymbol{\mu} = \boldsymbol{0}$. Rearranging terms, this proves that $d_M(\boldsymbol{x}, \boldsymbol{\mu})^2 \leq \Phi^2 d^2$. Multiplying by α^2, we obtain:

$$\alpha^2 d_M(\boldsymbol{x}, \boldsymbol{\mu})^2 \leq \left(\frac{\gamma}{\sqrt{2}c\Phi d}\right)^2 \Phi^2 d^2 = \frac{\gamma^2}{2c^2} \tag{2.47}$$

as desired. □

The proof of the theorem is complete.

2.B.2 Low-stretch separators and proof of Theorem 2.9

In this section we show how to compute the separator of Theorem 2.9. In fact, computing the separator is easy; the nontrivial part is Theorem 2.9 itself, that is, showing that such a separator always exists.

To compute the separator we first compute the MVEE $E_{\mathrm{J}} = (M^\star, \boldsymbol{\mu}^\star)$ of S_C (see Section 2.5). We then solve the following semidefinite program:

$$\begin{aligned} & \max_{\alpha \in \mathbb{R}, \boldsymbol{\mu} \in \mathbb{R}^d, M \in \mathbb{R}^{d \times d}} \quad \alpha \\ \text{s.t.} \quad & M \succeq \alpha\, M^\star \\ & \left\langle M, (\boldsymbol{x} - \boldsymbol{\mu})(\boldsymbol{x} - \boldsymbol{\mu})^\top \right\rangle \leq 1 \qquad \forall \boldsymbol{x} \in S_C \\ & \left\langle M, (\boldsymbol{y} - \boldsymbol{\mu})(\boldsymbol{y} - \boldsymbol{\mu})^\top \right\rangle > 1 \qquad \forall \boldsymbol{y} \in S_{\bar{C}} \end{aligned} \tag{2.48}$$

where, for any two symmetric matrices A and B, $\langle A, B \rangle = \operatorname{tr}(AB)$ is the usual Frobenius inner product, implying $\langle M, (\boldsymbol{x} - \boldsymbol{\mu})(\boldsymbol{x} - \boldsymbol{\mu})^\top \rangle = d_M(\boldsymbol{x}, \boldsymbol{\mu})^2$. In words, the constraint $M \succeq \alpha\, M^\star$ says that E must fit into E_{J} if we scale E_{J} by a factor $\Phi = 1/\sqrt{\alpha}$. The other constraints require E to contain all of S_C but none of the points of $S_{\bar{C}}$. The objective function thus minimizes the stretch Φ of E.

In the rest of this paragraph we prove Theorem 2.9.

Proof of Theorem 2.9 (sketch). To build the intuition, we first give a proof sketch where the involved quantities are simplified. The analysis is performed in the latent space $\mathbb{R}^d$ with inner product $\langle \boldsymbol{u}, \boldsymbol{v} \rangle = \boldsymbol{u}^\top W \boldsymbol{v}$. Setting conventionally $\boldsymbol{z} = \boldsymbol{0}$, C then lies in the unit ball $\mathcal{B}_0$ and all points of $X \setminus C$ lie outside $\sqrt{1+\gamma}\,\mathcal{B}_0$. For simplicity we assume $\gamma \ll 1$ so that $\sqrt{1+\gamma} \simeq 1+\gamma$, but we can easily extend the result to any $\gamma > 0$. Now fix the subset $S_C \subseteq C$, and let $E_{\mathrm{J}} = E_{\mathrm{J}}(S_C)$ be the MVEE of S_C. Observe the following fact: $\mathcal{B}_0$ trivially satisfies (1), but in general violates (2); in contrast, E_{J} trivially satisfies (2), but in general violates (1). The key idea is thus to "compare" $\mathcal{B}_0$ and E_{J} and take, loosely speaking, the best of the two. To see how this works, suppose for instance E_{J} has small radius, say less than $\gamma/4$. In this case, $E = E_{\mathrm{J}}$ yields the thesis. Indeed, since the center $\boldsymbol{\mu}^\star$ of E_{J} is in $\mathcal{B}_0$, then any point of E is within distance $1 + \gamma/4 \leq \sqrt{1+\gamma}$ of the center of $\mathcal{B}_0$, and lies inside $\sqrt{1+\gamma}\,\mathcal{B}_0$. Thus E_{J} separates S_C from $X \setminus C$, satisfying (1). At the other extreme, suppose E_{J} is large, say with all its d semiaxes longer than $\gamma/4$. In this case, $E = \mathcal{B}_0$ yields the thesis: indeed, by

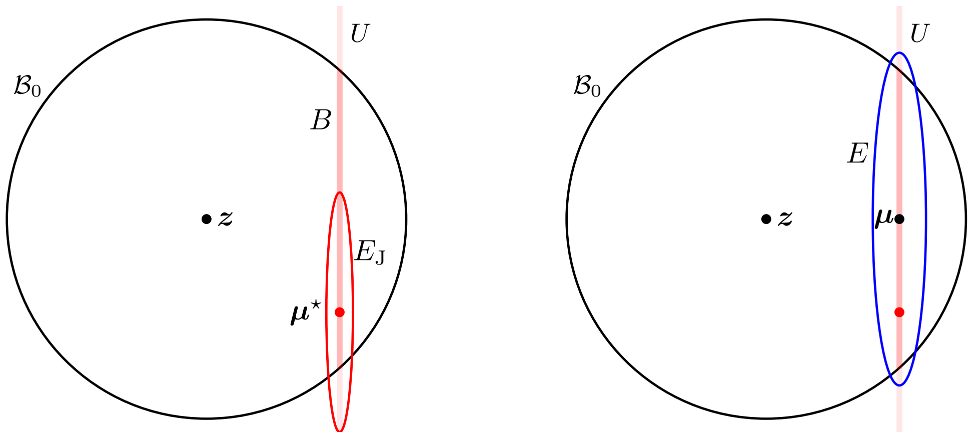


FIGURE 2.B.1: Left: the MVEE E_{J} of S_C and the affine subspace $U + \boldsymbol{\mu}^\star$ (marked simply as U) spanned by its largest semiaxes. There is no guarantee that $E_{\mathrm{J}} \subseteq \sqrt{1+\gamma}\,\mathcal{B}_0$. Right: the separator E, centered in the center $\boldsymbol{\mu}$ of B, with the largest semiaxis in U and the smallest one in $U_\perp$. We can guarantee that $S_C \subset E \subset \sqrt{1+\gamma}\,\mathcal{B}_0$.

hypothesis E fits entirely inside $^4/_\gamma\, E_{\mathrm{J}}$, satisfying (2). Unfortunately, the general case is more complex, since E_{J} may be large along some axes and small along others. In this case, both $\mathcal{B}_0$ and E_{J} fail to satisfy the properties. This requires us to choose the axes and the center of E more carefully. We show how to do this with the help of Figure 2.B.1.

Let $\{\boldsymbol{u}_1, \dots, \boldsymbol{u}_d\}$ be the orthonormal basis defined by the semiaxes of E_{J} and $\ell_1^\star, \dots, \ell_d^\star$ be the corresponding semiaxes lengths. We define a threshold $\epsilon = {}^{\gamma^3}/_{d^2}$, and partition $\{\boldsymbol{u}_1, \dots, \boldsymbol{u}_d\}$ as $A_P = \{i : \ell_i^\star > \epsilon\}$ and $A_Q = \{i : \ell_i^\star \le \epsilon\}$. Thus A_P contains the large semiaxes of E_{J} and A_Q the small ones. Let $U, U_\perp$ be the subspaces spanned by $\{\boldsymbol{u}_i : i \in A_P\}$ and $\{\boldsymbol{u}_i : i \in A_Q\}$, respectively. Consider the subset $B = \mathcal{B}_0 \cap (\boldsymbol{\mu}^\star + U)$. Note that B is a ball in at most d dimensions, since it is the intersection of a d-dimensional ball and an affine linear subspace of $\mathbb{R}^d$. Let $\boldsymbol{\mu}$ and ℓ be, respectively, the center and radius of B. We set the center of E at $\boldsymbol{\mu}$, and the lengths ℓ_i of its semiaxes as follows:

$$\ell_i = \begin{cases} \frac{\ell}{\sqrt{1-\gamma}} & \text{if } i \in A_P \\ \frac{\ell_i^\star}{\sqrt{\epsilon}} & \text{if } i \in A_Q \end{cases} \tag{2.49}$$

Loosely speaking, we are "copying" the semiaxes from either $\mathcal{B}_0$ or E_{J} depending on $\ell_i^\star$. In particular, the large semiaxes (in A_P) are set so to contain all of B and exceed it by a little, taking care of not intersecting $\sqrt{1+\gamma}\,\mathcal{B}_0$. Instead, the small semiaxes (in A_Q) are so small that we can safely set them to $1/\sqrt{\varepsilon}$ times those of E_{J}, so that we add some "slack" to include S_C without risking to intersect $\sqrt{1+\gamma}\,\mathcal{B}_0$. Now we are done, and our low-stretch separator is $(M, \boldsymbol{\mu})$ where $M = \sum_{i=1}^d \ell_i^{-2} \boldsymbol{u}_i \boldsymbol{u}_i^\intercal$. This the ellipsoid E that yields Theorem 2.9. In the next paragraph, we show how we can find efficiently all points in E that belong to C.

2.B.3 Proof of Theorem 2.9 (full).

We prove the theorem for $\gamma \le {}^1/_5$ and use the fact that whenever C has weak margin γ then it also has weak margin γ' for all $\gamma' > \gamma$. As announced, the analysis is carried out in the latent space $\mathbb{R}^d$ equipped with the inner product $\langle \boldsymbol{u}, \boldsymbol{v} \rangle = \boldsymbol{u}^\top W \boldsymbol{v}$. All norms $\|\boldsymbol{u}\|$, distances $d(\boldsymbol{u}, \boldsymbol{v})$, and (cosine of) angles $\langle \boldsymbol{u}, \boldsymbol{v} \rangle \big/ \big(\|\boldsymbol{u}\|\,\|\boldsymbol{v}\|\big)$ are computed according to this inner product unless otherwise specified. Let $\mathcal{B}_0$ be the unit ball centered at the origin, which we conventionally set at $\boldsymbol{z}$, the point in the convex hull of C according to which the margin is computed. Then, by assumption, $C \subset \mathcal{B}_0$, and $\boldsymbol{x} \notin \sqrt{1+\gamma}\,\mathcal{B}_0$ for all $\boldsymbol{x} \notin C$. For ease of notation, in this proof be denote the MVEE by $E^\star$ rather than E_{J}. Let then $(E^\star, \boldsymbol{\mu}^\star)$ be the MVEE of S_C; note that $\boldsymbol{\mu}^\star \in \mathrm{conv}(S_C) \subseteq \mathcal{B}_0$. We let $\boldsymbol{u}_1, \dots, \boldsymbol{u}_d$ be the orthonormal eigenvector basis given by the axes of $E^\star$ and $\lambda_1^\star, \dots, \lambda_d^\star$ the corresponding eigenvalues. Note that if $\min_i \lambda_i^\star \ge {}^5/_{\gamma^2}$ then $E^\star$ has radius $\le {}^\gamma/_{\sqrt{5}}$ and thus, since $\boldsymbol{\mu}^\star \in \mathcal{B}_0$ and $\gamma \le {}^1/_5$, its distance from $\mathcal{B}_0$ is at most

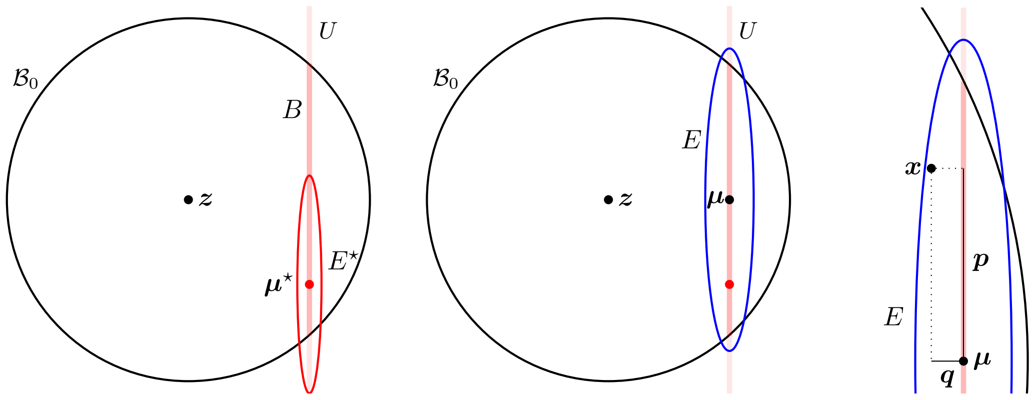


FIGURE 2.B.2: Left: the separating ball $\mathcal{B}_0$ of C, the MVEE $E^\star$ of S_C, and the affine subspace $U + \boldsymbol{\mu}^\star$ spanned by its largest semiaxes. Middle: E is our separator centered in the center $\boldsymbol{\mu}$ of the ball $B = U \cap \mathcal{B}_0$. Right: a point $\boldsymbol{x} \in S_C$ with its projections onto U and $U_\perp$ with respect to the origin, which we conventionally set at $\boldsymbol{\mu}$ (the center of E).

$1+\gamma/\sqrt{5} = \sqrt{1 + 2\gamma/\sqrt{5} + \gamma^2/5} < \sqrt{1+\gamma}$. In this case we can simply set $E = E^\star$ and the thesis is proven. Thus, from now on we assume $\min_i \lambda_i^\star < 5/\gamma^2$.

Now let:

$$\epsilon = \frac{\gamma^3}{32d^2} \tag{2.50}$$

and partition (the indices of) the basis $\{\boldsymbol{u}_1, \dots, \boldsymbol{u}_d\}$ as follows:

$$A_P = \{i : \lambda_i^\star < 1/\epsilon^2\}, \quad A_Q = [d] \setminus A_P \tag{2.51}$$

Since $\min_i \lambda_i^\star < 5/\gamma^2$ and $5/\gamma^2 \le 1/\epsilon^2$, then by construction the set A_P is not empty. We now define the ellipsoid E. Let $U, U_\perp$ be the subspaces spanned by $\{\boldsymbol{u}_i : i \in A_P\}$ and $\{\boldsymbol{u}_i : i \in A_Q\}$ respectively, and let $B = \mathcal{B}_0 \cap (\boldsymbol{\mu}^\star + U)$. Note that B is a ball, since it is the intersection of a ball and an affine linear subspace. Let $\boldsymbol{\mu}$ and ℓ be, respectively, the center and radius of B and define

$$\lambda_i = \begin{cases} (1-\sqrt{5\gamma/4})\ell^{-2} & i \in A_P \\ \epsilon\lambda_i^\star & i \in A_Q \end{cases} \qquad M = \sum_{i=1}^d \lambda_i \boldsymbol{u}_i \boldsymbol{u}_i^\intercal \tag{2.52}$$

Then our ellipsoidal separator is $E = \{\boldsymbol{x} \in \mathbb{R}^d : d_M(\boldsymbol{x}, \boldsymbol{\mu}) \le 1\}$. See Figure 2.B.2 for a pictorial representation. We now prove that E satisfies: **(1)** $S_C \subset E$, **(2)** $E \subseteq \frac{64\sqrt{2}d^2}{\gamma^3} E^\star(S_C)$, **(3)** $E \subset \sqrt{1+\gamma}\,\mathcal{B}_0$.

Proof of (1). Set the center $\boldsymbol{\mu}$ of E as the origin. For all $i \in [d]$ let $U_i = \boldsymbol{u}_i \boldsymbol{u}_i^\intercal$ and define the following matrices:

$$P_0 = \sum_{i \in A_P} U_i, \qquad Q_0 = \sum_{i \in A_Q} U_i \tag{2.53}$$

$$P = \sum_{i \in A_P} \lambda_i U_i, \qquad Q = \sum_{i \in A_Q} \lambda_i U_i \tag{2.54}$$

$$P_\star = \sum_{i \in A_P} \lambda_i^\star U_i, \quad Q_\star = \sum_{i \in A_Q} \lambda_i^\star U_i \tag{2.55}$$

We want to show that $d_M^2(\boldsymbol{x}, \boldsymbol{\mu}) \le 1$ for all $\boldsymbol{x} \in S_C$. Note that $d_M(\boldsymbol{x}, \boldsymbol{\mu})^2$ equals (recall that $\boldsymbol{\mu} = \boldsymbol{0}$):

$$\boldsymbol{x}^\intercal P \boldsymbol{x} + \boldsymbol{x}^\intercal Q \boldsymbol{x} \tag{2.56}$$

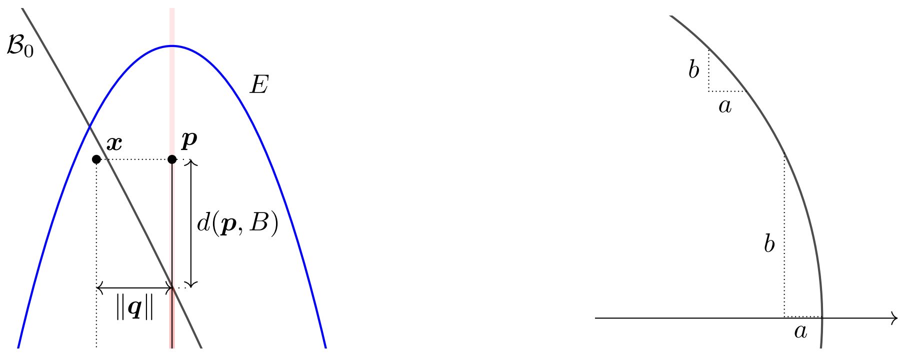


FIGURE 2.B.3: Left: a point $\boldsymbol{x} \in S_C \subset \mathcal{B}_0$ which lies in E as well. Right: for a fixed $a > 0$, the ratio b/a is maximized when the segment of length a lies on the line passing through the center of $\mathcal{B}_0$, in which case $b/a = \frac{\sin\theta}{1-\cos\theta}$ for some $\theta \in (0, \pi/2)$.

Let us start with the second term of (2.56). By definition of $Q_\star$ and since

$$\boldsymbol{\mu}^{\star\intercal} Q_\star = (\boldsymbol{\mu}^\star - \boldsymbol{\mu})^\intercal Q_\star = \mathbf{0}$$

because $\boldsymbol{\mu}^\star - \boldsymbol{\mu} \in U$,

$$\boldsymbol{x}^\intercal Q \boldsymbol{x} = \epsilon\, \boldsymbol{x}^\intercal Q_\star \boldsymbol{x} = \epsilon\, (\boldsymbol{x} - \boldsymbol{\mu}^\star)^\intercal Q_\star (\boldsymbol{x} - \boldsymbol{\mu}^\star) \le \epsilon < \frac{\gamma}{4} \tag{2.57}$$

where the penultimate inequality follows from $\boldsymbol{x} \in E^\star$.

We turn to the first term of (2.56). If we let $\boldsymbol{p}, \boldsymbol{q}$ be the projections of $\boldsymbol{x} - \boldsymbol{\mu} = \boldsymbol{x}$ onto $U, U_\perp$, so that

$$\|\boldsymbol{p}\|^2 = \boldsymbol{x}^\intercal P_0 \boldsymbol{x}, \qquad \|\boldsymbol{q}\|^2 = \boldsymbol{x}^\intercal Q_0 \boldsymbol{x} \tag{2.58}$$

then by definition of the λ_i we have:

$$\boldsymbol{x}^\intercal P \boldsymbol{x} = \frac{1 - \sqrt{5\gamma/4}}{\ell^2} \|\boldsymbol{p}\|^2 \tag{2.59}$$

We can thus focus on bounding $\|\boldsymbol{p}\|$. Since B is a ball of radius ℓ, then $\|\boldsymbol{p}\| \le \ell + d(\boldsymbol{p}, B)$, where $d(\boldsymbol{p}, B)$ is the distance of $\boldsymbol{p}$ from its projection on B —see Figure 2.B.3, left.

Now, since $\boldsymbol{x} \in \mathcal{B}_0$, the ratio $\frac{d(\boldsymbol{p},B)}{\|\boldsymbol{q}\|}$ is maximized when $\ell \to 0$ (i.e., B has a vanishing radius), in which case $d(\boldsymbol{p}, B) \le \sin\theta$ and $\|\boldsymbol{q}\| \ge 1 - \cos\theta$, where $\theta \in (0, \pi/2]$; see Figure 2.B.3 right. Then:

$$\frac{\|\boldsymbol{q}\|}{d(\boldsymbol{p}, B)} \ge \frac{1 - \cos\theta}{\sin\theta} = \tan\frac{\theta}{2} \ge \frac{\theta}{2} \ge \frac{\sin\theta}{2} \ge \frac{d(\boldsymbol{p}, B)}{2} \tag{2.60}$$

where we used the tangent half-angle formula and the Taylor expansion of $\tan\theta$. This yields $d(\boldsymbol{p}, B) \le \sqrt{2\, \|\boldsymbol{q}\|_2}$. Thus:

$$\|\boldsymbol{p}\| \le \ell + \sqrt{2\|\boldsymbol{q}\|} \tag{2.61}$$

But since $\lambda_i^\star \ge 1/\epsilon^2$ for all $i \in A_Q$:

$$\|\boldsymbol{q}\|^2 = \boldsymbol{x}^\intercal Q_0 \boldsymbol{x} \le \epsilon^2\, \boldsymbol{x}^\intercal Q_\star \boldsymbol{x} = \epsilon^2 (\boldsymbol{x} - \boldsymbol{\mu}^\star)^\intercal Q_\star (\boldsymbol{x} - \boldsymbol{\mu}^\star) \le \epsilon^2 \tag{2.62}$$

Therefore:

$$\boldsymbol{x}^\intercal P \boldsymbol{x} \le \frac{1 - \sqrt{5\gamma/4}}{\ell^2} \big(\ell + \sqrt{2\epsilon}\big)^2 \le (1 - \sqrt{5\gamma/4}) \big(1 + \sqrt{2\epsilon}/\ell\big)^2 \tag{2.63}$$

Next, we show that $\frac{\sqrt{2\epsilon}}{\ell} \leq \frac{1}{2}\sqrt{5\gamma/4}$. First,

$$\sqrt{2\epsilon} = \sqrt{2\frac{\gamma^3}{32d^2}} = \frac{\gamma\sqrt{\gamma}}{4d} \tag{2.64}$$

We now temporarily set $\boldsymbol{\mu}^\star$ as the origin. We want to show that the projection of ${}^1/_d\, E^\star$ on U is contained in B. Now, the projection of an ellipsoid on the subspace spanned by a subset of its axes is a subset of the ellipsoid itself, and U is by definition spanned by a subset of the axes of $E^\star$. Therefore the projection P of ${}^1/_d\, E^\star$ on U satisfies $P \subseteq {}^1/_d\, E^\star$. Suppose then by contradiction that $P \not\subseteq B$. Since $B = U \cap \mathcal{B}_0$, this implies that ${}^1/_d\, E^\star \notin \mathcal{B}_0$. But by John's theorem, ${}^1/_d\, E^\star \subseteq \operatorname{conv}(S_C)$, and therefore $\operatorname{conv}(S_C) \notin \mathcal{B}_0$, which is absurd. Therefore $P \subseteq B$.

Let us get back to the proof, with $\boldsymbol{\mu}$ as the origin. On the one hand, the definitions of A_P and U imply that the largest semiaxis of $E^\star$ of length $\ell^\star = 1/\sqrt{\min_i \lambda_i^\star}$ lies in U, thus P has radius at least $\frac{1}{d}\ell^\star$. On the other hand B has radius ℓ, and we have seen that $P \subseteq B$. Therefore, $\ell \geq \frac{1}{d}\ell^\star$. Finally, by our assumption on $\min_i \lambda_i^\star$, we have $\min_i \lambda_i^\star < {}^5/_{\gamma^2}$ and so $\ell^\star > {}^\gamma/_{\sqrt{5}}$. Therefore, $\ell \geq {}^\gamma/_{\sqrt{5}d}$, which together with (2.64) guarantees $\frac{\sqrt{2\epsilon}}{\ell} \leq \frac{\sqrt{5\gamma}}{4} = \frac{1}{2}\sqrt{5\gamma/4}$. Thus, continuing (2.63):

$$\boldsymbol{x}^\intercal P \boldsymbol{x} \leq (1 - \sqrt{5\gamma/4})\left(1 + \frac{1}{2}\sqrt{5\gamma/4}\right)^2 \tag{2.65}$$

Now $(1-x)(1+\frac{x}{2})^2 < 1 - \frac{3}{4}x^2$ for all $x > 0$, thus with $x = \sqrt{5\gamma/4} > \sqrt{\gamma}$ we get:

$$\boldsymbol{x}^\intercal P \boldsymbol{x} < 1 - \frac{3}{4}\gamma \tag{2.66}$$

By summing (2.57) and (2.66), we get:

$$\boldsymbol{x}^\intercal P \boldsymbol{x} + \boldsymbol{x}^\intercal Q \boldsymbol{x} < 1 - \frac{3}{4}\gamma + \frac{\gamma}{4} < 1 \tag{2.67}$$

Proof of (2). Comparing the eigenvalues of E and $E^\star$, and using $\ell \leq 1$ and $\gamma \leq {}^1/_5$, we obtain:

$$\frac{\lambda_i}{\lambda_i^\star} \geq \begin{cases} \frac{(1-\sqrt{5\gamma/4})/\ell^2}{1/\epsilon^2} \geq \frac{\epsilon^2}{2} & i \in A_P \\ \epsilon > \frac{\epsilon^2}{2} & i \in A_Q \end{cases} \tag{2.68}$$

Thus the semiaxes lengths of E are at most ${}^{\sqrt{2}}/_\epsilon$ times those of $E^\star$. Now let $E^\star_+$ be the set obtained by scaling $E^\star$ by a factor ${}^{2\sqrt{2}}/_\epsilon = {}^{64\sqrt{2}d^2}/_{\gamma^3}$ about its origin $\boldsymbol{\mu}^\star$. Note that $\boldsymbol{\mu}^\star \in \operatorname{conv}(S_C)$ and, by item **(1)**, $\operatorname{conv}(S_C) \subset E$, which implies $\boldsymbol{\mu}^\star \in E$. Now, $E^\star_+$ contains any set of the form $\boldsymbol{y} + \frac{1}{2}E^\star_+$ if the latter contains $\boldsymbol{\mu}^\star$; this includes the set $\frac{\sqrt{2}}{\epsilon}E^\star$ centered in $\boldsymbol{\mu}$, which in turn contains E as we already said.

Proof of (3). We prove that $d(\boldsymbol{x}, \mathcal{B}_0)^2 < \gamma$ for all $\boldsymbol{x} \in E$. Since $\mathcal{B}_0$ is the unit ball, this implies $E \subset \sqrt{1+\gamma}\,\mathcal{B}_0$. Consider then any such $\boldsymbol{x}$. Let again $\boldsymbol{p}, \boldsymbol{q}$ be the projections of $\boldsymbol{x}$ on U and $U_\perp$ respectively. Because $B \subseteq \mathcal{B}_0$, $d(\boldsymbol{x}, \mathcal{B}_0)^2 \leq d(\boldsymbol{x}, B)^2 = d(\boldsymbol{p}, B)^2 + \|\boldsymbol{q}\|^2$. See again Figure 2.B.3, left, but with $\boldsymbol{x}$ possibly outside $\mathcal{B}_0$. For the first term, note that

$$d(\boldsymbol{p}, B) \leq \max_{i \in A_P} \sqrt{{}^1/_{\lambda_i}} - \ell \tag{2.69}$$

By definition of λ_i, this yields:

$$d(\boldsymbol{p}, B)^2 \leq \left(\frac{\ell}{\sqrt{1-\sqrt{5\gamma/4}}} - \ell\right)^2 \leq \left(\frac{1}{\sqrt{1-\sqrt{5\gamma/4}}} - 1\right)^2 \qquad \text{(because } \ell \leq 1\text{)}$$

Now we show that the right-hand side is bounded by $\frac{3}{4}\gamma$. Consider $f(x) = \frac{1}{\sqrt{1-x}} - 1$ for $x \in [0, 1/2]$. Now $\frac{\partial^2 f}{\partial x^2} = \frac{3}{4}(1-x)^{-5/2} > 0$, so f is convex. Moreover, $f(1/2) = \sqrt{2}-1 < 0.83 \cdot 1/2$, and clearly $f(0) = 0 \leq 0.83 \cdot 0$. By convexity then, for all $x \in [0, 1/2]$ we have $f(x) \leq 0.83\,x$ which implies $f(x)^2 < 0.75\,x^2$. By substituting $x = \sqrt{5\gamma/4}$, for all $\gamma \leq 1/5$ we obtain:

$$d(\boldsymbol{p}, B)^2 \leq \left(\frac{1}{\sqrt{1 - \sqrt{5\gamma/4}}} - 1 \right)^2 < \frac{3}{4} \cdot \frac{5}{4}\gamma = \frac{15}{16}\gamma \tag{2.70}$$

Let us now turn to $\boldsymbol{q}$. By definition of Q_0, of Q, and of λ_i for $i \in A_Q$, we have:

$$\|\boldsymbol{q}\|^2 = \boldsymbol{x}^\intercal Q_0 \boldsymbol{x} \leq \max_{i \in A_Q} \frac{1}{\lambda_i} \boldsymbol{x}^\intercal Q \boldsymbol{x} = \max_{i \in A_Q} \frac{1}{\epsilon \lambda_i^\star} \boldsymbol{x}^\intercal Q \boldsymbol{x} \tag{2.71}$$

But $\boldsymbol{x}^\intercal Q \boldsymbol{x} \leq 1$ since $\boldsymbol{x} \in E$, and recalling that $\lambda_i^\star \geq 1/\epsilon^2$ for all $i \in A_Q$, we obtain:

$$\|\boldsymbol{q}\|^2 \leq \frac{1}{\epsilon(1/\epsilon^2)} = \epsilon < \frac{\gamma}{16} \tag{2.72}$$

Finally, by summing (2.70) and (2.72):

$$d(\boldsymbol{x}, \mathcal{B}_0)^2 \leq d(\boldsymbol{p}, B)^2 + \|\boldsymbol{q}\|^2 < \gamma \tag{2.73}$$

The proof is complete.

2.C Supplementary material for Section 2.6

2.C.1 Lemma 2.22

Lemma 2.22. *Let $b > 0$ be a sufficiently large constant. Let S be a sample of points drawn independently and uniformly at random from X. Let $C = \arg\max_{C_j \in \mathcal{C}} |S \cap C_j|$, let $S_C = S \cap C$, and suppose $|S_C| \geq bd^2 \ln k$. If E is any (possibly degenerate) ellipsoid in $\mathbb{R}^d$ such that $S_C = C \cap E$, then with probability at least $1/2$ we have $|C \cap E| \geq |X|\frac{1}{4k}$. The same holds if we require that $E \cap (S \setminus S_C) = \emptyset$, i.e., that E separates S_C from $S \setminus S_C$.*

Proof. Let $n = |X|$ for short, and for any ellipsoid E let $E_X = E \cap X$. We show that, with C defined as above, **(i)** with probability at least $1 - 1/4$ we have $|C| \geq n/2k$, and **(ii)** with probability at least $1 - 1/4$, if $|C| \geq n/2k$ then $|E_X \Delta C| \leq 1/2|C|$ where Δ denotes symmetric difference. By a union bound, then, with probability at least $1/2$ we have $|E \cap C| \geq |C| - |E_X \Delta C| \geq \frac{1}{2}|C| \geq n/4k$.

(i). Let S be the multiset of samples drawn from X, and for every cluster $C_i \in \mathcal{C}$ let N_i be the number of samples in C_i. Let $s = kbd^2 \ln k$; note that $|S| \leq s$ since there are at most k clusters. Now fix any C_i with $|C_i| < \frac{n}{2k}$. Then $\mathbb{E}[N_i] \leq s\frac{|C_i|}{n} < \frac{bd^2 \ln k}{2}$, and by standard concentration bounds (Lemma 2.16 in this supplementary material), we have $\mathbb{P}(N_i \geq bd^2 \ln k) = \exp(-\Omega(b \ln k))$, which for b large enough drops below $1/4k$. Therefore, the probability that $N_i \geq bd^2 \ln k$ when taking $s \leq kbd^2 \ln k$ samples is at most $1/4k$. By a union bound on all C_i with $|C_i| < n/2k$, then, $|C| \geq n/2k$ with probability $1 - 1/4$.

(ii). Consider now any C_i with $|C_i| \geq n/2k$. We invoke the generalization bounds of Theorem 2.15 in this supplementary material with $\epsilon = 1/4k$ and $\delta = 1/4k$, on the hypothesis class $\mathcal{H}$ of all (possibly degenerate) ellipsoids in $\mathbb{R}^d$. For b large enough, the generalization error of any ellipsoid E that contains S_C is, with probability at least $1 - 1/4k$, at most $1/4k$, which means $|E_X \Delta C_i| \leq n/4k \leq 1/2|C_i|$, as desired. By a union bound on all clusters, with probability at least $1 - 1/4$ this holds for all C_i with $|C_i| \geq n/2k$. The same argument holds if we require E to separate $S \cap C_i$ from $S \setminus C_i$, see again Theorem 2.15. By a union bound with point (i) above, we have $E \cap C \leq 1/2|C|$ with probability at least $1/2$, as claimed. □

2.C.2 Proof of Lemma 2.12

Let $X_0 = X$ and $N_0 = n$, and for all $i \geq 1$, let X_i be the set of points not yet labeled at the end of round i, let $N_i = |X_i|$, and let $R_i = \{N_i \leq N_{i-1}(1 - {}^1\!/\!{}_{4k})\}$. Recall that S_C is large enough so that, by Lemma 2.22 in this supplementary material, we have $\mathbb{P}(R_i = 1 \,|\, X_{i-1}) \geq {}^1\!/\!{}_2$ for all i. For every $t \geq 1$ let $\rho_t = \sum_{i=1}^t R_i$. Note that:

$$N_t \leq N_0(1 - {}^1\!/\!{}_{4k})^{\rho_t} < ne^{-\frac{\rho_t}{4k}} \tag{2.74}$$

If $\rho_t \geq 4k\ln(1/\epsilon)$, then $N_t < \epsilon n$ and $\textsc{recur}(X, k, \gamma, \epsilon)$ stops. The number of rounds executed by $\textsc{recur}(X, k, \gamma, \epsilon)$ is thus at most $r_\epsilon = \min\{t : \rho_t \geq 4k\ln(1/\epsilon)\}$.

Now, for all $i \geq 1$ consider the σ-algebra $\mathcal{F}_{i-1}$ generated by $X_0, \ldots, X_{i-1}$, and define: $Z_i = R_i B_i$, where $B_1, B_2, \ldots$ are Bernoulli random variables where each B_i has parameter $1/\big(2\,\mathbb{E}[R_i \mid \mathcal{F}_{i-1}]\big)$. Obviously, $Z_i \leq R_i$, and thus for all t we deterministically have:

$$\rho_t = \sum_{i=1}^t R_i \geq \sum_{i=1}^t Z_i \tag{2.75}$$

Now note that:

$$\mathbb{E}[Z_i \mid \mathcal{F}_{i-1}] = \mathbb{E}[R_i \mid \mathcal{F}_{i-1}] \frac{1}{2\,\mathbb{E}[R_i \mid \mathcal{F}_{i-1}]} = \frac{1}{2} \tag{2.76}$$

Now we can prove the theorem. For the first claim, simply note that $\mathbb{E}[r_\epsilon] \leq 8k\ln(1/\epsilon)$, as this is the expected number of fair coin tosses to get $4k\ln(1/\epsilon)$ heads.

For the second claim, consider any $t \geq 8k\ln n + 6a\sqrt{k}\ln n$. Letting $\zeta_t = \sum_{i=1}^t Z_t$, the event $r_0 \geq t$ implies $\zeta_t < 4k\ln n = \frac{t}{2} - 3a\sqrt{k}\ln n = \mathbb{E}[\zeta_t] - \delta$ where $\delta = 3a\sqrt{k}\ln n$. By Hoeffding's inequality this event has probability at most $e^{-2\delta^2/t}$, and one can check that for all $a \geq 1$ we have $\frac{2\delta^2}{t} \geq a\ln n$.

2.D Supplementary material for Section 2.7

2.D.1 Proof of Theorem 2.13

We state and prove two distinct theorems which immediately imply Theorem 2.13.

Theorem 2.23. *For all $0 < \gamma < {}^1\!/\!{}_7$, all $d \geq 2$, and every (possibly randomized) learning algorithm, there exists an instance on $n \geq 2(\frac{1+\gamma}{8\gamma})^{\frac{d-1}{2}}$ points and $|\mathcal{C}| = 3$ latent clusters such that (1) all clusters have margin γ, and (2) to return with probability ${}^2\!/\!{}_3$ a clustering $\hat{\mathcal{C}}$ such that $\triangle(\hat{\mathcal{C}}, \mathcal{C}) = 0$ the algorithm must make $\Omega(n)$ same-cluster queries in expectation.*

Proof. The idea is the following. We define a single set of points $X \subset \mathbb{R}^d$ and randomize over the choice of the latent PSD matrix W; the claim of the theorem follows by applying Yao's minimax principle. Specifically, we let X be a $\Theta(\sqrt{\gamma})$-packing of points on the unit sphere in $\mathbb{R}^d$. We show that, for $\boldsymbol{x} \in X$ drawn uniformly at random, setting $W = (1+\gamma)\operatorname{diag}(x_1^2, \ldots, x_d^2)$ makes $\boldsymbol{x}$ an outlier, as its distance $d_W(\boldsymbol{x}, \boldsymbol{0})$ from the origin is $1 + \gamma$, while every other point is at distance ≤ 1. Since there are roughly $({}^1\!/\!{}_\gamma)^d$ such points $\boldsymbol{x}$ in our set, the bound follows.

We start by defining the points X in terms of their entry-wise squared vectors. Consider $S_d^+ = \mathbb{R}_+^d \cap S_d$ where $S_d = \{\boldsymbol{x} \in \mathbb{R}^d : \|\boldsymbol{x}\|_2 = 1\}$ is the unit sphere in $\mathbb{R}^d$. We want to show that there exists a set of $\frac{1}{2}({}^1\!/\!{}_\epsilon)^{d-1}$ points in S_d^+ whose pairwise distance is bigger than ${}^\epsilon\!/\!{}_2$, where ϵ will be defined later. To see this, recall that the packing number of the unit ball $B_d = \{\boldsymbol{x} \in \mathbb{R}^d : \|\boldsymbol{x}\|_2 \leq 1\}$ is $\mathcal{M}(B, \epsilon) \geq ({}^1\!/\!{}_\epsilon)^d$ —see, e.g., Vershynin [2018]. For ${}^\epsilon\!/\!{}_2$ and $d-1$, this implies there exists $Y \subseteq B_{d-1}$ such that $|Y| \geq ({}^2\!/\!{}_\epsilon)^{d-1}$ and $\|\boldsymbol{y} - \boldsymbol{y}'\|_2 > {}^\epsilon\!/\!{}_2$ for all distinct $\boldsymbol{y}, \boldsymbol{y}' \in Y$. Now, consider the lifting function $f : B_{d-1} \to \mathbb{R}^d$ defined by $f(\boldsymbol{y}) = (\sqrt{1 - \|\boldsymbol{y}\|_2^2}, y_1, \ldots, y_{d-1})$. Define the lifted set $Z = \{f(\boldsymbol{y}) : \boldsymbol{y} \in Y\}$. Clearly, every $\boldsymbol{z} \in Z$ satisfies $\|\boldsymbol{z}\|_2 = 1$ and $z_0 \geq 0$, so $\boldsymbol{z}$ lies on the northern hemisphere of the sphere S_d. Moreover, $\|f(\boldsymbol{y}) - f(\boldsymbol{y}')\|_2 \geq \|\boldsymbol{y} - \boldsymbol{y}'\|_2$ for any two $\boldsymbol{y}, \boldsymbol{y}' \in Y$. Hence, we have a set Z of $({}^2\!/\!{}_\epsilon)^{d-1}$ points on the d-dimensional sphere such that $\|\boldsymbol{z} - \boldsymbol{z}'\|_2 > {}^\epsilon\!/\!{}_2$ for all distinct $\boldsymbol{z}, \boldsymbol{z}' \in Z$. But a hemisphere is the union of 2^{d-1} orthants, hence some orthant contains at least $2^{-(d-1)}(2/\epsilon)^{d-1} = (1/\epsilon)^{d-1}$ of the points of Z. Without loss of generality we may assume this is the positive orthant and denote the set as Z^+.

We now define the input set $X \subseteq \mathbb{R}^d$ as follows:

$$X = X^+ \cup X^- = \{\sqrt{\boldsymbol{z}} \,:\, \boldsymbol{z} \in Z^+\} \cup \{-\sqrt{\boldsymbol{z}} \,:\, \boldsymbol{z} \in Z^+\}$$

Note that $n = |X| = 2|Z^+| = 2(1/\epsilon)^{d-1}$. Next, we show how every $\boldsymbol{z} \in Z^+$ defines a clustering instance satisfying the constraints of the thesis. For any $\boldsymbol{z}^* \in Z^+$; let $\boldsymbol{w} = (1+\gamma)\boldsymbol{z}^*$ and $W = \operatorname{diag}(w_1, \dots, w_d)$, which is PSD as required. Define the following three clusters:

$$C' = \{-\sqrt{\boldsymbol{z}^*}\} \qquad C'' = \{\sqrt{\boldsymbol{z}^*}\} \qquad C = X \setminus (C' \cup C'')$$

where, for $f : \mathbb{R} \to \mathbb{R}$, $f(\boldsymbol{x}) = \big(f(x_1), \dots, f(x_d)\big)$. Since C' and C'' are singletons, they trivially have weak margin γ. We now show that C has weak margin γ w.r.t. to $\boldsymbol{\mu} = \boldsymbol{0}$; that is, $d_W(\boldsymbol{x}, \boldsymbol{\mu})^2 > 1 + \gamma$ for $\boldsymbol{x} = \pm\sqrt{\boldsymbol{z}^*}$ and $d_W(\boldsymbol{x}, \boldsymbol{\mu})^2 \leq 1$ otherwise. First, note that $d_W(\boldsymbol{x}, \boldsymbol{\mu})^2 = \langle \boldsymbol{w}, \boldsymbol{x}^2 \rangle$. Now,

$$d_W(\boldsymbol{x}, \boldsymbol{\mu})^2 = \begin{cases} (1+\gamma)\langle \boldsymbol{z}^*, \boldsymbol{z}^* \rangle = 1 + \gamma & \text{if } \boldsymbol{x} \in C', C'' \\ (1+\gamma)\langle \boldsymbol{z}^*, \boldsymbol{x}^2 \rangle & \text{if } \boldsymbol{x} \in C \end{cases} \tag{2.77}$$

However, by construction of Z^+, we have that for all $\boldsymbol{x} \in C$ and $\boldsymbol{z} = \boldsymbol{x}^2$,

$$(\epsilon/2)^2 \leq \|\boldsymbol{z} - \boldsymbol{z}^*\|_2^2 = \|\boldsymbol{z}\|_2^2 - 2\langle \boldsymbol{z}, \boldsymbol{z}^* \rangle + \|\boldsymbol{z}^*\|_2^2 = 2(1 - \langle \boldsymbol{z}, \boldsymbol{z}^* \rangle)$$

which implies $\langle \boldsymbol{z}^*, \boldsymbol{x}^2 \rangle \leq 1 - (\epsilon/2)^2/2 = 1 - \epsilon^2/8 = 1/(1+\gamma)$ for $\epsilon = \sqrt{8\gamma/(1+\gamma)}$. Therefore (2.77) gives $d_W(\boldsymbol{x}, \boldsymbol{\mu})^2 = (1+\gamma)\langle \boldsymbol{z}^*, \boldsymbol{x}^2 \rangle \leq 1$. This proves C has weak margin γ as desired.

The size of X is:

$$n \geq 2\Big(\frac{1}{\sqrt{8\gamma/(1+\gamma)}}\Big)^{d-1} = 2\Big(\frac{1+\gamma}{8\gamma}\Big)^{\frac{d-1}{2}}$$

Now the distribution of the instances is defined by taking $\boldsymbol{z}^*$ from the uniform distribution over Z^+. Consider any deterministic algorithm running over such a distribution. Note that same-cluster queries always return $+1$ unless at least one of the two queried points is not in C. As C contains all points in X but the symmetric pair $\sqrt{\boldsymbol{z}^*}, -\sqrt{\boldsymbol{z}^*}$ for a randomly drawn $\boldsymbol{z}^*$, a constant fraction of the points in X must be queried before one element of the pair is found with probability bounded away from zero. Thus, any deterministic algorithm that returns a zero-error clustering with probability at least δ for any constant $\delta > 0$ must perform $\Omega(n)$ queries. By Yao's principle for Monte Carlo algorithms then (see Section 2.A.4 above), any randomized algorithm that errs with probability at most $\frac{1-\delta}{2} \leq \frac{1}{2}$ for any constant $\delta > 0$ must make $\Omega(n)$ queries as well. □

Theorem 2.24. *For all $\gamma > 0$, all $d \geq 48(1+\gamma)^2$, and every (possibly randomized) learning algorithm, there exists an instance on $n = \Omega\big(\exp(d/(1+\gamma)^2)\big)$ points and $|\mathcal{C}| = 2$ latent clusters such that (1) all clusters have margin at least γ, and (2) to return with probability $2/3$ a clustering $\hat{\mathcal{C}}$ such that $\triangle(\hat{\mathcal{C}}, \mathcal{C}) = 0$ the algorithm must make $\Omega(n)$ same-cluster queries in expectation.*

Proof. We exhibit a distribution of instances that gives a lower bound for every algorithm, and then use Yao's minimax principle. Let $p = \frac{1}{2(1+\gamma)}$. Consider a set of vectors $\boldsymbol{x}_1, \dots, \boldsymbol{x}_n$ where every entry of each vector $x_{j,i}$ is i.i.d. and it is equal to 1 with probability. p. Define $X = \{\boldsymbol{x}_1, \dots, \boldsymbol{x}_n\}$; note that in general $|X| \leq n$ since the points might not be all distinct. Let $\boldsymbol{x}^\star = \boldsymbol{x}_n$, $C = \{\boldsymbol{x}_1, \dots, \boldsymbol{x}_{n-1}\}$, $C' = \{\boldsymbol{x}^\star\}$. The latent clustering is $\mathcal{C} = \{C, C'\}$, and the matrix and center of C are respectively $W = \operatorname{diag}(\boldsymbol{x}^\star)$ and $\boldsymbol{c} = \boldsymbol{0}$. The algorithms receive in input a random permutation of X; clearly, if it makes $o(|X|)$ queries, then it has vanishing probability to find $\boldsymbol{x}^\star$, which is necessary to return the latent clustering $\mathcal{C}$.

Now we claim that, if $d \geq 48(1+\gamma)^2$, then we can set $n = \Omega\Big(\exp\big(\frac{d}{48(1+\gamma)^2}\big)\Big)$ and with constant probability we will have **(i)** $|X| = \Omega(n)$, and **(ii)** C, C' have margin γ. This is sufficient, since the theorem then follows by applying Yao's minimax principle.

Let us first bound the probability that $|X| < n$. Note that for any two points $\boldsymbol{x}_i, \boldsymbol{x}_{i'}$ with $i \neq i'$ we have $\mathbb{P}(\boldsymbol{x}_i = \boldsymbol{x}_{i'}) = ((1-p)^2 + p^2)^d < (1 - \frac{1}{2(1+\gamma)})^d < e^{-\frac{d}{2(1+\gamma)}}$. Therefore, by a simple union bound over all pairs, $\mathbb{P}(|X| < n) < n^2 e^{-\frac{d}{2(1+\gamma)}}$.

Next we want show that, $d_W(\boldsymbol{x}, \boldsymbol{c})^2 \simeq dp$ for $\boldsymbol{x} \in C'$ whereas $d_W(\boldsymbol{x}, \boldsymbol{c})^2 \simeq dp^2$ for $\boldsymbol{x} \in C$; this will give the margin.

Now, for any $\boldsymbol{x}$,

$$d_W(\boldsymbol{x}, \boldsymbol{c})^2 = \sum_{i=1}^{d} x_i^\star \, (x_i - 0)^2 = \begin{cases} \sum_{i=1}^{d} x_i^\star \, x_i \sim B(d, p^2) & \boldsymbol{x} \in C \\ \sum_{i=1}^{d} x^\star \sim B(d, p) & \boldsymbol{x} \in C' \end{cases} \tag{2.78}$$

Where in the last equality we use the fact that the entries are unary, and where with the notation $B(d, p)$ we refer to a vector of length d where each entry is equal to 1 with probability p. Let $\mu = dp^2$ and $\mu' = dp$, let $\epsilon = 1/(1+\sqrt{2})$, and define

$$\phi = \mu(1+\epsilon), \qquad \phi' = \mu'(1 - \epsilon\sqrt{p}) \tag{2.79}$$

By standard tail bounds,

$$\mathbb{P}(d_W(\boldsymbol{x}, \boldsymbol{c})^2 \geq \phi) \leq e^{-\frac{\epsilon^2 \mu}{3}} \quad \text{for } \boldsymbol{x} \in C \tag{2.80}$$

$$\mathbb{P}(d_W(\boldsymbol{x}, \boldsymbol{c})^2 < \phi') < e^{-\frac{\epsilon^2 p \mu'}{3}} = e^{-\frac{\epsilon^2 \mu}{3}} \quad \text{for } \boldsymbol{x} \in C' \tag{2.81}$$

By a union bound on all points, the margin γ_C of C fails to satisfy the following inequality with probability at most $|X| e^{-\frac{\epsilon^2 \mu}{3}} \leq n e^{-\frac{\epsilon^2 \mu}{3}}$:

$$1 + \gamma_C = \frac{\min_{\boldsymbol{x} \notin C} d_W(\boldsymbol{x}, \boldsymbol{c})^2}{\max_{\boldsymbol{x} \in C} d_W(\boldsymbol{x}, \boldsymbol{c})^2} \geq \frac{\phi'}{\phi} = \frac{dp(1-\epsilon\sqrt{p})}{dp^2(1+\epsilon)} = \frac{1-\epsilon\sqrt{p}}{p(1+\epsilon)} \geq \frac{1}{2p} = 1 + \gamma \tag{2.82}$$

where the penultime inequality holds since $\frac{1-\epsilon\sqrt{p}}{1+\epsilon} \geq \frac{1}{2}$ for our values of p and ϵ. Note that, since $p = \frac{1}{2(1+\gamma)}$ and $n \leq \frac{1}{c} \exp\left(\frac{d}{48(1+\gamma)^2}\right) + 1$,

$$n e^{-\frac{\epsilon^2 \mu}{3}} = n e^{-\frac{dp^2}{12}} = n e^{-\frac{d}{48(1+\gamma)^2}} \tag{2.83}$$

By one last union bound, the probability that $|X| = n$ and $\gamma_C \geq \gamma$ is at least

$$1 - n e^{-\frac{d}{48(1+\gamma)^2}} - n^2 e^{-\frac{d}{2(1+\gamma)}} \tag{2.84}$$

If $d \geq \frac{48}{(1+\gamma)^2}$, then we can let $n = \Omega\left(e^{\frac{d}{48(1+\gamma)^2}}\right)$ while ensuring the above probability is bounded away from 0.

The rest of the proof and the application of Yao's principle is essentially identical to the proof of Theorem 2.23 above. □

2.E Comparison with SCQ-k-means

In this section we compare our algorithm to SCQ-k-means of Ashtiani et al. [2016]. We show that, in our setting, SCQ-k-means fails even on very simple instances, although it can still work under (restrictive) assumptions on γ, W, and the centers.

SCQ-k-means works as follows. First, the center of mass $\boldsymbol{\mu}_C$ of some cluster C is estimated using $\mathcal{O}\big(\text{poly}(k, 1/\gamma)\big)$ SCQ queries; second, all points in X are sorted by their distance from $\boldsymbol{\mu}_C$ and the radius of C is found via binary search. The binary search is done using same-cluster queries between the sorted points and any point already known to be in C. The margin condition ensures that, if we have an accurate enough estimate of $\boldsymbol{\mu}_C$, then the binary search will be successful (there are no inversions of the sorted points w.r.t. their cluster). This approach thus yields a $\mathcal{O}(\ln n)$ SCQ queries bound (the number of queries to estimate $\boldsymbol{\mu}_C$ is independent of n).

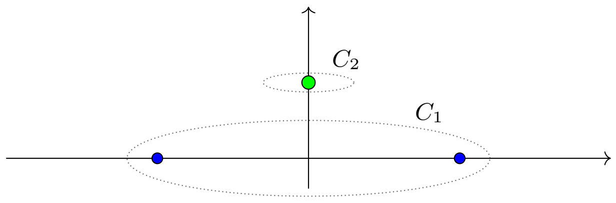


FIGURE 2.E.1: A bad instance for SCQ-k-means. With good probability, the algorithm classifies all points in a single cluster, incurring error $\simeq 1/2$, the same as a random labeling.

It is easy to see that this algorithm relies crucially on (1) each cluster C must be spherical, and (2) the center of the sphere must coincide with the centroid $\boldsymbol{\mu}_C$. In formal terms, the setting of Ashtiani et al. [2016] is a special cases of ours where for all C we have $W_C = I_d$ and $\boldsymbol{c} = \mathbb{E}_{\boldsymbol{x}\in C}[\boldsymbol{x}]$. If any of these two assumptions does not hold, then it is easy to construct instances where Ashtiani et al. [2016] fails to recover the clusters and, in fact, achieves error very close to a completely random labeling. Formally:

Lemma 2.25. *For any fixed $d \geq 2$, any $p \in (0,1)$, and any sufficiently small $\gamma > 0$, there are arbitrarily large instances on n points and $k = 2$ clusters on which* SCQ-*k-means incurs error $\triangle(\hat{\mathcal{C}}, \mathcal{C}) \geq \frac{1-p}{2}$ with probability at least $1-p$.*

Sketch of the proof. We describe the generic instance on n points for $d = 2$. The latent clustering $\mathcal{C}$ is formed by two clusters C_1, C_2 of size respectively $n_1 = n^{\frac{1+p}{2}}$ and $n_2 = n^{\frac{1-p}{2}}$. In C_1, half of the points are in $(1,0)$ and half in $(-1,0)$. In C_2, all points are in $(0, \frac{\sqrt{1+\gamma}}{2})$. (One can in fact perturb the instance so that all points are distinct without impairing the proof). For both clusters, the center coincide with their center of mass, $\boldsymbol{\mu}_1 = (0,0)$ and $\boldsymbol{\mu}_2 = (0, \frac{\sqrt{1+\gamma}}{2})$. For both clusters, the latent metric is given by the PSD matrix $W = \left(\begin{smallmatrix} .25 & 0 \\ 0 & 1 \end{smallmatrix}\right)$.

It is easy to see that $d_W(\boldsymbol{x}, \boldsymbol{\mu}_1)^2 = 1/4$ if $\boldsymbol{x} \in C_1$ and $d_W(\boldsymbol{x}, \boldsymbol{\mu}_1)^2 = (1+\gamma)/4$ if $\boldsymbol{x} \in C_2$, and so C_1 has margin exactly γ. On the other hand C_2 has margin γ since $d_W(\boldsymbol{x}, \boldsymbol{\mu}_2)^2 = 0$ if $\boldsymbol{x} \in C_2$ and $d_W(\boldsymbol{x}, \boldsymbol{\mu}_2)^2 > 0$ otherwise.

Now consider SCQ-k-means. The algorithm starts by sampling at least $\frac{k\ln(k)}{\gamma^4}$ points from X and setting $\hat{\boldsymbol{\mu}}$ to the average of the points with the majority label. By standard concentration bounds then, for γ small enough, with probability at least $1-p$ the majority cluster will be C_1 and the estimate $\hat{\boldsymbol{\mu}}$ of its center of mass $(0,0)$ will be sufficiently close to $\boldsymbol{\mu}_1$ that the ordering of all points in X by their Euclidean distance w.r.t. $\hat{\boldsymbol{\mu}}$ will set all of C_2 before all of C_1. But since $n_2 = n^{\frac{1-p}{2}}$, the median of the sorted sequence will be a point of C_1. Thus the binary search will make its first query on a point of C_1 and will continue thereafter classifying all of X as belonging to C_1. Thus the algorithm will output the clustering $\hat{\mathcal{C}} = \{X, \emptyset\}$ which gives $\triangle(\hat{\mathcal{C}}, \mathcal{C}) = \frac{1-p}{2}$. □

Next, we show that the approach Ashtiani et al. [2016] still works if one relaxes the assumption $W = I$, at the price of strengthening the margin γ. Let $\lambda_{\max}$ and $\lambda_{\min} > 0$ be, respectively, the largest and smallest eigenvalues of W. The condition number κ_W of W is the ratio $\lambda_{\max}/\lambda_{\min}$. If κ_W is not too large, then W does not significantly alter the Euclidean metric, and the ordering of the points is preserved. Formally:

Lemma 2.26. *Let κ_W be the condition number of W. If every cluster C has margin at least $\kappa_W - 1$ with respect to its center of mass $\boldsymbol{\mu}_C$, and if we know $\boldsymbol{\mu}_C$, then we can recover C with $\mathcal{O}(\ln n)$* SCQ *queries.*

Proof. Fix any cluster C and let $\boldsymbol{\mu} = \boldsymbol{\mu}_C$. For any $\boldsymbol{z} \in \mathbb{R}^d$ we have $\lambda_{\min}\|\boldsymbol{z}\|_2^2 \leq \|\boldsymbol{z}\|_W^2 \leq \lambda_{\max}\|\boldsymbol{z}\|_2^2$ where $\lambda_{\min}$ and $\lambda_{\max}$ are, respectively, the smallest and largest eigenvalue of W. Sort all other points $\boldsymbol{x}$ by their Euclidean distance $\|\boldsymbol{x} - \boldsymbol{\mu}\|_2$ from $\boldsymbol{\mu}$. Then, for any $\boldsymbol{x} \in C$ and any $\boldsymbol{y} \notin C$ we have:

$$\frac{\|\boldsymbol{y} - \boldsymbol{\mu}\|_2^2}{\|\boldsymbol{x} - \boldsymbol{\mu}\|_2^2} \geq \frac{\lambda_{\min}}{\lambda_{\max}} \frac{\|\boldsymbol{y} - \boldsymbol{\mu}\|_W^2}{\|\boldsymbol{x} - \boldsymbol{\mu}\|_W^2} = \frac{1}{\kappa_W} \frac{d(\boldsymbol{y}, \boldsymbol{\mu})^2}{d(\boldsymbol{x}, \boldsymbol{\mu})^2} > \frac{1+\gamma}{\kappa_W} \tag{2.85}$$

Hence, if $\gamma \geq \kappa_W - 1$, there is $r \geq 0$ such that $\|\boldsymbol{x} - \boldsymbol{\mu}\|_2 \leq r$ for all $\boldsymbol{x} \in C$ and $\|\boldsymbol{y} - \boldsymbol{\mu}\|_2 \geq r$ all $\boldsymbol{y} \notin C$. We can thus recover C via binary search as in Ashtiani et al. [2016]. □

As a final remark, we observe that the above approach is rather brittle, since κ_W is unknown (because W is), and if the condition $\kappa_W \leq 1 + \gamma$ fails, then once again the binary search can return a clustering far from the correct one.

2.F Comparison with metric learning

In this section we show that metric learning, a common approach to latent cluster recovery and related problems, does not solve our problem even when combined with same-cluster and comparison queries. Intuitively, we want to learn an approximate distance $\hat{d}$ that preserves the ordering of the distances between the points. That is, for all $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z} \in X$, $d(\boldsymbol{x}, \boldsymbol{y}) \leq d(\boldsymbol{x}, \boldsymbol{z})$ implies $\hat{d}(\boldsymbol{x}, \boldsymbol{y}) \leq \hat{d}(\boldsymbol{x}, \boldsymbol{z})$. If this holds then d and $\hat{d}$ are equivalent from the point of view of binary search. To simplify the task, we may equip the algorithm with an additional *comparison query* CMP, which takes in input two pairs of points $\boldsymbol{x}, \boldsymbol{x}'$ and $\boldsymbol{y}, \boldsymbol{y}'$ from X and tells precisely whether $d(\boldsymbol{x}, \boldsymbol{x}') \leq d(\boldsymbol{y}, \boldsymbol{y}')$ or not. It turns out that, even with SCQ+CMP queries, learning such a $\hat{d}$ requires to query essentially all the input points.

Theorem 2.27. *For any $d \geq 3$, learning any $\hat{d}$ such that, for all $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z} \in X$, if $d(\boldsymbol{x}, \boldsymbol{y}) \leq d(\boldsymbol{x}, \boldsymbol{z})$ then $\hat{d}(\boldsymbol{x}, \boldsymbol{y}) \leq \hat{d}(\boldsymbol{x}, \boldsymbol{z})$, requires $\Omega(n)$ SCQ+CMP queries in the worst case, even with an arbitrarily large margin γ.*

Proof. We reduce the problem of learning the order of pairwise distances induced by W, which we call ORD, to the problem of learning a separator hyperplane, which we call SEP and whose query complexity is linear in n.

Problem SEP is as follows. The inputs are a set $X = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\} \subset \mathbb{R}^d$ (the observations) and a set $\mathcal{H} = \{\boldsymbol{h}_1, \ldots, \boldsymbol{h}_k\} \subset \mathbb{R}^d_+$ (the hypotheses). We require that $\boldsymbol{h}_j \in \mathbb{R}^d_+$. We have oracle access to $\sigma : X \to \{+1, -1\}$ such that $\sigma(\cdot) = \operatorname{sign}\langle \boldsymbol{h}, \cdot \rangle$ for some $\boldsymbol{h} \in \mathcal{H}$. The output is the $\boldsymbol{h} \in \mathcal{H}$ that agrees with σ. We assume $\mathcal{H}, X$ support a margin: $\exists \epsilon > 0$, possibly dependent on the instance, such that $\operatorname{sign}\langle \boldsymbol{h}, \boldsymbol{x} \rangle = \operatorname{sign}\langle \boldsymbol{h}, \boldsymbol{x}' \rangle$ for all $\boldsymbol{x}'$ with $\|\boldsymbol{x} - \boldsymbol{x}'\| \leq \epsilon$. (Note that this is *not* the cluster margin γ).

Let $Q_{\mathrm{ORD}}(n)$ and $Q_{\mathrm{SEP}}(n)$ be the query complexities of ORD and SEP on n points. We show:

Lemma 2.28. *$Q_{ORD}(3n) \leq Q_{SEP}(n)$.*

Proof. Let $X = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\} \subseteq \mathbb{R}^d$ be the input points for SEP and let $\boldsymbol{h} \in \mathbb{R}^d_+$ be the target hypothesis. By scaling the dataset we can assume $\|\boldsymbol{x}_i\| \leq \epsilon$ for any desired ϵ (even dependent on n). We define an instance of ORD on $n' = 3n$ points as follows. First, $W = \operatorname{diag}(\boldsymbol{h})$. Second, the input set is $X' = S_1 \cup \ldots \cup S_n$ where for $i = 1, \ldots, n$ we define $S_i = \{\boldsymbol{a}_i, \boldsymbol{b}_i, \boldsymbol{c}_i\}$ with:

$$\boldsymbol{a}_i = 6^i \cdot \boldsymbol{1} \tag{2.86}$$
$$\boldsymbol{b}_i = 2 \cdot \boldsymbol{a}_i \tag{2.87}$$
$$\boldsymbol{c}_i = 3 \cdot \boldsymbol{a}_i + \boldsymbol{x}_i \tag{2.88}$$

We first show that a solution to ORD gives a solution of SEP. Suppose indeed that for all pairs of points $\{\boldsymbol{q}, \boldsymbol{p}\}, \{\boldsymbol{x}, \boldsymbol{y}\}$ we know whether $d_W(\boldsymbol{q}, \boldsymbol{p}) \leq d_W(\boldsymbol{x}, \boldsymbol{y})$. This is equivalent to knowing the output of CMP$(\{\boldsymbol{q}, \boldsymbol{p}\}, \{\boldsymbol{x}, \boldsymbol{y}\})$, which is

$$\text{CMP}(\{\boldsymbol{q}, \boldsymbol{p}\}, \{\boldsymbol{x}, \boldsymbol{y}\}) = \operatorname{sign}\left\langle \boldsymbol{h}, (\boldsymbol{q} - \boldsymbol{p})^2 - (\boldsymbol{x} - \boldsymbol{y})^2 \right\rangle \tag{2.89}$$

Consider then the point $\boldsymbol{q} = \boldsymbol{c}_i, \boldsymbol{p} = \boldsymbol{x} = \boldsymbol{b}_i, \boldsymbol{y} = \boldsymbol{a}_i$ for each i. Then:

$$\begin{aligned} \text{CMP}(\{\boldsymbol{q}, \boldsymbol{p}\}, \{\boldsymbol{x}, \boldsymbol{y}\}) &= \operatorname{sign}\left\langle \boldsymbol{h}, (\boldsymbol{a}_i - \boldsymbol{b}_i)^2 - (\boldsymbol{b}_i - \boldsymbol{c}_i)^2 \right\rangle && (2.90) \\ &= \operatorname{sign}\left\langle \boldsymbol{h}, (\boldsymbol{a}_i)^2 - (-\boldsymbol{a}_i - \boldsymbol{x}_i)^2 \right\rangle && (2.91) \\ &= \operatorname{sign}\left\langle \boldsymbol{h}, 2 \cdot 6^i \boldsymbol{x}_i - \boldsymbol{x}_i^2 \right\rangle && (2.92) \end{aligned}$$

$$= \operatorname{sign} \left\langle \boldsymbol{h}, \boldsymbol{x}_i \left(1 - \frac{\boldsymbol{x}_i}{2 \cdot 6^i}\right)\right\rangle \tag{2.93}$$

By the margin hypothesis, for ϵ small enough this equals $\operatorname{sign}(\langle \boldsymbol{h}, \boldsymbol{x}_i \rangle)$, i.e., the label of $\boldsymbol{x}_i$ in SEP.

We now show that all the other queries reveal no information about the solution of SEP. Suppose then the points are not in the form $\boldsymbol{q} = \boldsymbol{c}_i, \boldsymbol{p} = \boldsymbol{x} = \boldsymbol{b}_i, \boldsymbol{y} = \boldsymbol{a}_i$. Without loss of generality, we can assume that $\boldsymbol{q} > \boldsymbol{p}$ and $\boldsymbol{q} \geq \boldsymbol{x} > \boldsymbol{y}$. It is then easy to see that, for ϵ small enough, $(\boldsymbol{q} - \boldsymbol{p})^2 - (\boldsymbol{x} - \boldsymbol{y})^2 > 0$ or $(\boldsymbol{q} - \boldsymbol{p})^2 - (\boldsymbol{x} - \boldsymbol{y})^2 < 0$. This holds independently of the $\boldsymbol{x}_i$ and of W and therefore gives no information about the solution of SEP.

It follows that, if we can solve ORD in $f(3n)$ CMP queries, then we can solve SEP in $f(n)$ queries. Finally, note that adding SCQ queries does not reduce the query complexity (e.g., let X lie in a single cluster). For the same reason, we can even assume an arbitrarily large cluster margin γ. □

It remains to show that SEP requires $\Omega(n)$ CMP queries in the worst case. This is well known, but we need to ensure that $\mathcal{H} \subset \mathbb{R}^d_+$ and that any $h \in \mathcal{H}$ supports a margin as described above.

Consider the following set $X = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\} \subseteq \mathbb{R}^3$:

$$\boldsymbol{x}_i = (1 - \delta, -\cos(\theta_i), -\sin(\theta_i)) \tag{2.94}$$

where $\theta_i = i\frac{\pi}{2n}$ and δ is sufficiently small. Let $\mathcal{H} = \{\boldsymbol{h}_1, \ldots, \boldsymbol{h}_n\}$, where

$$\boldsymbol{h}_j = (1, \cos(\theta_j), \sin(\theta_j)) \tag{2.95}$$

Note that $\mathcal{H} \subset \mathbb{R}^d_+$ as required. Clearly:

$$\langle \boldsymbol{h}_j, \boldsymbol{x}_i \rangle = \begin{cases} -\delta & \text{if } j = i \\ 1 - (\delta + \cos(\theta_i - \theta_j)) & \text{if } j \neq i \end{cases} \tag{2.96}$$

By choosing $\delta = \frac{1-\cos(\pi/2n)}{2}$ we have $\operatorname{sign} \langle \boldsymbol{h}, \boldsymbol{x}_i \rangle = -1$ if and only if $i = j$. Clearly, any algorithm needs to probe $\Omega(n)$ labels to learn h with constant probability for some $h \in \mathcal{H}$. Finally, note that any h supports a margin, as required. □

Chapter 3

Exact recovery of Margin-Based Clusterings

The scope of this chapter is to improve upon the results presented in Chapter 2, both in terms of generality and in terms of query complexity. In particular, we begin by introducing a simple but general notion of margin between clusters that captures, as special cases, the margins used in previous works (including that used in chapter 2), the classic SVM margin, and standard notions of stability for center-based clusterings. We then show that under this margin assumptions, there exist algorithms that recover all clusterings exactly using only $\mathcal{O}(\log n)$ queries in a variety of settings including convex clusters in $\mathbb{R}^m$ and possibly non-convex clusters in general pseudo-metric spaces. Finally for clusterings realized by binary concept classes we give a combinatorial characterization of recoverability with $\mathcal{O}(\log n)$ queries, and we show that, for many concept classes in Euclidean spaces, this characterization is equivalent to our margin condition.

3.1 Introduction

This chapter investigates the problem of exact cluster recovery using oracle queries, in the well-known framework introduced by Ashtiani et al. [2016]. We are given a set X of n points from some domain $\mathcal{X}$ (e.g., from the Euclidean m-dimensional space $\mathbb{R}^m$) and an oracle answering to same-cluster queries of the form "are these two points in the same cluster?" or, equivalently, to label queries of the form "which cluster does this point belong to?". The oracle answers are consistent with some clustering $\mathcal{C} = (C_1, \dots, C_k)$ of X unknown to the algorithm, where k is a fixed constant. The goal is to design an algorithm that recovers $\mathcal{C}$ deterministically by using as few queries as possible.

Clearly, if there are no restrictions on $\mathcal{C}$, then any algorithm needs n queries in the worst case. Thus, the question is what assumptions on $\mathcal{C}$ yield query-efficient algorithms. Since a good clustering is often thought of as having well-separated clusters, a natural assumption is that $\mathcal{C}$ satisfies some margin property; and previous work shows precisely that some margin properties yield cluster recovery algorithms that achieve the "gold standard" bound of $\mathcal{O}(\log n)$ queries. The first such algorithm appeared in Ashtiani et al. [2016] for the Euclidean case (i.e., when $X \subseteq \mathbb{R}^m$), with the following result. If every cluster C_i is separated from $X \setminus C_i$ by a ball that is centered in the center of mass of C_i, then $\mathcal{O}(\log n)$ queries are sufficient to recover $\mathcal{C}$ with high probability, provided that, for some fixed $\gamma > 0$, every ball does not include points of other clusters, even if expanded by a factor of $1 + \gamma$. The parameter γ is called *margin*, and the number of queries needed to recover $\mathcal{C}$ grows with $1/\gamma$.

In a first attempt at generalizing this result, Bressan et al. [2020a] showed that with $\mathcal{O}(\log n)$ queries one can actually recover clusters with *ellipsoidal* separators, at the price of a dependence on γ and m of approximately $(m/\gamma)^m$. Interestingly, this result was achieved via boosting of one-sided error learning, which works as follows. Suppose that, by making $\mathcal{O}(1)$ queries, we could identify correctly (with zero mistakes) a constant fraction of the points in some cluster C_i. Then, we could label those points as i, remove them from the dataset, and repeat. It is not hard to show that, after $\mathcal{O}(\log n)$ rounds, we have correctly labeled all the input points with high probability. The difficult task is, of course, using only $\mathcal{O}(1)$ queries to identify correctly a constant fraction of some cluster C_i (this is called learning with one-sided error, since all points predicted to be in C_i are indeed in C_i, while points predicted *not* to be in C_i can be anywhere). The key insight of Bressan et al. [2020a] is that, if the clusters have margin γ with respect to their ellipsoidal separators, then roughly $(m/\gamma)^m$ queries are sufficient. This leads to the following question: how much can this margin-based approach be extended?

3.1.1 Contributions

In this chapter we provide several answers, revealing how margin-based cluster recovery and one-sided error learning are intimately connected. Our main contributions are as follows.

1. We introduce a new notion of margin in Euclidean spaces, that we call "convex hull margin" (Definition 3.2). This is a strict generalization of the margins of [Bressan et al., 2020a; Ashtiani et al., 2016] and of the usual SVM margin, and allows the clusters to have *any shape whatsoever* as long as they are convex. Under the convex hull margin, we develop a novel technique for learning with one-sided error that we call *convex hull expansion trick*. It essentially amounts to sampling many points from a single cluster and "inflate" their convex hull by a factor of $(1+\gamma)$. This simple technique (whose proof, however, is quite technical) yields an algorithm for exact cluster recovery in $\mathbb{R}^m$ which runs in polynomial time and makes $\mathcal{O}(\log n)$ queries (Theorem 3.3). The dependence of the query bound on γ and m is of order $(1+1/\gamma)^m$, which is significantly better than [Bressan et al., 2020a] and closer to the query complexity lower bound of order $(1+1/\gamma)^{m/2}$.
2. We introduce a notion of cluster margin for general pseudometric spaces called *one-versus-all margin* (Definition 3.5). This notion of margin is strictly more general than convex hull margin, and captures as special cases some standard notions of stability for hard clustering problems, such as k-means or k-centers. We show that, if a clustering has one-versus-all margin, then it can be recovered with $\mathcal{O}(\log n)$ queries by an algorithm that uses a purely learning-based approach (Theorem 3.10). The $\mathcal{O}(\log n)$ query bound hides a dependence on the complexity of the pseudometric space expressed in terms of its packing number. We show that such a dependence is essentially optimal, thus characterizing the recoverability of clusterings in this setting.
3. Finally, we show a new connection between margin-based learning and exact active cluster recoverability, when clusters are realized by some concept class $\mathcal{H}$ (that is, when for each cluster C_i there is a concept $h_i \in \mathcal{H}$ such that $X \cap h_i = C_i$). We show that if a certain combinatorial parameter of $\mathcal{H}$, called *coslicing dimension*, is bounded, then one can learn clusterings with $\mathcal{O}(\log n)$ label queries; otherwise, $\Omega(n)$ queries are needed in the worst case (Theorem 3.12). Moreover, we show that if $\mathcal{H}$ is *any* concept class in $\mathbb{R}^m$ that is closed under affine transformations and well-behaved in a natural sense, then the clusters realized by $\mathcal{H}$ are recoverable with $\mathcal{O}(\log n)$ label queries if and only if they have positive one-versus-all margin (Theorem 3.14).

Our contributions have implications for active learning of binary and multiclass classifiers, too. More precisely, our $\mathcal{O}(\log n)$ query bounds imply $\tilde{\mathcal{O}}(\log 1/\epsilon)$ query bounds for pool-based active learning [McCallum and Nigam, 1998], where ϵ is the generalization error. To see this, draw a set X of $\Theta\big(\epsilon^{-1}(K \log 1/\epsilon + \log 1/\delta)\big)$ unlabeled samples from the underlying distribution, where K is the relevant measure of capacity (the VC-dimension or the Natarajan dimension), run our algorithms over X, and compute a hypothesis consistent with the recovered labeling $\mathcal{C}$. These types of reductions are standard in active learning, see for instance [Kane et al., 2017a].

Additional related work. Same-cluster queries are natural to implement in crowd-sourcing and for this reason they have been extensively studied both in theory [Ailon et al., 2018b,c; Gamlath et al., 2018; Huleihel et al., 2019; Mazumdar and Pal, 2017; Mazumdar and Saha, 2017b,a; Saha and Subramanian, 2019b; Vitale et al., 2019] and in practice [Firmani et al., 2018; Gruenheid et al., 2015; Verroios and Garcia-Molina, 2015; Verroios et al., 2017]. Note that same-cluster queries are essentially equivalent to label queries, the basic mechanism of active learning [Hanneke, 2014].

Various notions of margin are central in both active learning and cluster recovery [Xu et al., 2004; Balcan et al., 2007a; Balcan and Long, 2013a; Kane et al., 2017a; Bressan et al., 2021a]. Our coslicing dimension is similar to the slicing dimension of Kivinen [1995] and the star number of Hanneke and Yang [2015]. Our arguments based on packing numbers are similar to those based on the inference dimension of Kane et al. [2017a] or the lossless sample compression of [Hopkins et al., 2021], as we cannot infer the label of a point only when it is far from already-labeled points. Combinatorial characterizations of multiclass learning have been also proposed in the passive case by Ben-David et al. [1995]; Rubinstein et al. [2009]; Daniely and Shalev-Shwartz [2014]. Other learning settings related to one-sided and active learning are RPU learning [Rivest and Sloan, 1988] and perfect selective classification [El-Yaniv and Wiener, 2012] — see [Hopkins et al., 2019] for a discussion.

3.2 Preliminaries and notation

The input is a pair (X, O), where X is a set of n points from some domain $\mathcal{X}$, and O is a label oracle that, when queried on any $x \in X$, returns the cluster id $\mathcal{C}(x)$ of x. The oracle O is consistent with a *latent clustering* $\mathcal{C} = (C_1, \dots, C_k)$ of X, i.e., a k-uple of pairwise disjoint sets whose union is X.[1] Note that we allow clusters to be empty. Our goal is to recover $\mathcal{C}$ exactly, by making as few queries as possible to O. Queries can be made adaptively, that is, the j-th point to be queried can be chosen as a function of the answers to the first $j-1$ queries. We express the number of queries as a function of k, n, and other parameters to be introduced later. This setting is essentially equivalent to the semi-supervised active clustering (SSAC) framework of Ashtiani et al. [2016], where the oracle answers same-cluster queries $\textsc{scq}(x, y)$ that, for any two points $x, y \in X$, return TRUE iff $\mathcal{C}(x) = \mathcal{C}(y)$. We use label queries instead of SCQ queries only for simplicity. Indeed, any SCQ query can be emulated with two label queries; conversely, the label of any point can be learned with k SCQ queries, up to a relabeling of the clusters. This implies that our bounds on the number of queries for cluster recovery hold for an SCQ oracle as well, up to a multiplicative factor of k.

We often assume a metric or a pseudometric d over $\mathcal{X}$ (a pseudometric allows two distinct points to have distance 0). For any $X \subset \mathcal{X}$, we denote by $\phi_d(X) = \sup_{x,x' \in X} d(x, x')$ the diameter of X measured by d, and we define $\phi_d(\emptyset) = 0$. For any two sets $U, S \subset \mathcal{X}$, we denote by $d(U, S) = \inf_{x \in U, y \in S} d(U, S)$ their distance according to d, and we define $d(U, \emptyset) = \infty$. For any $X \subset \mathbb{R}^m$, we write $\operatorname{conv}(X)$ for the convex hull of X. The unit Euclidean sphere in $\mathbb{R}^m$ is $S^{m-1} = \{x \in \mathbb{R}^m : \|x\|_2 = 1\}$. We recall some learning-theoretic facts. Let $\mathcal{H}$ be an arbitrary collection of subsets of $\mathcal{X}$ (i.e., a concept class). The intersection class of $\mathcal{H}$ is $I(\mathcal{H}) = \bigcup_{i \in \mathbb{N}} \{h_1 \cap \ldots \cap h_i : h_1, \dots, h_i \in \mathcal{H}\}$. Given any $S \subset \mathcal{X}$ and any $S' \subseteq S$ realized by some $h^\star \in \mathcal{H}$, the smallest concept in $I(\mathcal{H})$ consistent with S' is defined as $h^\circ = \bigcap\{h \in \mathcal{H} : h \cap S = S'\}$. Note that $h^\circ \subseteq h^\star$. Finally, we recall the definition of learning with one-sided error:

Definition 3.1 (Kivinen [1995], Definition 4.4)**.** *An algorithm $\mathcal{A}$ learns $\mathcal{H}$ with one-sided error ϵ and confidence δ with r examples if, for any target concept $h^\star \in \mathcal{H}$ and any probability measure $\mathcal{P}$ over $\mathcal{X}$, by drawing r independent labeled examples from $\mathcal{P}$, the algorithm outputs a concept $h \subseteq h^\star$ such that $\mathcal{P}(h^\star \setminus h) \le \epsilon$ with probability at least $1 - \delta$.*

3.3 Margin-based exact recovery of clusters in Euclidean spaces

In this section, we consider the Euclidean setting $\mathcal{X} = \mathbb{R}^m$. We show that the ellipsoidal margin assumption of Bressan et al. [2020a] can be significantly generalized, while retaining the $\mathcal{O}(\log n)$ query complexity, by introducing what we call the *convex hull margin.* In a nutshell the convex hull margin says that, given any cluster C, any point not in C is separated by the convex hull of C by a distance at least γ times the diameter of C. Instead of using the Euclidean metric, however, we allow distances to be measured by *any* pseudometric over $\mathbb{R}^m$, which we do not need to know, and which may even differ from cluster to cluster. The only requirement is that the pseudometric be homogeneous and invariant under translation (i.e., induced by a seminorm).

[1]In line with previous works, we assume k is fixed and known.

Definition 3.2 (Convex hull margin)**.** *Let D be the family of all pseudometrics induced by the seminorms over $\mathbb{R}^m$, and let $X \subset \mathbb{R}^m$ be a finite set. A clustering $\mathcal{C} = (C_1, \dots, C_k)$ of X has convex hull margin γ if for every $i \in [k]$ there exists $d_i \in D$ such that:*

$$d_i\big(X \setminus C_i, \operatorname{conv}(C_i)\big) > \gamma\, \phi_{d_i}(C_i) \tag{3.1}$$

This new definition has a few interesting properties. First, it strictly generalizes the ellipsoidal margin of Bressan et al. [2020a] and the spherical margin of Ashtiani et al. [2016]. To see this, let D be the class of all pseudometrics over $\mathbb{R}^m$ that can be written as $d_W(x,y) = \langle x-y, W(x-y)\rangle$ for some positive semidefinite matrix $W \in \mathbb{R}^{m\times m}$ (for the spherical margin, take $W = rI$ where I is the identity matrix). Second, it strictly generalizes the classic SVM margin, which prescribes a distance of $\gamma\,\phi(X)$ between the clusters where $\phi(X)$ is the Euclidean diameter of X. Indeed, since every cluster has Euclidean diameter at most $\phi(X)$, a SVM margin of γ implies a convex hull margin of γ. On the other hand, there are cases with arbitrarily small SVM margin but arbitrarily large convex hull margin, as we see in Section 3.4. Third, the use of pseudometrics allows us to capture the Euclidean distance between the points after projection on a subspace $V \subset \mathbb{R}^m$, modelling scenarios where each cluster only "cares" about a certain subset of the features.

Under the convex hull margin, we give a polynomial-time algorithm, named CheatRec (for Convex Hull ExpAnsion Trick Recovery) that recovers $\mathcal{C}$ using $\mathcal{O}(\log n)$ queries.

Theorem 3.3. *Let (X, O) be an instance whose clustering $\mathcal{C}$ has convex hull margin $\gamma > 0$. Then* CheatRec(X, O, γ) *deterministically outputs $\mathcal{C}$, runs in time* $\operatorname{poly}(k, n, m)$, *and with high probability makes at most $\mathcal{O}\big(k^2 m^5 \, (1 + 1/\gamma)^m \log(1 + 1/\gamma) \log n\big)$ queries.*

To put this result in perspective, consider the algorithm of Bressan et al. [2020a]. Under an ellipsoidal margin of γ_{EL}, that algorithm achieves a query bound of roughly $\big(\frac{m}{\gamma_{EL}}\big)^m \log n$. One can check that their ellipsoidal margin of γ_{EL} implies[2] convex hull margin $\gamma \geq \frac{\gamma_{EL}}{3}$ for all $\gamma_{EL} \leq 1$. Hence, in this range, our dependence on γ is better by $\Theta(m^m)$ factors.

As we said, CheatRec is based on boosting learners with one-sided error. What makes CheatRec different, however, is the technique used to learn with one-sided error. To explain this, let us recall how the algorithm of Bressan et al. [2020a] works. Their idea is to learn an approximate ellipsoidal separator that contains a constant fraction of some cluster C_i, and then remove from it all points that do not belong to C_i. Crucially, to learn the approximate separator, the algorithm needs a number of queries proportional to the VC-dimension of the corresponding class, which for m-dimensional ellipsoids is $\mathcal{O}(m^2)$. Unfortunately, this approach does not work with the convex hull margin, since the class of allowed separators has unbounded VC-dimension (it is the class of all polytopes in $\mathbb{R}^m$).

To bypass this obstacle we develop a novel technique for learning with one-sided error that we call *convex hull expansion trick*. This technique essentially amounts to the following procedure: draw a large labeled sample from X, take all the sampled points that belong to the same cluster C, and inflate their convex hull by a factor $1 + \gamma$. This can be seen as a way to exploit the convex hull margin directly, without going through the VC-dimension of the separators. Such a trick may appear natural, but proving it to work is not trivial and requires a combination of results from probability, convex geometry, and PAC learning. In the rest of the section, we describe the trick and sketch the proof of Theorem 3.3. For the complete proof, see the appendix.

3.3.1 CheatRec and the convex hull expansion trick

The starting point of CheatRec is the following idea. Let s be a parameter to be set later. In each round, CheatRec draws from X a uniform random sample S of size $|S| = ks$. Then, using the oracle, it determines the partition $S_1, \dots, S_k$ induced by $\mathcal{C}$. As there are at most k clusters, for at least one of them (say cluster C) we have $|S_C| \geq s$, a fact that allow us to use PAC bounds later on. Now, let $K = \operatorname{conv}(S_C)$, and let $d \in D$ be the pseudometric that gives the margin of C. Since d is homogeneous and invariant under translation (recall

[2]Their definition of margin uses squared distances, so if a cluster has margin γ_{EL} by their definition, then it has convex hull margin $\sqrt{1+\gamma_{EL}} - 1$. This is why we need to make this apparently misplaced observation.

that D contains pseudometrics induced by seminorms), we know that any point y such that $d(y, K) \leq \gamma\phi_d(K)$ must belong to C as well. Therefore, if we pick any point $z \in K$ and compute the scaling $Q = (1+\gamma)K$ with respect to z, Q will not intersect clusters other than C. Hence, we can safely label all points in $X \cap Q$ as belonging to C. As long as we stick with this, we will not make mistakes — in other words, we learn with one-sided error. However, this is not sufficient: in order to make actual progress, we must guarantee that $X \cap Q$ contains a good fraction of C. It is not obvious why this should be true: in the worst case, $X \cap Q$ could simply coincide with S_C and we would have learned nothing.

It is here that the convex hull expansion trick enters the game. The key idea is to choose $z = \mu_K$, the center of mass of K. Since however computing μ_K is hard [Rademacher, 2007], we use an approximation. We give here the technical statement and a sketch of the proof (full proof in the appendix). The uniform probability measure $\mathcal{U}$ over K is defined by $\mathcal{U}(K') = \frac{\mathrm{vol}(K')}{\mathrm{vol}(K)}$ for all measurable $K' \subseteq K$. A probability measure $\mathcal{P}$ is ϵ-uniform if $|\mathcal{P}(K') - \mathcal{U}(K')| \leq \epsilon$ for all measurable $K' \subseteq K$.

Lemma 3.4 (Convex hull expansion trick). *Fix $\gamma > 0$, and let $s = \Theta\big(m^5 \big(1 + {}^1\!/\!_\gamma\big)^m \log\big(1 + {}^1\!/\!_\gamma\big)\big)$ large enough. Let S_C be a sample of s independent uniform random points from some cluster C, and let $K = \mathrm{conv}(S_C)$. Let $X_1, \ldots, X_N$ be independent random points sampled ϵ-uniformly from K, with $\epsilon \in \Theta(m^{-1})$ small enough and $N \in \Theta(m^2)$ large enough, and let $z = \frac{1}{N}\sum_{i=1}^N X_i$. Finally, let $Q = (1+\gamma)K$ where the center of the scaling is z. Then, with probability arbitrarily close to 1, we have $|Q \cap C| \geq \frac{1}{2}|C|$.*

Sketch of the proof. To begin, suppose that Q was obtained by scaling K about its own center of mass μ_K. Then, by a probabilistic argument, a result of Naszódi [2018] implies the existence of a polytope P on roughly $(1 + {}^1\!/\!_\gamma)^m$ vertices such that $K \subseteq P \subseteq Q$. Thus, if we can show that $X \cap P$ contains a good fraction of C, then $X \cap Q$ will contain a good fraction of C, too. To ensure that $X \cap P$ contains a good fraction of C, we observe that the class $\mathcal{P}_t$ of all polytopes on at most t vertices has VC-dimension at most $8m^2 t \log_2 t$, by a recent result [Kupavskii, 2020]. Hence, if we let $|S_C| = s \simeq (1 + {}^1\!/\!_\gamma)^m$, then by standard PAC bounds $X \cap P$ contains half of C with probability arbitrarily close to 1. This holds because, as $K \subseteq P \subseteq Q$, then P is consistent with S_C.

Now suppose that, in place of μ_K, we use $z = \frac{1}{N}\sum_{i=1}^N X_i$. If K is in isotropic position, then its radius is bounded by m, and therefore $\|X_i\|_2 \leq m$. This implies that, if $N = \Theta(m^2)$ and each X_i comes from a distribution that is $\Theta(1/m)$-uniform over K, then with high probability z is at distance $\eta = O(1)$ from μ_K. In particular, by increasing N by constant factors, we can make arbitrarily small the probability that $\eta \leq {}^1\!/\!_3$. Then, by adapting the result of Naszódi [2018] with an extension of Grunbaum's inequality for convex bodies due to Bertsimas and Vempala [2004], we can show that $|Q \cap C| \geq \frac{1}{2}|C|$ with probability arbitrarily large. □

We conclude with a note on how to implement CheatRec in polynomial time. The central point is to avoid computing $K = \mathrm{conv}(S_C)$ explicitly as an intersection of halfspaces, since this could take time $|S_C|^{\Theta(m)}$. Thus, we proceed as follows. First, we transform S_C so that $\mathrm{conv}(S_C)$ is in what is called a near-isotropic position. To this end, we compute John's ellipsoid for S_C, and apply the map that turns that ellipsoid into a ball, which takes polynomial time. Afterwards, we can draw $X_1, \ldots, X_N$ efficiently via the "hit-and-run from a corner" algorithm of Lovász and Vempala [2006], in time $\mathcal{O}\big(\mathrm{poly}(m, |S_C|) \log \frac{m}{\epsilon}\big)$ per sample, including the time to solve a linear program to determine when the walk hits the boundary of K. Once we have z, we rescale S_C about z itself, and label all points in $(1+\gamma)\,\mathrm{conv}(S_C)$ as belonging to C. This amounts to solving for every $x \in X$ a feasibility problem, which takes polynomial time. Finally, observe that the polytope P appearing in the proof is only for the analysis, and is never computed. For a complete discussion, see the full proof.

3.4 A more abstract view: the one-versus-all margin

In this section, we consider the case where $\mathcal{X}$ is a generic space equipped with a set of pseudometrics. Like in Section 3.3, we want to formulate a notion of margin between clusters. However, now we cannot express the margin in terms of the diameter of convex hulls (since $\mathcal{X}$ need not be a vector space and may have no notion of convexity at all), and we cannot

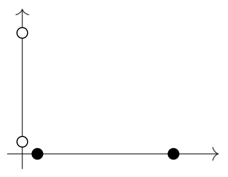


FIGURE 3.4.1: An instance with arbitrarily small SVM margin but unbounded one-versus-all margin.

apply any "expansion trick" (since the pseudometrics may not be homogeneous and invariant under translation). Instead, we adopt what we call the *one-versus-all margin*. We prove that clusterings with positive one-versus-all margin can be recovered again with $\mathcal{O}(\log n)$ oracle queries. However, we do not provide running time bounds like those of Section 3.3: our algorithm is essentially based on computing an ERM, which in general may take time superpolynomial in n.

Next, we introduce the one-versus-all margin.

Definition 3.5 (One-versus-all margin). *Choose k pseudometrics $d_1, \dots, d_k$ over $\mathcal{X}$. A clustering $\mathcal{C} = (C_1, \dots, C_k)$ of a finite set $X \subset \mathcal{X}$ has one-versus-all margin γ with respect to $d_1, \dots, d_k$ if for all $i \in [k]$ we have:*

$$d_i(X \setminus C_i, C_i) > \gamma\, \phi_{d_i}(C_i) \tag{3.2}$$

Remark. Unlike the convex hull margin, here the pseudometrics $d_1, \dots, d_k$ are fixed in advance. This is not a weakness of the above definition, but rather a strength of the convex hull margin, which allows one to recover clusters without even knowing $d_1, \dots, d_k$.

Note also that, just like the convex hull margin, the one-versus-all margin in $\mathbb{R}^m$ is strictly more general than the SVM margin. The fact that SVM margin implies one-versus-all margin follows by the observation in Section 3.3. The fact that one-versus-all margin does not imply SVM margin is shown in Lemma 3.6 below.

Lemma 3.6. *For any $u \in \mathbb{R}^2$ let $d_u(x, y) = |\langle u, x - y \rangle|$. For any $\eta > 0$ there exists a clustering $\mathcal{C} = (C_1, C_2)$ on a set $X \subset \mathbb{R}^2$ that has arbitrarily large one-versus-all margin with respect to $d_{(0,1)}, d_{(1,0)}$, and yet $d_u(C_1, C_2) \le \eta\, \phi_{d_u}(X)$ for any $u \in \mathbb{R}^2 \setminus \mathbf{0}$.*

Proof sketch. Consider Figure 3.4.1. The two points along the x axis belong to C_1, and the two points along the y axis belong to C_2. The pseudometric $d_1 = d_{(0,1)}$ measures the distance along the y axis, and the pseudometric $d_2 = d_{(1,0)}$ measures the distance along the x axis. It is easy to see that $d_1(C_1, C_2) > 0 = \phi_{d_1}(C_1)$, and that $d_2(C_1, C_2) > 0 = \phi_{d_2}(C_2)$. Hence, the one-versus-all margin of $\mathcal{C}$ with respect to d_1, d_2 is unbounded. Yet, for any $\eta > 0$ we can make $d_u(C_1, C_2) \le \eta\, \phi_{d_u}(X)$ for any nonzero vector $u \in \mathbb{R}^2$, by placing the endpoints of the clusters arbitrarily near the origin. □

3.4.1 Stability of center-based clusterings

Fix any pseudometric d over $\mathcal{X}$. A clustering $\mathcal{C} = (C_1, \dots, C_k)$ of X is *center-based* if there exist k points $c_1, \dots, c_k \in \mathcal{X}$, called *centers*, such that for every $i \in [k]$ and every $x \in C_i$ we have $d(x, c_j) > d(x, c_i)$ for all $j \ne i$. In other terms, every point is assigned to the nearest center. It is well known that many popular center-based clustering problems, such as k-means or k-centers, are NP-hard to solve in general. However, those problems become polynomial-time solvable (or approximable) if the solution $\mathcal{C}$ meets certain stability properties. Here we show that two of these properties, the α-center proximity of Awasthi et al. [2012a] and the $(1+\epsilon)$-perturbation resilience of Bilu and Linial [2012], imply a positive one-versus-all margin. Let us recall these properties. We define a $(1+\epsilon)$-perturbation of d as any function d' (which need not be a pseudometric) such that $d \le d' \le (1+\epsilon)d$.

Definition 3.7. *Let $\mathcal{C}$ be a center-based clustering.*

- *$\mathcal{C}$ satisfies α-center proximity with $\alpha > 1$ if, for all $i \in [k]$, for all $x \in C_i$ and all $j \ne i$ we have $d(x, c_j) > \alpha\, d(x, c_i)$.*

- *$\mathcal{C}$ is $(1+\epsilon)$-perturbation resilient with $\epsilon > 0$ if it is induced by the same centers $c_1, \ldots, c_k$ under any $(1+\epsilon)$-perturbation of d.*

It is known that $(1+\epsilon)$-perturbation resilience implies α-center proximity with $\alpha = 1+\epsilon$, see [Awasthi et al., 2012a]. Our result is:

Theorem 3.8. *If $\mathcal{C}$ satisfies α-center proximity, then it has one-versus-all margin at least $\frac{(\alpha-1)^2}{2(\alpha+1)}$. Hence, if $\mathcal{C}$ satisfies $(1+\epsilon)$-perturbation stability, then it has one-versus-all margin at least $\frac{\epsilon^2}{2(\epsilon+2)}$.*

The proof of Theorem 3.8 is found in the appendix.

3.4.2 Cluster recovery with one-versus-all margin

We conclude by showing that, if $\mathcal{C}$ has one-versus-all margin $\gamma > 0$, then one can recover $\mathcal{C}$ with $\mathcal{O}(\log n)$ queries. Our algorithm, mRec, considers the set of all possible clusters that satisfy the margin, and for each label selects the smallest hypotheses consistent with the sampled points. To prove that this approach works, we establish a formal connection between one-versus-all margin and one-sided-error learnability of the concept classes induced by all possible clusters with margin.

We need some further notation. As usual, let d be any pseudometric over $\mathcal{X}$. For any $X \subset \mathcal{X}$ and any $r > 0$, we denote by $\mathcal{M}(X, r, d)$ the maximum cardinality of any $A \subseteq X$ such that $d(x, y) > r$ for all distinct $x, y \in A$. From now on we assume that $\mathcal{M}(X, r, d)$ is bounded, and for any $\gamma > 0$, we define $M(\gamma, d) = \max\{\mathcal{M}(B(x, r), \gamma r, d) : x \in \mathcal{X}, r > 0\}$. Hence, for any $\gamma > 0$, any ball B in $\mathcal{X}$ contains at most $M(\gamma, d)$ points at pairwise distance greater than γ times the radius of B, and some B attains this bound.[3] Finally, by vc-dim(H, X) we denote the VC-dimension of a generic concept class H over a set X.

Lemma 3.9 (One-versus-all margin implies one-sided-error learnability)**.** *Let d be any pseudometric over $\mathcal{X}$. For any finite $X \subset \mathcal{X}$ and any $\gamma > 0$, define the effective concept class over X:*

$$H = \{C \subseteq X \,:\, d(X \setminus C, C) > \gamma\, \phi_d(C)\} \tag{3.3}$$

Then $H = I(H)$, and $\text{vc-dim}(H, X) \leq M$ where $M = \max(2, M(\gamma, d))$. As a consequence, H can be learned with one-sided error ϵ and confidence δ with $\mathcal{O}\big(\epsilon^{-2}(M \log {}^1\!/\!_\epsilon + {}^1\!/\!_\delta)\big)$ examples by choosing the smallest consistent hypothesis in H.

Sketch of the proof. To prove that $H = I(H)$, one can take any two $C_1, C_2 \in H$ and show that $C_1 \cap C_2$ satisfies the margin condition, too. To prove that $\text{vc-dim}(H, X) \leq M$, we have two steps. First, let $\text{sl}(H, X)$ be the *slicing dimension* of H. This is the size of the largest subset $S \subseteq X$ *sliced* by H, i.e., such that for every $x \in S$ there is $C \in H$ giving $S \setminus x = S \cap C$, see [Kivinen, 1995]. As the same work shows, we have $\text{vc-dim}(I(H), X) \leq \text{sl}(H, X)$; hence, to prove $\text{vc-dim}(H, X) \leq M$ it suffices to prove that $\text{sl}(H, X) \leq M$. To this end, we use a packing argument. Suppose that $S \subseteq X$ is sliced by H, choose any $x \in S$, and let $C \in H$ such that $S \setminus x = S \cap C$. By construction of H, we know that $d(C, x) > \gamma \phi_d(C)$. Since $S \setminus x \subseteq C$, this yields:

$$d\big(S \setminus x, x\big) \geq d(C, x) > \gamma\, \phi_d(C) \geq \gamma\, \phi_d(S \setminus x) \tag{3.4}$$

It can be shown that this implies $d(S \setminus x, x) > \gamma \phi_d(S)$ for all $x \in S$, and, in turn, $|S| \leq M$. The claim on the learnability with one-sided error holds by choice of the smallest consistent hypothesis in $H = I(H)$, combined with standard PAC bounds. □

We can now present our main result. We let $M(\gamma) = \max_{d \in \{d_1, \ldots, d_k\}} M(\gamma, d)$, with $d_1, \ldots, d_k$ as in Definition 3.5.

[3] If $\mathcal{M}(X, r, d)$ is not bounded, then our results can be extended in the natural way, that is, we can prove a lower bound of $\Omega(n)$ queries for instances of n points.

Theorem 3.10. *Let (X, O) be any instance whose latent clustering $\mathcal{C}$ has one-versus-all margin $\gamma > 0$ with respect to $d_1, \ldots, d_k$. Then* mRec(X, O, γ) *deterministically outputs $\mathcal{C}$ while making, with high probability, at most $\mathcal{O}(Mk \log k \log n)$ queries to O, where $M = \max(2, M(\gamma))$. Moreover, for any algorithm $\mathcal{A}$ and for any $\gamma > 0$, there are instances with one-versus-all margin γ on which $\mathcal{A}$ makes $\Omega(M(2\gamma))$ queries in expectation.*

Sketch of the proof. For the lower bounds, we take a set X on $M(2\gamma)$ points at pairwise distance larger than 2γ times the radius of X, which is at least γ times the diameter of X, and we draw a random clustering $\mathcal{C}$ in the form $(x, X \setminus x)$. One can see that $\mathcal{C}$ has one-versus-all margin γ, and simple arguments, coupled with Yao's principle for Monte Carlo algorithms, show that any algorithm needs $\Omega(M(2\gamma))$ queries in the worst case to return $\mathcal{C}$. For the upper bounds, we show how to learn an expected constant fraction of X with one-sided error using $\Theta(Mk \lg k)$ queries; the rest follows by our general boosting argument. To begin, for each $i \in [k]$ we let $H_i = \{C \subseteq X : d_i(C, X \setminus C) > \gamma\, \phi_{d_i}(C)\}$. We then set $\epsilon = 1/2k$ and $\delta = 1/2$, and draw a labeled sample S of size $\Theta(\epsilon^{-1}(M \log 1/\epsilon + \log 1/\delta)) = \Theta(Mk \log k)$. Finally, for each $i \in [k]$ we choose the smallest hypothesis $\hat{C}_i \in H_i$ consistent with the subset $S_i \subseteq S$ labeled as i, and we assign label i to all points in $\hat{C}_i$. By Lemma 3.9, with probability at least $1/2$ we have $|\hat{C}_i| \geq |C_i| - \epsilon|X| = |C_i| - |X|/2k$. As $|\hat{C}_i| \geq 0$, this implies $\mathbb{E}|\hat{C}_i| \geq |C_i|/2 - |X|/4k$. By summing over all i, this shows that we are labelling correctly at least $|X|/4$ points in expectation. □

Remark. By Theorem 3.10, in $\mathbb{R}^m$ mRec yields a $\mathcal{O}(\log n)$ query bound even when the clusters are *not* convex. However, this does not mean that mRec subsumes CheatRec. First, as noted above, here $d_1, \ldots, d_k$ are known in advance. Second, mRec works by computing the smallest hypothesis $\hat{C}_i$ consistent with S_i (see the proof of Theorem 3.10), which in general may take superpolynomial time. Indeed, CheatRec runs in polynomial time by *not* computing $\hat{C}_i$ at all.

3.5 One-versus-all clusterings

In this section, we study active cluster recovery when the clusters can be realized by binary concepts from some concept class $\mathcal{H}$. In other words, this is the clustering equivalent of the classic one-versus-all formulation of multiclass classifiers (see, e.g., [Shalev-Shwartz and Ben-David, 2014a]). We show that the active recoverability of clusterings in this setting is driven by the *coslicing dimension* of $\mathcal{H}$, a combinatorial quantity similar to the star number of Hanneke and Yang [2015] and the slicing dimension of [Kivinen, 1995]. We also show that, for many natural concept classes in $\mathbb{R}^m$, a bounded coslicing dimension is equivalent to a positive one-versus-all margin.

As usual, let $\mathcal{X}$ be any domain, $X \subset \mathcal{X}$ be any finite set, and $\mathcal{C} = (C_1, \ldots, C_k)$ be a clustering of X. Let $\mathcal{H}$ be any concept class over $\mathcal{X}$. We say that $\mathcal{C}$ is realized by $\mathcal{H}$ if for all $i \in [k]$ there is some $h_i \in \mathcal{H}$ such that $C_i = X \cap h_i$. For example, the ellipsoidal clusters of Bressan et al. [2020a] can be formulated by letting $\mathcal{X} = \mathbb{R}^m$ and letting $\mathcal{H}$ to be the family of all ellipsoids in $\mathbb{R}^m$, and the convex clusters of Section 3.3 can be formulated by letting $\mathcal{X} = \mathbb{R}^m$ and letting $\mathcal{H}$ to be the family of all polytopes in $\mathbb{R}^m$. Clearly, we expect that the number of active queries needed to recover $\mathcal{C}$ depends on the complexity of $\mathcal{H}$. This leads us to the following question: what can we say about the recoverability of $\mathcal{C}$ in terms of the concept class $\mathcal{H}$?

3.5.1 A general characterization: the coslicing dimension

In this section, we characterize the active recoverability of $\mathcal{C}$ via the *coslicing dimension* of $\mathcal{H}$, a combinatorial quantity similar to the slicing dimension of Kivinen [1995] and the star number of Hanneke and Yang [2015].

Definition 3.11. *We say that $\mathcal{H}$ coslices $X \subseteq \mathcal{X}$ if for all $x \in X$ there exist two concepts $h_x^+, h_x^- \in \mathcal{H}$ such that $X \cap h_x^+ = \{x\}$ and $X \cap h_x^- = X \setminus \{x\}$. The coslicing dimension of $\mathcal{H}$ is:*

$$\operatorname{cosl}(\mathcal{H}) = \sup\{|X| : X \text{ is cosliced by } \mathcal{H}\}$$

If $\mathcal{H}$ coslices arbitrarily large sets then we let $\operatorname{cosl}(\mathcal{H}) = \infty$.

For instance, let $\mathcal{X} = \mathbb{R}^m$. If $\mathcal{H}$ is the class of linear separators, then $\operatorname{cosl}(\mathcal{H}) = \infty$ (take X to be the set of vertices of an n-vertex polytope and use the hyperplane separator theorem). If instead $\mathcal{H}$ is the class of axis-aligned boxes, it can be shown that $\operatorname{cosl}(\mathcal{H}) = 2m$. Our main result is:

Theorem 3.12. *If* $\operatorname{cosl}(\mathcal{H}) < \infty$ *then there is an algorithm that, given any n-point instance whose latent clustering $\mathcal{C}$ is realized by $\mathcal{H}$, recovers $\mathcal{C}$ with* $\mathcal{O}(\operatorname{cosl}(\mathcal{H})\, k \log k \log n)$ *queries with high probability. Moreover, for any algorithm $\mathcal{A}$ there are instances on* $\operatorname{cosl}(\mathcal{H})$ *points whose latent clustering $\mathcal{C}$ is realized by $\mathcal{H}$ where $\mathcal{A}$ makes* $\Omega(\operatorname{cosl}(\mathcal{H}))$ *queries in expectation to return $\mathcal{C}$. As a consequence, if* $\operatorname{cosl}(\mathcal{H}) = \infty$ *then any algorithm needs $\Omega(n)$ queries in expectation to recover an n-point clustering realized by $\mathcal{H}$.*

Sketch of the proof. The proof follows the same ideas of the proof of Theorem 3.10. For the lower bounds, we take any X cosliced by $\mathcal{H}$ with $|X| = \operatorname{cosl}(\mathcal{H})$, and we draw a random clustering of X in the form $(x, X \setminus x)$. For the upper bounds, for each $i \in [k]$ we define:

$$H_i = \{C \,:\, C = C'_i \wedge (C'_1, \ldots, C'_k) \in P_k(X)\} \tag{3.5}$$

where $P_k(X)$ is the set of all clusterings of X realized by $\mathcal{H}$. As observed in the proof of Lemma 3.9, we have the general relationship $\text{vc-dim}(I(H_i), X) \leq \operatorname{sl}(H_i, X)$ whenever $\operatorname{sl}(H_i, X) < \infty$. Therefore, if we show that $\operatorname{sl}(H_i, X) \leq \operatorname{cosl}(\mathcal{H})$, by drawing a labeled sample of size $\Theta(\operatorname{cosl}(\mathcal{H})\, k \log k)$ we can recover the labels of an expected constant fraction of X, as in the proof of Lemma 3.9. To prove that $\operatorname{sl}(H_i, X) \leq \operatorname{cosl}(\mathcal{H})$, let $U = \{x_1, \ldots, x_\ell\} \subseteq X$ be sliced by H_i. By construction of H_i, there are ℓ clusterings $\mathcal{C}_1, \ldots, \mathcal{C}_\ell$ realized by $\mathcal{H}$ and such that $\mathcal{C}_j = (x_j, U \setminus x_j)$ for all $j \in [\ell]$. This implies that U is cosliced by $\mathcal{H}$. Hence, $|U| \leq \operatorname{cosl}(\mathcal{H})$ and so $\operatorname{sl}(H_i, X) \leq \operatorname{cosl}(\mathcal{H})$. □

Remark. In the case of convex clusters in $\mathbb{R}^m$, we would let $\mathcal{H}$ be the class of all convex polytopes, obtaining $\operatorname{cosl}(\mathcal{H}) = \infty$ and thus Theorem 3.12 would not provide any useful bound. This is true even if $\mathcal{C}$ has convex hull margin $\gamma > 0$, although by Theorem 3.3 we know that $\mathcal{C}$ can be recovered with $\mathcal{O}(\log n)$ queries. The same holds for the one-versus-all margin. This is because we defined the coslicing dimension as a function of $\mathcal{H}$, whereas the margin depends on the instance (X, O). To fix this, one can redefine the coslicing dimension in the form $\operatorname{cosl}(\mathcal{H}, \mathcal{I})$ where $\mathcal{I}$ is a class of instances, and adapt Theorem 3.12 correspondingly. Then, if every instance $(X, O) \in \mathcal{I}$ has margin $\gamma > 0$, one can bound $\operatorname{cosl}(\mathcal{H}, \mathcal{I})$ as a function of γ, retrieving the same type of bounds of Theorem 3.3 and Theorem 3.10.

3.5.2 The one-versus-all margin, again!

We look again at the Euclidean case, $\mathcal{X} = \mathbb{R}^m$. Consider any concept class $\mathcal{H}$. Theorem 3.12 and the above remark say that, if $\operatorname{cosl}(\mathcal{H}, \mathcal{I}) < \infty$, where $\mathcal{I}$ is the class of allowed instances, then any clustering $\mathcal{C}$ realized by $\mathcal{H}$ can be recovered with $\mathcal{O}(\log n)$ queries, with no need for the one-versus-all margin (Definition 3.5). We show that, for a wide family of concept classes in $\mathbb{R}^m$, both things are actually equivalent: we can achieve the $\mathcal{O}(\log n)$ bound *if and only if* the instances have positive one-versus-all margin. This establishes a connection between one-versus-all margin and active learnability of clusterings realized by concept classes in $\mathbb{R}^m$. In what follows, we assume that $\mathcal{H}$ satisfies:

Definition 3.13. *A concept class $\mathcal{H}$ in $\mathbb{R}^m$ is non-fractal if there is $h \in \mathcal{H}$ such that both h and its complement contain a ball of positive radius.*

This assumption avoids pathological cases (for instance, $\mathcal{H}$ containing only hypotheses that are affine transformations of Cantor's set). Our result is:

Theorem 3.14. *Let $\mathcal{H}$ be a concept class in $\mathbb{R}^m$ that is non-fractal and closed under affine transformations. There is an algorithm that, given any instance whose latent clustering $\mathcal{C}$ has one-versus-all margin γ and is realized by $\mathcal{H}$, returns $\mathcal{C}$ while making $\mathcal{O}(Mk \log k \log n)$ queries with high probability, where* $M = \max\left(2, (1 + 4/\gamma)^m\right)$. *Moreover, for any algorithm $\mathcal{A}$,*

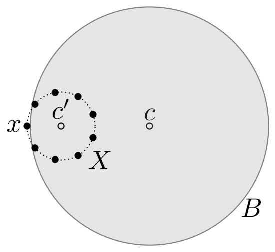


FIGURE 3.5.1: The η-packing X is in B, and thus in h, except for x that lies in $\bar{h}$. By taking B arbitrarily close to x, we can make η arbitrarily small and thus X arbitrarily large.

there exist arbitrarily large n-point instances, whose latent clustering $\mathcal{C}$ has arbitrarily small one-versus-all margin and is realized by $\mathcal{H}$, where $\mathcal{A}$ makes $\Omega(n)$ queries in expectation to recover $\mathcal{C}$.

Sketch of the proof. The upper bounds follow by Theorem 3.10 and the packing number of $\mathbb{R}^m$. For the lower bounds, we show that arbitrarily large packings of a sphere are cosliced by $\mathcal{H}$. We use Figure 3.5.1 for reference. Let $h \in \mathbb{R}^m$ be such that both h and its complement $\bar{h}$ contain a ball of positive radius. Then, for any $\rho > 0$ there exist a ball $B = B(c, r) \subseteq h$ with $r > 0$, and a point $x \in \bar{h}$ such that $d(B, x) \leq \rho$. Now take a sphere S of radius $r' \ll r$ with center on the segment xc. Let $\eta = \sup_{y \in S \setminus B} d(x, y)$, and let X be an η-packing of S, that is, a subset of points of S such that $d(x', x'') > \eta$ for all distinct $x', x'' \in X$. Note how this implies that every $x' \in X \setminus x$ necessarily lies in $S \cap B$, and therefore, in $S \cap h$. Moreover, by letting $\eta/r' \to 0$, we can take X arbitrarily large. Since $\mathcal{H}$ is closed under affine transformations, by rotating X it follows that for every $x \in X$ there is $h_x \in \mathcal{H}$ such that $X \cap h_x = X \setminus x$. By applying the same argument to $\bar{h}$ and by complementation we can show that $X \cap h'_x = \{x\}$ for some $h'_x \in \mathcal{H}$ for all $x \in X$ as well. Hence $\mathcal{H}$ coslices arbitrarily large sets. To conclude, invoke Theorem 3.12. □

Note that Theorem 3.14 applies to several basic concept classes $\mathcal{H}$. For instance, when $\mathcal{H}$ is the class of all linear separators, the class of all ellipsoids, the class of all polytopes, and the class of all convex bodies (bounded or not, and possibly degenerate). It also includes more complex classes whose hypotheses are not convex: for instance, the class of all finite or infinite disjoint unions of balls, polytopes, or convex bodies.

Appendix

3.A Proof of Lemma 3.4

Lemma 3.4 (Convex hull expansion trick)**.** *Fix $\gamma > 0$, and let $s = \Theta\big(m^5\big(1+{}^1\!/\!_\gamma\big)^m \log\big(1+{}^1\!/\!_\gamma\big)\big)$ large enough. Let S_C be a sample of s independent uniform random points from some cluster C, and let $K = \operatorname{conv}(S_C)$. Let $X_1, \dots, X_N$ be independent random points sampled ϵ-uniformly from K, with $\epsilon \in \Theta(m^{-1})$ small enough and $N \in \Theta(m^2)$ large enough, and let $z = \frac{1}{N}\sum_{i=1}^N X_i$. Finally, let $Q = (1+\gamma)K$ where the center of the scaling is z. Then, with probability arbitrarily close to 1, we have $|Q \cap C| \geq \frac{1}{2}|C|$.*

Preliminaries. Without loss of generality, we assume that K has full rank (as one can always work in the subspace spanned by S_C, which can be computed in time $\mathcal{O}(|S_C|m)$). For technical reasons, we let $\vartheta = \frac{1}{1+\gamma} \in (0,1)$ and prove the lemma for $s = \Omega\big(\frac{m^5}{\vartheta(1-\vartheta)^m}\ln\frac{1}{\vartheta(1-\vartheta)}\big)$ large enough. To see that any $s \in \Theta\big(m^5\big(1+{}^1\!/\!_\gamma\big)^m \ln\big(1+{}^1\!/\!_\gamma\big)\big)$ satisfies this assumption, first substitute ϑ to get:

$$\frac{1}{\vartheta(1-\vartheta)^m}\ln\frac{1}{\vartheta(1-\vartheta)} = (1+\gamma)\left(\frac{1+\gamma}{\gamma}\right)^m \ln\frac{(1+\gamma)^2}{\gamma} \tag{3.6}$$

which is in $\mathcal{O}\big((1+{}^1\!/\!_\gamma)^m(1+\gamma)\ln(1+{}^1\!/\!_\gamma)\big)$. Now note that $(1+\gamma)\ln\big(1+{}^1\!/\!_\gamma\big)$ is bounded by $\mathcal{O}\big(\gamma\cdot{}^1\!/\!_\gamma\big) = \mathcal{O}(1)$ for $\gamma > 1$, and by $2\ln\big(1+{}^1\!/\!_\gamma\big) = \mathcal{O}\big(\ln{}^1\!/\!_\gamma\big)$ when $\gamma \leq 1$. Hence, in any case the term $(1+\gamma)\ln(1+{}^1\!/\!_\gamma)$ is in $\mathcal{O}(\ln(1+{}^1\!/\!_\gamma))$. Therefore:

$$\big(1+{}^1\!/\!_\gamma\big)^m \ln\big(1+{}^1\!/\!_\gamma\big) = \Omega\left(\frac{1}{\vartheta(1-\vartheta)^m}\ln\frac{1}{\vartheta(1-\vartheta)}\right) \tag{3.7}$$

as claimed.

Before starting with the actual proof, we introduce some further definitions and notation.

Definition 3.15. *A convex body $K \subset \mathbb{R}^m$ is in* isotropic position[4] *if it has center of mass in the origin, $\int_K x\,dx = 0$, and moment of inertia 1 in every direction, $\int_K \langle x,u\rangle^2\,dx = 1$ for all $u \in S^{m-1}$.*

We define the norm $\|\cdot\|_K = \|f(\cdot)\|_2$ where $f = f_K$ is the unique affine transformation such that $f(K)$ is in isotropic position. We let $R(K) = \sup_{x\in K}\|x\|_2$ denote the Euclidean radius of K, and we let $R_K(K) = \sup_{x\in K}\|x\|_K$ denote the isotropic radius of K. We also let $d_K(x,y) = \|x-y\|_K$ be the isotropic distance of K.

Now, the proof has two steps. First, we show that the point $z = \frac{1}{N}\sum_{i=1}^N X_i$ is close to the centroid μ_K of K, according to $d_K(\cdot)$, with good probability. Second, we show that if this is the case, then $\frac{K}{\vartheta}$, where the scaling is meant about z, contains a polytope P which contains K and thus S_C, and which belongs to a class with VC dimension $\mathcal{O}\big(\frac{m^5}{\vartheta(1-\vartheta)^m}\big)$. By standard PAC bounds this implies that $|P \cap C| \geq \frac{1}{2}|C|$, with a probability that can be made arbitrarily close to 1 by adjusting the constants.

Step 1: z is close to the centroid of K

Let μ_K be the center of mass of K. We prove:

[4] Not to be confused with the definition of [Giannopoulos, 2003], where the assumption $\int_K \langle x,u\rangle^2\,dx = 1$ is replaced by $\operatorname{vol}(K) = 1$.

Lemma 3.16. *Fix any $\eta, p \geq 0$, and choose any $\epsilon \leq \frac{\eta}{4(m+1)}$ and $N \geq \frac{16(m+1)^2}{p^2\eta^2}$. If $X_1, \ldots, X_N$ are drawn independently and ϵ-uniformly at random from K, and $\bar{X} = \frac{1}{N} X_i$, then:*

$$\mathbb{P}\left(d_K\left(\bar{X}, \mu_K\right) \leq \eta\right) \geq 1 - p \tag{3.8}$$

For the proof of Lemma 3.16, we need two ancillary results.

Lemma 3.17. *$R_K(K) \leq m + 1$.*

Proof. Consider K in isotropic position, and let $K' = \vartheta K$ where $\vartheta = \mathrm{vol}(K)^{-1/m}$, so that $\mathrm{vol}(K') = 1$. Then, K' is in isotropic position according to the definition of Giannopoulos [2003]. In this case, [Giannopoulos, 2003, Theorem 1.2.4] implies $R(K') \leq (m+1)L_K$, where L_K is the *isotropic constant* which, for all $u \in S^{m-1}$, satisfies $\int_{K'} \langle x, u \rangle^2 \, dx = L_K^2$. Since $K' = \vartheta K$ and $\int_K \langle x, u \rangle^2 \, dx = 1$ by the isotropy of K, we have $L_K = \vartheta$. Hence $R(K') \leq (m+1)\vartheta$, that is, $R(K) \leq m + 1$. □

Lemma 3.18. *If X is drawn from an ϵ-uniform distribution over K, then $\|\mathbb{E}X\|_K \leq 2\epsilon(m+1)$.*

Proof. Since X is ϵ-uniform over K, there exists a coupling (X, Y) with $\mathbb{P}(X \neq Y) \leq \epsilon$ and Y uniform over K. Since $\|\mathbb{E}Y\|_K = 0$, we have:

$$\|\mathbb{E}X\|_K = \|\mathbb{E}[X - Y]\|_K \leq \mathbb{P}(X \neq Y) \sup_{x,y \in K} d_K(x, y) \leq \epsilon \, 2R_K(K) \leq 2\epsilon(m+1) \tag{3.9}$$

where the last inequality is given by Lemma 3.17. □

Proof of Lemma 3.16. For the sake of the analysis we look at K from its isotropic position. Note that the X_i are still ϵ-uniform over K, since the affine map that places K in isotropic position preserves volume ratios. The claim becomes:

$$\mathbb{P}\big(\|\bar{X}\|_2 \leq \eta\big) \geq 1 - p \tag{3.10}$$

Now, $\|\bar{X}\|_2 \leq \|\mathbb{E}\bar{X}\|_2 + \|\bar{X} - \mathbb{E}\bar{X}\|_2$. Thus, we show that $\|\mathbb{E}\bar{X}\|_2 \leq \frac{\eta}{2}$, and that $\|\bar{X} - \mathbb{E}\bar{X}\|_2 \leq \frac{\eta}{2}$ with probability at least $1 - p$. For the first part, by Lemma 3.18, and since $\epsilon \leq \frac{\eta}{4(m+1)}$, we obtain:

$$\|\mathbb{E}\bar{X}\|_2 = \|\mathbb{E}X_i\|_2 \leq 2\epsilon(m+1) \leq \frac{\eta}{2} \tag{3.11}$$

For the second part, by Lemma 3.17 we have $\|X_i\|_2 \leq m + 1$ for all i. Therefore, if we let $Y_i = X_i - \mathbb{E}X_i$ for all i, we have $\|Y_i\|_2 \leq 2(m+1)$ and thus $\|Y_i\|_2^2 \leq 4(m+1)^2$. Now let $\bar{Y} = \frac{1}{N}\sum_{i=1}^N Y_i$. Since the Y_i are independent and with $\mathbb{E}Y_i = 0$, then $\mathbb{E}\langle Y_i, Y_j \rangle = 0$ whenever $i \neq j$ and therefore:

$$\mathbb{E}\|\bar{Y}\|_2^2 = \mathbb{E}\left(\frac{1}{N^2}\sum_{i,j=1}^N \langle Y_i, Y_j \rangle\right) = \frac{1}{N^2}\sum_{i=1}^N \mathbb{E}\|Y_i\|_2^2 \leq \frac{4(m+1)^2}{N} \tag{3.12}$$

Plugging in our value $N \geq \frac{16(m+1)^2}{p^2\eta^2}$, and using Jensen's inequality, we obtain:

$$\left(\mathbb{E}\|\bar{Y}\|_2\right)^2 \leq \mathbb{E}\|\bar{Y}\|_2^2 \leq \frac{p^2\eta^2}{4} \tag{3.13}$$

Therefore $\mathbb{E}\|\bar{Y}\|_2 \leq \frac{p\eta}{2}$, which by Markov's inequality implies that $\mathbb{P}\big(\|\bar{Y}\|_2 > \frac{\eta}{2}\big) < p$. By noting that $\bar{Y} = \bar{X} - \mathbb{E}\bar{X}$, the proof is complete. □

Step 2: showing a tight enclosing polytope

We prove:

Lemma 3.19. *Let $z \in \mathbb{R}^m$ such that $d_K(z, \mu_K) \le \frac{1}{e} - \frac{1}{3}$. For any $\vartheta \in (0,1)$ there exists a polytope P on $t = \mathcal{O}\big(\frac{m^2}{\vartheta(1-\vartheta)^m}\big)$ vertices such that $K \subseteq P \subseteq \frac{K}{\vartheta}$, where the scaling is about z.*

For the proof, we need some ancillary results.

Theorem 3.20 (Bertsimas and Vempala [2004], Theorem 3)**.** *Let K be a convex set in isotropic position and z be a point at distance t from its centroid. Then any halfspace containing z contains at least $\frac{1}{e} - t$ of the volume of K.*

Now we adapt a result by Naszódi [2018], recalled here for convenience. We say that a halfspace F supports a convex body from outside if F intersects the boundary of the body, but not its interior.

Lemma 3.21 (Lemma 2.1, Naszódi [2018])**.** *Let $0 < \vartheta < 1$, and F be a halfspace that supports ϑK from outside, where the scaling is about μ_K. Then:*

$$\mathrm{vol}(K \cap F) \ge \mathrm{vol}(K) \cdot (1-\vartheta)^m \frac{1}{e} \tag{3.14}$$

Our adaptation is:

Lemma 3.22. *Let $z \in \mathbb{R}^m$, let $0 < \vartheta < 1$, and let F be a half-space that supports ϑK from outside, where the scaling is about z. Then:*

$$\mathrm{vol}(K \cap F) \ge \mathrm{vol}(K) \cdot (1-\vartheta)^m \left(\frac{1}{e} - d_K(\mu_K, z)\right) \tag{3.15}$$

Proof. The proof is similar to the proof of the original lemma. See Figure 3.A.1 for reference. Let F_0 be a translate of F whose boundary contains z, and let $K_0 = K \cap F_0$. Let F_1 be a translate of F that supports K from outside, and let $p \in F_1 \cap K$. Now consider $K_0' = \vartheta p + (1-\vartheta)K_0$, that is, the homothetic copy of K_0 with center p and ratio $1-\vartheta$. The crucial observation is that $F = \vartheta p + (1-\vartheta)F_0$, which implies $K_0' \subset K \cap F$. Clearly $\mathrm{vol}(K_0') = (1-\vartheta)^m \,\mathrm{vol}(K_0)$. Moreover, by Theorem 3.20 we have $\mathrm{vol}(K_0) \ge \mathrm{vol}(K)(1/e - t)$ where $t = d_K(\mu_K, z)$; this holds because mapping K to its isotropic position preserves volume ratios. This concludes the proof. □

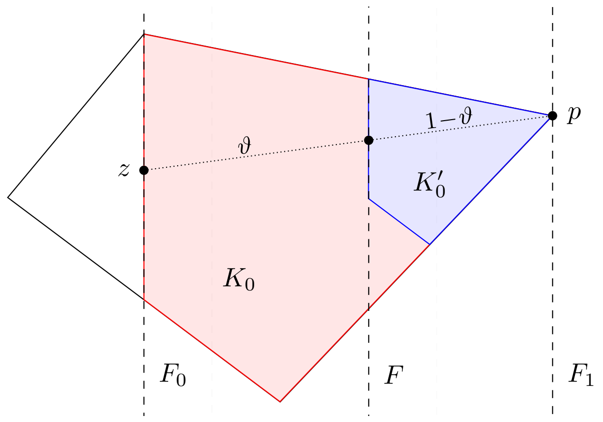


FIGURE 3.A.1: a visual proof of Lemma 3.22, with $d(z,p) = 1$ for simplicity.

Proof of Lemma 3.19. We adapt the construction behind [Naszódi, 2018, Theorem 1.2], by replacing $\varepsilon = \frac{(1-\vartheta)^m}{e}$ with $\varepsilon = \frac{(1-\vartheta)^m}{3}$. The theorem then says that, if we set:

$$t = \left\lceil C \frac{(m+1)3}{(1-\vartheta)^m} \ln \frac{3}{(1-\vartheta)^m} \right\rceil \tag{3.16}$$

and we let $X_1, \dots, X_t$ be t points chosen independently and uniformly at random from K, and let $P = \operatorname{conv}(X_1, \dots, X_t)$, then $\vartheta K \subseteq P \subseteq K$ with probability at least $1 - p$, where

$$p = 4\left(11C^2\left(\frac{(1-\vartheta)^m}{3}\right)^{C-2}\right)^{m+1} \tag{3.17}$$

Now, we choose $C = \Theta(\frac{1}{\vartheta}\ln\frac{1}{\vartheta})$ large enough. On the one hand, we obtain:

$$t = \left\lceil C\frac{(m+1)3}{(1-\vartheta)^m}\ln\frac{3}{(1-\vartheta)^m}\right\rceil = \mathcal{O}\left(\frac{m^2}{\vartheta(1-\vartheta)^m}\ln\frac{1}{\vartheta}\ln\frac{1}{1-\vartheta}\right) \tag{3.18}$$

Since $\vartheta \in (0,1)$, we have $\ln\frac{1}{\vartheta}\ln\frac{1}{1-\vartheta} = \ln(1 + {}^1\!/\!_x)\ln(1+x)$ where $x = \frac{\vartheta}{1-\vartheta} > 0$. However, $\ln(1+{}^1\!/\!_x)\ln(1+x) < 1$ for all $x > 0$. Therefore, $t \in \mathcal{O}\big(\frac{m^2}{\vartheta(1-\vartheta)^m}\big)$. On the other hand, by setting C large enough we can make $C^2\big(\frac{(1-\vartheta)^m}{3}\big)^{C-2}$ arbitrarily small, and therefore $p < 1$.

Since $p < 1$, we conclude that *there exists* a polytope P on $t \in \mathcal{O}\big(\frac{m^2}{\vartheta(1-\vartheta)^m}\big)$ vertices such that $\vartheta K \subseteq P \subseteq K$. To conclude, instead of P simply take $\frac{P}{\vartheta}$ where the scaling is about z. □

Wrap-up

First, by Lemma 3.16, by taking $N \in \mathcal{O}(m^2)$ large enough we can make $d_K(\mu_K, z) \le \frac{1}{e} - \frac{1}{3}$ with probability arbitrarily close to 1. Now let $\mathcal{P}_t$ be the family of all polytopes in $\mathbb{R}^m$ on at most t vertices. For $t \in \mathcal{O}\big(\frac{m^2}{\vartheta(1-\vartheta)^m}\big)$ large enough, Lemma 3.19 implies that there exists some $P \in \mathcal{P}_t$ such that $K \subseteq P \subseteq \frac{K}{\vartheta}$.

Now we prove that, by choosing s large enough, with probability arbitrarily close to 1 we have $|\frac{K}{\vartheta} \cap C| \ge \frac{1}{2}|C|$. First, by a recent result of Kupavskii [2020], we have $\operatorname{vc-dim}(\mathcal{P}_t) \le 8m^2 t\log_2 t$. For our t this yields

$$\operatorname{vc-dim}(\mathcal{P}_t) = \mathcal{O}\left(m^2\frac{m^2}{\vartheta(1-\vartheta)^m}\ln\frac{m^2}{\vartheta(1-\vartheta)^m}\right) = \mathcal{O}\left(\frac{m^5}{\vartheta(1-\vartheta)^m}\ln\frac{1}{\vartheta(1-\vartheta)}\right) \tag{3.19}$$

where in the last equality we used $\ln\frac{m^2}{\vartheta(1-\vartheta)^m} = \ln\frac{m^{(2/m)m}}{\vartheta(1-\vartheta)^m} \le m\ln\frac{m^{2/m}}{\vartheta(1-\vartheta)} = \mathcal{O}\big(m\ln\frac{1}{\vartheta(1-\vartheta)}\big)$. Hence, by choosing $|S_C| = s = \mathcal{O}\big(\frac{m^5}{\vartheta(1-\vartheta)^m}\ln\frac{1}{\vartheta(1-\vartheta)}\big)$ large enough, for any constant c, ϵ, δ we can make:

$$|S_C| \ge c\,\frac{\operatorname{vc-dim}(\mathcal{P}_t)\,\ln\frac{1}{\epsilon} + \ln\frac{1}{\delta}}{\epsilon} \tag{3.20}$$

Since P is consistent with S_C, that is, $P \supset S_C$, then by standard PAC bounds we have $|P \cap C| \ge (1-\epsilon)|C|$ with probability at least $1-\delta$. But $P \subseteq \frac{K}{\vartheta}$, and therefore $|\frac{K}{\vartheta} \cap C| \ge (1-\epsilon)|C|$ with probability at least $1-\delta$. By adjusting the constants this yields the thesis of Lemma 3.4.

3.B Proof of Theorem 3.3

Theorem 3.3. *Let (X, O) be an instance whose clustering $\mathcal{C}$ has convex hull margin $\gamma > 0$. Then* $\textsc{CheatRec}(X, O, \gamma)$ *deterministically outputs $\mathcal{C}$, runs in time* $\operatorname{poly}(k, n, m)$*, and with high probability makes at most $\mathcal{O}\big(k^2 m^5 (1 + {}^1\!/\!_\gamma)^m \log(1 + {}^1\!/\!_\gamma)\log n\big)$ queries.*

We give the pseudocode of the algorithm for reference. First, we prove the correctness and the query bound. Then we show the running time bound. Note that, for readability, the pseudocode given here is high-level; the actual implementation is more complex, see below.

Correctness and query bound. We prove that, at each round, for some i we recover at least half of the points in C_i with probability $1-\delta$, where δ can be made arbitrarily small by adjusting the constants. Let S_i be the subset of the sample S having label i. Since there are at most k clusters and $|S| = ks$, for some i we will have $|S_i| \ge s$. Now we apply Lemma 3.4 to $K = \operatorname{conv}(S_i)$. Since s satisfies the hypotheses, the lemma says that $\hat{C}_i = Q \cap X$ has size $|\hat{C}_i| \ge \frac{|C_i|}{2}$ with probability arbitrarily close to 1 (that is, with probability $1-\delta$ as above). It

Algorithm 3 HullTrick$(K, 1+\gamma)$

```
let $N = \Omega(m^2)$ large enough
let $\epsilon = \mathcal{O}(m^{-1})$ small enough
draw $N$ i.i.d. random points $X_1, \dots, X_N$ from any $\epsilon$-uniform distribution over $K$
let $z = \frac{1}{N}\sum_{i=1}^{N} X_i$
return $z + (1+\gamma)(K - z)$
```

Algorithm 4 CheatRec(X, O, γ)

```
while $X \neq \emptyset$ do
    let $s = \mathcal{O}\big(m^5 (1 + 1/\gamma)^m \ln (1 + 1/\gamma)\big)$ large enough
    draw a uniform random sample $S$ of size $\min(|X|, ks)$ from $X$, without repetition
    learn the labels of $S$ with $ks$ queries to $O$
    let $S_i$ be the points of $S$ having label $i$
    for $i = 1, \dots, k$ do
        $K = \mathrm{conv}(S_i)$
        $Q = $ HullTrick$(K, 1+\gamma)$
        $\hat{C}_i = Q \cap X$
        label all points of $\hat{C}_i$ with label $i$
        $X = X \setminus \hat{C}_i$
```

remains to show that $\hat{C}_i \subseteq C_i$. Let d be any pseudometric that is homogeneous and invariant under translation. Then, any point $y \in (1+\gamma)K \cap X$ satisfies $d(y, K) \leq \gamma\, \phi_d(K)$. But $K = \mathrm{conv}(S_i) \subseteq \mathrm{conv}(C_i)$. Therefore $\phi_d(K) \leq \phi_d(C_i)$. Hence $d(y, \mathrm{conv}(C_i)) \leq \gamma\, \phi_d(C_i)$. By the convex margin assumption, this implies that $y \in C_i$. This also proves the correctness of the algorithm. The total query bound follows as in Lemma 3 of [Bressan et al., 2020a], whose algorithm also recovers an expected fraction $\frac{1}{4k}$ of all points in each round.

Running time bound. First, we analyze the running time of CheatRec excluding the call to HullTrick. Drawing and labeling the samples obviously cost $\mathcal{O}(n)$ time throughout the entire execution. In the main loop, K is actually not computed explicitly — see below. Similarly, Q is simply a set of points obtained by rescaling S_i about some point in space. Thus, computing $\hat{C}_i = Q \cap X$ and labeling its points boils down to deciding, for all $x \in X$, if x can be written as $\sum_{x_j \in Q_j} \lambda_j x_j$ for a set of coefficients $\lambda_1, \dots, \lambda_{|Q|} \in [0, 1]$. This can be done with polynomial precision using any polynomial-time solver for linear programs (say, the ellipsoid method).

Let us now turn to HullTrick. The computationally expensive part is drawing N points from an ϵ-uniform distribution over $K = \mathrm{conv}(S_i)$. This can be done with any method of choice. Here, we consider the "hit-and-run from a corner" algorithm of Lovász and Vempala [2006], which implements a fast mixing random walk whose stationary distribution is uniform over any convex body. We remark that other methods for computing approximate centers exist, see for example [Basu and Oertel, 2017].

The implementation is as follows. First, we put K in near-isotropic position by computing the minimum volume enclosing ellipsoid (MVEE) and then applying an affine transformation to make the MVEE into the ball of unit radius. As shown in [Khachiyan, 1996b], this operation takes time $|S_i| m^2 \big(\ln m + \ln\ln |S_i|\big)$. After this transformation, let μ be the center of the unit ball that contains K. Observe that μ is at distance at least $\frac{1}{m}$ from the boundary of K: this holds since, by John's theorem, the ball of radius $\frac{1}{m}$ centered at μ is entirely contained in K. Now we execute the hit-and-run from a corner algorithm starting at μ. By the results of Lovász and Vempala [2006], we have the following bound.

Lemma 3.23 (See Lovász and Vempala [2006], Corollary 1.2)**.** *Assume hit-and-run is started from μ. For any $\epsilon > 0$, the distribution of the random walk after*

$$t = \Theta\left(m^4 \ln \frac{m}{\epsilon}\right)$$

steps is ϵ-uniform over K.

It remains to implement the hit-and-run algorithm over K. To this end we need to solve the following problem: given a generic point $x \in K$ and a vector $u \in S^{n-1}$, determine the intersection of the ray $\{x+\alpha u\}_{\alpha \geq 0}$ with the boundary of K. This amounts to solving a linear program that searches for the maximum value $\alpha \geq 0$ such that $x + \alpha u$ can be written as $\sum_{x_j \in S_i} \lambda_j x_j$ for a set of coefficients $\lambda_1, \ldots, \lambda_{|S_i|} \in [0,1]$. We can solve such an LP in time $t_K = \text{poly}(|S_i|, m)$ with polynomial precision using any polynomial-time solver for linear programs. The total time to draw the N samples is therefore:

$$\mathcal{O}\left(|S_i| m^2 \big(\ln m + \ln\ln |S_i|\big) + N t_K \, m^4 \ln \frac{m}{\epsilon}\right) \tag{3.21}$$

As we set $N = \mathcal{O}(m^2)$ and $\epsilon = \mathcal{O}(m^{-1})$, the total running time is:

$$\mathcal{O}\left(|S_i| m^2 \big(\ln m + \ln\ln |S_i|\big) + t_K \, m^6 \ln m\right) \tag{3.22}$$

which is in $\text{poly}(|S_i|, m) = \text{poly}(n, m)$.

3.C Proof of Lemma 3.6

Lemma 3.6. *For any $u \in \mathbb{R}^2$ let $d_u(x,y) = |\langle u, x-y\rangle|$. For any $\eta > 0$ there exists a clustering $\mathcal{C} = (C_1, C_2)$ on a set $X \subset \mathbb{R}^2$ that has arbitrarily large one-versus-all margin with respect to $d_{(0,1)}, d_{(1,0)}$, and yet $d_u(C_1, C_2) \leq \eta \, \phi_{d_u}(X)$ for any $u \in \mathbb{R}^2 \setminus \mathbf{0}$.*

Let $u_1 = (1,0)$ and $u_2 = (0,1)$, and for some constant a independent of η and to be fixed later, consider the set X consisting of the four points (see Figure 3.4.1):

$$p_1 = \eta \, u_1, \quad q_1 = a \, u_1 \tag{3.23}$$

$$p_2 = \eta \, u_2, \quad q_2 = a \, u_2 \tag{3.24}$$

Finally, let $\mathcal{C} = (C_1, C_2)$ where $C_1 = \{p_1, q_1\}$ and $C_2 = \{p_2, q_2\}$.

Consider the two pseudometrics $d_1 = d_{(0,1)}$ and $d_2 = d_{(1,0)}$. Then $\phi_{d_1}(C_1) = 0$ and $d_1(C_1, C_2) = \eta$, and vice versa, $\phi_{d_2}(C_2) = 0$ and $d_2(C_1, C_2) = \eta$. Thus, the one-versus-all margin of $\mathcal{C}$ with respect to d_1, d_2 is unbounded.

Now choose any $u \in \mathbb{R}^2 \setminus \mathbf{0}$. Without loss of generality, by rescaling we can assume u to be a unit vector. In this case, we have $d_u(C_1, C_2) \leq d_u(p_1, p_2) \leq \|p_1 - p_2\|_2 = \eta\sqrt{2}$. Yet, the convex hull of X contains a ball of radius $\Omega(1)$, and therefore $\phi_d(X) = \Omega(1)$, where the constants depend on a. Hence, we can make $d(C_1, C_2) \leq \eta \, \phi_d(X)$ by choosing a appropriately.

3.D Proof of Theorem 3.8

Theorem 3.8. *If $\mathcal{C}$ satisfies α-center proximity, then it has one-versus-all margin at least $\frac{(\alpha-1)^2}{2(\alpha+1)}$. Hence, if $\mathcal{C}$ satisfies $(1+\epsilon)$-perturbation stability, then it has one-versus-all margin at least $\frac{\epsilon^2}{2(\epsilon+2)}$.*

Consider any cluster C_i with center c_i. We must show that any $y \in C_j$ with $j \neq i$ satisfies $d(y,x) > \frac{(\alpha-1)^2}{2(\alpha+1)} \phi_d(C_i)$ for all $x \in C_i$. Let $x' = \arg\max_{x \in C_i} d(x, c_i)$. Clearly, if $d(x', c_i) = 0$ then all points of C_i coincide and $\phi_d(C_i) = 0$. If this is the case, then by the α-center proximity, for any $x \in C_i$ and any $y \in C_j$ we have $d(y,x) = d(y, c_i) > \alpha \, d(y, c_j) \geq 0$. Therefore $d(y,x) > a \, \phi_d(C_i)$ for any $a > 0$, which in particular proves our thesis.

Suppose instead that $d(x', c_i) > 0$. Choose any $x \in C_i$ and any $y \in C_j$. Now, we have two cases.

Case 1: $x = c_i$. In this case we proceed as follows. Bear in mind that $d(x', c_i) = \phi_d(C_i)$, $d(y, c_i) = d(y, x)$, and $d(y, c_j) < \frac{1}{\alpha} d(y, c_i)$.

$$\phi_d(C_i) = d(x', c_i) \tag{3.25}$$

$$< \frac{1}{\alpha} d(x', c_j) \tag{3.26}$$

$$\leq \frac{1}{\alpha}\Big(d(x', c_i) + d(c_i, y) + d(y, c_j)\Big) \tag{3.27}$$

$$< \frac{1}{\alpha}\left(\phi_d(C_i) + d(y, x) + \frac{1}{\alpha}d(y, c_i)\right) \tag{3.28}$$

$$= \frac{1}{\alpha}\phi_d(C_i) + \frac{1}{\alpha}d(y, x) + \frac{1}{\alpha^2}d(y, x) \tag{3.29}$$

from which we infer

$$d(y, x) > \frac{\alpha(\alpha-1)}{\alpha+1}\phi_d(C_i) > \frac{(\alpha-1)^2}{2(\alpha+1)}\phi_d(C_i) \tag{3.30}$$

Case 2: $x \neq c_i$**.** In this case, $d(x, c_i) > 0$, and we start by deriving:

$$d(y, x) \geq d(x, c_j) - d(y, c_j) \tag{3.31}$$

$$> d(x, c_j) - \frac{1}{\alpha}d(y, c_i) \tag{3.32}$$

$$\geq d(x, c_j) - \frac{1}{\alpha}\big(d(y, x) + d(x, c_i)\big) \tag{3.33}$$

$$= -\frac{1}{\alpha}d(y, x) + d(x, c_j) - \frac{1}{\alpha}d(x, c_i) \tag{3.34}$$

and thus

$$d(y, x)\left(\frac{\alpha+1}{\alpha}\right) > d(x, c_j) - \frac{1}{\alpha}d(x, c_i) \tag{3.35}$$

which yields

$$d(y, x) > \frac{\alpha}{\alpha+1}d(x, c_j) - \frac{1}{\alpha+1}d(x, c_i) \tag{3.36}$$

Let $\beta = \frac{d(x', c_i)}{d(x, c_i)}$. Observe that $\phi_d(C_i) \leq 2\beta d(x, c_i)$ and therefore $d(x, c_i) \geq \frac{\phi_d(C_i)}{2\beta}$.

Now we consider two cases. First, suppose that $\beta \leq \frac{\alpha+1}{\alpha-1}$. In this case, we apply $d(x, c_j) > \alpha d(x, c_i)$ to (3.36) to obtain:

$$d(y, x) > \frac{\alpha^2}{\alpha+1}d(x, c_i) - \frac{1}{\alpha+1}d(x, c_i) \tag{3.37}$$

$$= d(x, c_i)(\alpha - 1) \tag{3.38}$$

$$\geq \frac{\phi_d(C_i)}{2\beta}(\alpha - 1) \tag{3.39}$$

$$\geq \phi_d(C_i)\frac{(\alpha-1)^2}{2(\alpha+1)} \tag{3.40}$$

Suppose instead that $\beta > \frac{\alpha+1}{\alpha-1}$. Since we chose $x' \in C_i$ such that $d(x', c_i) = \beta d(x, c_i)$, we obtain:

$$d(x, c_j) > d(x', c_j) - d(x, x') \tag{3.41}$$

$$> \alpha d(x', c_i) - \big(d(x, c_i) + d(x', c_i)\big) \tag{3.42}$$

$$= \alpha\beta d(x, c_i) - \big(d(x, c_i) + \beta d(x, c_i)\big) \tag{3.43}$$

$$= d(x, c_i)((\alpha - 1)\beta - 1) \tag{3.44}$$

Combining this with (3.36), we obtain:

$$d(y, x) > d(x, c_i)\left(\frac{\alpha}{\alpha+1}((\alpha-1)\beta - 1) - \frac{1}{\alpha+1}\right) \tag{3.45}$$

$$= d(x, c_i)\left(\frac{\alpha(\alpha-1)\beta}{\alpha+1} - 1\right) \tag{3.46}$$

$$\geq \frac{\phi_d(C_i)}{2\beta}\left(\frac{\alpha(\alpha-1)\beta}{\alpha+1}-1\right) \tag{3.47}$$
$$= \phi_d(C_i)\left(\frac{\alpha(\alpha-1)}{2(\alpha+1)}-\frac{1}{2\beta}\right) \tag{3.48}$$
$$> \phi_d(C_i)\left(\frac{\alpha(\alpha-1)-(\alpha-1)}{2(\alpha+1)}\right) \tag{3.49}$$
$$= \phi_d(C_i)\frac{(\alpha-1)^2}{2(\alpha+1)} \tag{3.50}$$

Hence, in all cases, we obtain $d(y,x) > \phi_d(C_i)\frac{(\alpha-1)^2}{2(\alpha+1)}$. This concludes the proof.

3.E Proof of Lemma 3.9

Lemma 3.9 (One-versus-all margin implies one-sided-error learnability)**.** *Let d be any pseudometric over $\mathcal{X}$. For any finite $X \subset \mathcal{X}$ and any $\gamma > 0$, define the effective concept class over X:*

$$H = \{C \subseteq X \,:\, d(X \setminus C, C) > \gamma\,\phi_d(C)\} \tag{3.3}$$

Then $H = I(H)$, and $\text{vc-dim}(H, X) \leq M$ where $M = \max(2, M(\gamma, d))$. As a consequence, H can be learned with one-sided error ϵ and confidence δ with $\mathcal{O}\big(\epsilon^{-2}(M \log {}^1/_\epsilon + {}^1/_\delta)\big)$ examples by choosing the smallest consistent hypothesis in H.

For the first claim, we start by showing that $H = I(H)$. Let $C_1, C_2 \in H$. We show that $C := C_1 \cap C_2 \in H$, too. Consider any $y \in X \setminus C$, and without loss of generality assume that $y \notin C_1$. Since $C \subseteq C_1$, we have $\phi_d(C_1) \geq \phi_d(C)$ and $d(y, C) \geq d(y, C_1)$. Moreover, $d(y, C_1) > \gamma\,\phi_d(C_1)$ since $C_1 \in H$. Therefore:

$$d(y, C) \geq d(y, C_1) > \gamma\,\phi_d(C_1) \geq \gamma\,\phi_d(C) \tag{3.51}$$

proving that $d(y, C) > \gamma\,\phi_d(C)$. Since this holds for all $y \in X \setminus C$, we have $C \in H$ as well. Therefore, $H = I(H)$.

Since $H = I(H)$, then $\text{vc-dim}(H, X) = \text{vc-dim}(I(H), X)$. Now, we use the following results of Kivinen [1995]:

Definition 3.24. *[Kivinen [1995], Definition 5.11] Let $\mathcal{X}$ be any domain and $\mathcal{H} \subseteq 2^{\mathcal{X}}$ be a concept class. We say that $\mathcal{H}$ slices $X \subset \mathcal{X}$ if, for each $x \in X$, there is $h \in \mathcal{H}$ such that $X \cap h = X \setminus \{x\}$. The slicing dimension of $(\mathcal{H}, \mathcal{X})$, denoted by $\text{sl}(\mathcal{H}, \mathcal{X})$, is the maximum size of a set sliced by $\mathcal{H}$. If $\mathcal{H}$ slices arbitrarily large sets, then we let $\text{sl}(\mathcal{H}, \mathcal{X}) = \infty$.*

Lemma 3.25 (Kivinen [1995], Lemma 5.19)**.** *If $\text{sl}(\mathcal{H}, \mathcal{X}) < \infty$, then $\text{vc-dim}(I(\mathcal{H}), \mathcal{X}) \leq \text{sl}(\mathcal{H}, \mathcal{X})$.*

We will now show that $\text{sl}(H, X) \leq M$, where M is finite by assumption. By Lemma 3.25, this will imply $\text{vc-dim}(H, X) \leq \text{sl}(H, X)$.

Let $S \subseteq X$ be any set of size $\text{sl}(H, X)$ that is sliced by H. To show that $\text{sl}(H, X) \leq M$, we need to show that $|S| \leq M$. If $|S| \leq 2$, then this is trivial, since $M \geq 2$. Suppose then that $|S| \geq 3$. Choose $a, b \in S$ such that $d(a, b) = \phi_d(S)$. Since S is sliced by H, for any $x \in S$ we must have $S \setminus x = S \cap C$ for some $C \in H$. Since $S \setminus x \subseteq C$, we have $d\big(S \setminus x, x\big) \geq d(C, x)$ and $\phi_d(C) \geq \phi_d(S \setminus x)$. Moreover, $d(C, x) > \gamma\,\phi_d(C)$, since $x \in X \setminus C$ and $C \in H$. Therefore:

$$d\big(S \setminus x, x\big) \geq d(C, x) \tag{3.52}$$
$$> \gamma\,\phi_d(C) \tag{3.53}$$
$$\geq \gamma\,\phi_d(S \setminus x) \tag{3.54}$$

Hence, $d(S \setminus x, x) > \gamma\,\phi_d(S \setminus x)$ for all $x \in S$.

Now, suppose first that $\gamma \geq 1$. Since $|S| \geq 3$, any $x \in S \setminus \{a, b\}$ yields the absurd:

$$\phi_d(S) \geq d(S \setminus x, x) \tag{3.55}$$

$$\begin{aligned} &> \phi_d(S \setminus x) && \text{since } \gamma \geq 1 && (3.56)\\ &= d(a,b) && \text{since } a,b \in S \setminus x && (3.57)\\ &= \phi_d(S) && \text{by the choice of } a,b && (3.58)\end{aligned}$$

Hence we must have $|S| \leq 2$, which implies again $|S| \leq M$. Suppose instead that $\gamma < 1$. Then, for any two distinct points $x, y \in S$, we have $d(x,y) > \gamma\phi_d(S)$. This is trivially true if $x = a$ and $y = b$; otherwise, assuming $x \notin \{a,b\}$, it follows by the fact that $d(x,y) \geq d(S \setminus x, x) > \gamma\phi_d(S \setminus x) = \gamma\phi_d(S)$, as seen above. Moreover, S is contained in the closed ball $B(x, \phi_d(S))$ for any $x \in S$. Therefore, $\mathcal{M}(B(x,r), \gamma r, d) \geq |S|$ for $r = \phi_d(S)$ and some $x \in \mathcal{X}$. By definition of $M(\gamma, d)$ this implies that $|S| \leq M(\gamma, d)$, and $M(\gamma,d) \leq M$. This concludes the proof.

3.F Proof of Theorem 3.10

Theorem 3.10. *Let (X, O) be any instance whose latent clustering $\mathcal{C}$ has one-versus-all margin $\gamma > 0$ with respect to $d_1, \dots, d_k$. Then* mRec(X, O, γ) *deterministically outputs $\mathcal{C}$ while making, with high probability, at most $\mathcal{O}(Mk \log k \log n)$ queries to O, where $M =$* $\max(2, M(\gamma))$. *Moreover, for any algorithm $\mathcal{A}$ and for any $\gamma > 0$, there are instances with one-versus-all margin γ on which $\mathcal{A}$ makes $\Omega(M(2\gamma))$ queries in expectation.*

For the lower bounds, let $\gamma' = 2\gamma$. By definition of $M(\gamma')$, there exists a set X of $M(\gamma')$ points that, according to some $d \in \{d_1, \dots, d_k\}$, lies within a ball of radius $r > 0$, and thus has diameter at most $2r$, and such that $d(x,y) > \gamma' r = 2\gamma r$ for all distinct $x, y \in X$. Now choose $x \in X$ uniformly at random, and define $\mathcal{C} = (x, X \setminus x)$. The argument above shows that $d(x, X \setminus x) > \gamma\phi_d(X)$, which implies that $\mathcal{C}$ has one-versus-all margin γ. Clearly, in expectation over the distribution of $\mathcal{C}$, any exact cluster recovery algorithm must make $\Omega(|X|) = \Omega(M(2\gamma))$ queries. By Yao's principle for Monte Carlo algorithms, then, any such algorithm makes $\Omega(M(2\gamma))$ queries on some instance.

Let us turn to the upper bounds. The pseudocode of mRec is given below. As in the proof sketch, for each $i \in [k]$ the class H_i is defined as:

$$H_i = \{C \subseteq X \,:\, d_i(X \setminus C, C) > \gamma\, \phi_{d_i}(C)\} \tag{3.59}$$

As said in the proof sketch, by Lemma 3.9 and by the definition of learning with one-sided error (Definition 3.1), we have $\hat{C}_i \subseteq C_i$ and therefore mRec never misclassifies any point, and moreover we have:

$$\mathbb{P}\left(\left|C_i \setminus \hat{C}_i\right| \leq \epsilon|X|\right) \geq 1 - \delta \tag{3.60}$$

In our case, that is, with $\epsilon = 1/2k$ and $\delta = 1/2$, and since $|C_i \setminus \hat{C}_i| = |C_i| - |\hat{C}_i|$, this yields:

$$\mathbb{P}\left(\left|\hat{C}_i\right| \geq |C_i| - \frac{|X|}{2k}\right) \geq \frac{1}{2} \tag{3.61}$$

which, since $\left|\hat{C}_i\right| \geq 0$, implies:

$$\mathbb{E}\left|\hat{C}_i\right| \geq \frac{1}{2} \cdot \left(|C_i| - \frac{|X|}{2k}\right) = \frac{|C_i|}{2} - \frac{|X|}{4k} \tag{3.62}$$

By summing over all $i \in [k]$, at each round mRec correctly labels, and removes from X, an expected number of points equal to:

$$\mathbb{E}\left|\hat{C}_1 \cup \ldots \cup \hat{C}_k\right| = \sum_{i=1}^{k} \mathbb{E}\left|\hat{C}_i\right| \geq \sum_{i=1}^{k} \left(\frac{|C_i|}{2} - \frac{|X|}{4k}\right) = \frac{|X|}{2} - \frac{|X|}{4} = \frac{|X|}{4} \tag{3.63}$$

Thus, at each round mRec gets rid of an expected fraction $1/4$ of all points still in X. By a standard probabilistic argument this implies that, with high probability, all points are correctly labeled within $\mathcal{O}(\log n)$ rounds, see Lemma 3 of [Bressan et al., 2020a]. Therefore, the total number of queries is bounded by $\mathcal{O}(Mk \log k)$ with high probability. This concludes the proof.

Algorithm 5 $\textsc{mRec}(X, O)$

```
if $X = \emptyset$ then return
for each $i \in [k]$ let $H_i$ as in Equation 3.59
draw a sample $S$ of $|S| = \Theta\big(Mk \ln k\big)$ points u.a.r. from $X$
use $O$ to learn the labels of $S$
for $i \in [k]$ do
    let $S_i$ be the subset of $S$ having label $i$
    let $\hat{C}_i$ be the smallest set in $H_i$ consistent with $S_i$
    give label $i$ to every $x \in \hat{C}_i$
let $X' = X \setminus \cup_{i\in[k]} \hat{C}_i$
$\textsc{mRec}(X', O)$
```

3.G Proof of Theorem 3.12

As said in the sketch, the proof follows the same ideas of the proof of Theorem 3.10.

For the lower bounds, suppose that $\mathrm{cosl}(\mathcal{H}) < \infty$, and let X be a set cosliced by $\mathcal{H}$ with $|X| = \mathrm{cosl}(\mathcal{H})$. We draw a random uniform element $x \in X$, and we consider the clustering $\mathcal{C} = (x, X \setminus x)$. By definition of sliced set, $\mathcal{C}$ is realised by $\mathcal{H}$, and therefore it satisfies the assumptions. Now the same arguments of the lower bounds of Theorem 3.10 imply that any algorithm needs $\Omega(|X|) = \Omega(\mathrm{cosl}(\mathcal{H}))$ queries on some instance to return $\mathcal{C}$. Clearly, if $\mathrm{cosl}(\mathcal{H}) = \infty$ this means that we can take $|X| = n$ arbitrarily large, whence the second lower bound.

For the upper bounds, let $P_k(X)$ be the set of all k-clusterings of X that are realised by $\mathcal{H}$. Then, for each $i \in [k]$ we define:

$$H_i = \{C' \,:\, C' = C'_i \,\wedge\, (C'_1, \ldots, C'_k) \in P_k(X)\} \tag{3.64}$$

As in $\textsc{mRec}$, we learn each class H_i with one-sided error by choosing the smallest hypothesis in $I(H_i)$ that is consistent with S_i, for a labeled sample S of size $\Theta(\text{vc-dim}(I(H_i), X)\, k \ln k)$. As shown in the proof of Lemma 3.9, a result of Kivinen [1995] implies that if $\mathrm{sl}(H_i, X) < \infty$ then $\text{vc-dim}(I(H_i), X) \le \mathrm{sl}(H_i, X)$. Therefore, to prove the theorem we only need to show that $\mathrm{sl}(H_i, X) \le \mathrm{cosl}(\mathcal{H})$. To this end, suppose that $U = \{x_1, \ldots, x_\ell\} \subseteq X$ is sliced by H_i. By construction of H_i, this means that there are ℓ clusterings $\mathcal{C}_1, \ldots, \mathcal{C}_\ell$, each one realised by $\mathcal{H}$, such that $\mathcal{C}_i = (\{x_i\}, U \setminus \{x_i\})$ for all $i \in [k]$. This implies that U is cosliced by $\mathcal{H}$. Hence, $|U| \le \mathrm{cosl}(\mathcal{H})$ and so $\mathrm{sl}(H_i, X) \le \mathrm{cosl}(\mathcal{H})$, as claimed. The rest of the proof is similar to the proof of Theorem 3.10, and shows that at each round we recover an expected constant fraction of all points, and that therefore all points will be recovered with high probability after $\mathcal{O}(\log n)$ rounds. This shows that we can recover $\mathcal{C}$ by making with high probability at most $\mathcal{O}(\mathrm{cosl}(\mathcal{H}) k \log k \log n)$ queries.

3.H Proof of Theorem 3.14

Theorem 3.14. *Let $\mathcal{H}$ be a concept class in $\mathbb{R}^m$ that is non-fractal and closed under affine transformations. There is an algorithm that, given any instance whose latent clustering $\mathcal{C}$ has one-versus-all margin γ and is realized by $\mathcal{H}$, returns $\mathcal{C}$ while making $\mathcal{O}(Mk \log k \log n)$ queries with high probability, where $M = \max\big(2, (1 + {}^{4}\!/\!_{\gamma})^m\big)$. Moreover, for any algorithm $\mathcal{A}$, there exist arbitrarily large n-point instances, whose latent clustering $\mathcal{C}$ has arbitrarily small one-versus-all margin and is realized by $\mathcal{H}$, where $\mathcal{A}$ makes $\Omega(n)$ queries in expectation to recover $\mathcal{C}$.*

The upper bound follows from Theorem 3.10, by using the standard fact that in the Euclidean metric the unit ball has covering number $\mathcal{N}(B(x, 1), \epsilon) \le (1 + {}^{2}\!/\!_{\epsilon})^m$ for all $\epsilon > 0$.

As it is well-known, we have $\mathcal{M}(B(x,1),\epsilon) \leq \mathcal{N}(B(x,1),\epsilon/2)$, which for $\epsilon = \gamma$ yields $M(\gamma) \leq (1+4/\gamma)^m$.

We now prove the lower bound. Having instances with "arbitrarily small one-versus-all margin" means that $\gamma = 0$, see Definition 3.5. Take any $h \in \mathcal{H}$ such that both h and its complement $\bar{h}$ contain a ball of positive radius. Note that this implies that, for any $\rho > 0$, there exists a closed ball B with radius $r > 0$ such that:

$$B \subseteq h \tag{3.65}$$

$$\exists x \in \bar{h} \,:\, d(B,x) \leq \rho \tag{3.66}$$

Let c be the center of B. Consider a sphere S of radius r' that contains x and whose center c' lies on the affine subspace $x + \alpha(c-x)$. Let $\eta = \sup_{y \in S \setminus B} d(x,y)$ and let X be an η-packing of S. Note that, since $\gamma = 0$, we can choose ρ and r' arbitrarily small, and in particular we can make the ratio $\frac{\eta}{r'}$ arbitrarily small. This implies that we can make X arbitrarily large, see Figure 3.H.1.

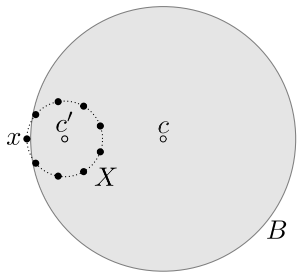


FIGURE 3.H.1: An η-packing X such that $X \cap h = X \setminus \{x\}$ and $X \cap \bar{h} = \{x\}$. The ball B is by construction entirely in h, whereas x is by construction in $\bar{h}$.

Now, consider $x' \in X \setminus \{x\}$. As by construction $d(x',x) > \eta$, and as $r' < r$, we must have $x' \in B$. Therefore, $x' \in h$. Hence the concept $h_x = h$ is such that $X \cap h_x = X \setminus \{x\}$. Now, for any $x' \in X \setminus \{x\}$, there is a rotation R with fixed point c' and such that $R(x') = x$. Hence, $h_{x'} = R^{-1}h_x$ is such that $X \cap h_{x'} = X \setminus \{x'\}$. Since R^{-1} is an affine transformation, $h_{x'} \in \mathcal{H}$ as well. Hence, for every $x \in X$ there exists some concept $h_x \in \mathcal{H}$ such that $X \cap h_x = X \setminus \{x\}$.

Now consider the complement $\bar{h}$ of h. Note that $\bar{h} \in \operatorname{co}(\mathcal{H})$. The first part of the argument above can be applied to $\bar{h}$ as well, showing that for every $x \in X$ there exists $\bar{h}_x \in \operatorname{co}(\mathcal{H})$ such that $X \cap \bar{h}_x = X \setminus \{x\}$. Now consider the complement h_x of $\bar{h}_x$. Clearly $h_x \in \mathcal{H}$, and moreover, $X \cap h_x = \{x\}$.

Hence, for any $x \in X$ there are two concepts $h_x^-, h_x^+ \in \mathcal{H}$ such that $X \cap h_x^- = X \setminus \{x\}$ and $X \cap h_x^+ = \{x\}$. Hence, every 2-clustering $\mathcal{C}$ of X in the form $C_1 = \{x\}, C_2 = X \setminus \{x\}$ is realized by $\mathcal{H}$. Note that this holds with X fixed; we just need to transform the concetps appropriately. It is they immediate to see that any algorithm must perform $\Omega(|X|)$ queries on some instance (X,O). As we can make X arbitrarily large, this completes the proof.

Chapter 4

Exact recovery of Density-Based Clusterings

The scope of this chapter is to analyze the exact cluster recovery problem in the context on non-convex density-based clustering. We begin by introducing a structural characterization of clusters, called (β, γ)-convexity, that can be applied to any finite set of points equipped with a semi-metric. Using this convexity notion, we design a deterministic algorithm that recovers (β, γ)-convex clusters using $\mathcal{O}(\log n)$ same-cluster queries. We then show that the algorithm has an optimal dependence on some of the problem parameters by proving lower bounds. Finally we show that by using an additional and more powerful type of query, it is possible to recover also clusters of different (and unknown) scales.

4.1 Introduction

We investigate the problem of exact reconstruction of clusters using oracle queries in the semi-supervised active clustering framework (SSAC) of Ashtiani et al. [2016]. In SSAC, we are given n points in a metric space, and the goal is to partition these points into k clusters with the help of an oracle answering queries of the form "do x and y belong to the same cluster?". When the metric is Euclidean, Ashtiani et al. [2016], Bressan et al. [2020b] and Bressan et al. [2021b] show that exact reconstruction is possible using only $\mathcal{O}(\log n)$ oracle queries, which mirrors the query complexity of efficient active learning. These results heavily rely on the Euclidean geometry of the clusters; in particular, clusters are assumed to be convex (e.g., ellipsoidal) and separable with a margin. These assumptions exclude many natural definitions of "cluster", such as those based on notions of density, or those computed by popular techniques like spectral clustering, linkage clustering, or DBSCAN.

In this chapter we study exact cluster recovery in metric spaces, or —more generally— finite semimetric spaces (where the triangle inequality is not necessarily satisfied). The use of semimetrics in clustering, and in other machine learning tasks, is motivated by the fact that in many applications domains the notion of distance is strongly non-Euclidean [Gottlieb et al., 2017]. Classic examples include the Wasserstein distance in vision, the Levenshtein distance in string matching, the cosine dissimilarity in document analysis, the Pearson dissimilarity in bioinformatics. In all these cases, the notions of convexity and separability with margin are lost, or exist only in certain generalized forms, so the cluster recovery techniques of Ashtiani et al. [2016]; Bressan et al. [2020b, 2021b] do not apply anymore. To fill this gap, we introduce a novel notion of cluster convexity that can be applied to any finite semimetric space and that can be exploited algorithmically.

We start by considering *geodesic convexity* in weighted graphs [Pelayo, 2013], a well-known generalization of Euclidean convexity that has been used, among others, for node classification in graphs Thiessen and Gärtner [2020]. Given a weighted graph $\mathcal{G}$, a subset $C \subseteq V(\mathcal{G})$ is said to be geodesically convex if every shortest path between any two vertices of C lies entirely in C. Thus, in a finite semimetric space, we could say that C is a convex cluster if it is geodesically convex in the weighted graph $\mathcal{G}$ encoding the semimetric (with the distances as weights). Unfortunately, this condition is too lax. To see this, take n distinct points on a circle with the Euclidean metric. In $\mathcal{G}$, the shortest path between any two points x, y is always the edge (x, y) itself. Thus, according to this definition, any subset of the n points will be geodesically convex, and so every clustering will be admitted, which means that $\Omega(n)$

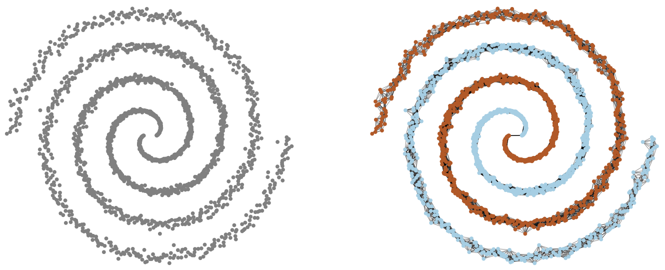


FIGURE 4.1.1: Left: a toy point set. Right: the graph $G_X(\epsilon)$ and a valid (β, γ)-convex clustering for X (clusters encoded by the color of the points), for any $\beta < \frac{1}{2}$ and $\gamma > 0$.

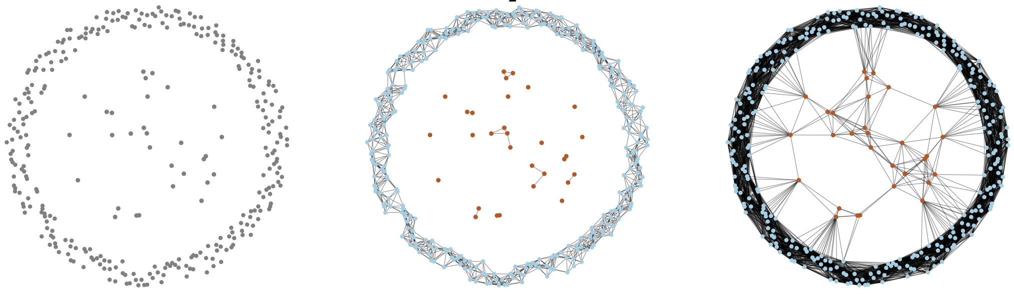


FIGURE 4.1.2: Left: a toy point set. Middle and right: the graphs $G_X(\epsilon_1)$ and $G_X(\epsilon_2)$ where $\epsilon_1 < \epsilon_2$ are the radii of the "outer" cluster C_1 and the "inner" cluster C_2. The clustering is (β, γ)-convex for $\beta \leq .5$ and $\gamma \leq .1$.

queries will be needed to recover the clustering. However, we will show that a variant of this approach gives a suitable notion of convexity, one that yields efficient recovery with only $\mathcal{O}(\log n)$ queries while capturing the density of the clusters.

4.1.1 Contributions.

We introduce (β, γ)-*convex clusterings*, a novel family of clusterings defined on the weighted graph $\mathcal{G}$ encoding the semimetric on X. For any $\epsilon > 0$, let $G_X(\epsilon)$ be the unweighted subgraph of $\mathcal{G}$ obtained by deleting all edges (x, y) with $d(x, y) > \epsilon$. We say that a clustering is (β, γ)-convex (with $\beta, \gamma \in (0, 1]$) if for some $\epsilon > 0$ every cluster C satisfies the following three properties. *Connectivity*: the subgraph of $G_X(\epsilon)$ induced by C is connected. *Local metric margin*: if $x \in C$ and $y \notin C$, then $d(x, y) > \beta\epsilon$[1]. *Geodesic convexity with margin*: in $G_X(\epsilon)$, if $x, y \in C$ and the shortest path between x and y has length ℓ, then any simple path between x and y of length at most $\ell(1+\gamma)$ does not leave C. The smallest ϵ for which these properties hold is called the *radius* of the clusters. It is important to observe that (β, γ)-convexity includes nontrivial cases. For instance, (β, γ)-convex clusters can be strongly non-convex in $\mathbb{R}^d$, as depicted in Figure 4.1.1. Moreover, we can allow the clusters to have different radii $\epsilon_1, \ldots, \epsilon_k$ (see below), as depicted in Figure 4.1.2. These examples suggest that, in a certain sense, one can view (β, γ)-convexity as a way of translating density into convexity. Moreover, if we drop any one of the three conditions —connectedness, local metric margin, and geodesic convexity with margin— the clusters can become disconnected, too close to one another, or interspersed with other clusters, see Section 4.3.1.

Our first result shows that (β, γ)-convex clusterings can be recovered efficiently using a small number of same-cluster queries (SCQ for short). More precisely, if ϵ, β, γ are known, and for each cluster we know an arbitrary initial point, called *seed*,[2] then we can deterministically recover all clusters with $\mathcal{O}\big(k^2 \log n + k^2 \left(6/\beta\gamma\right)^{\text{dens}(X)}\big)$ SCQ queries. Here, $\text{dens}(X)$ is the *density dimension* of X [Gottlieb and Krauthgamer, 2013], a generalization of the doubling

[1] Note that, for any clustering C in a finite semimetric space X and for any $\epsilon > 0$, a β such that the local metric margin condition holds can always be found. In this respect, β defines a hierarchy over clusterings, where large values of β identify clusterings that are easier to learn.

[2] We note here that this last assumptions can be dropped if the clusters have roughly similar sizes. In fact, if the gap between the size of the largest and smallest clusters is χ we can obtain the seed w.h.p. by sampling $\tilde{\mathcal{O}}(k\chi)$ nodes and retrieving their cluster membership using $\tilde{\mathcal{O}}(k^2\chi)$ same cluster queries

dimension that, in metric spaces, is used to bound the size of packings. This dependence of our exponent on dens(X) is asymptotically optimal, as we prove that, in the worst case, any algorithm needs $\Omega(2^{\mathrm{dens}(X)})$ SCQ queries to recover a (β,γ)-convex clustering. The running time of our algorithm is $\mathcal{O}\big(k^2(n+m)\big)$, where m is the number of edges of $\mathcal{G}$ (i.e., the number of finite distances between the input points). The key step of our algorithm consists in finding a *cluster separator* between each cluster C and the other clusters. To do this, we need to find edges between C and other clusters in the graph $G_X(\epsilon)$, which requires to carefully exploit the structural properties of (β,γ)-convexity. We note that, interestingly, in the r-dimensional Euclidean setting, the cluster recovery algorithm of [Bressan et al., 2021b] makes a number of queries roughly of the order of $\mathcal{O}\big((1/\gamma)^r \log n\big)$, where γ is the convex margin. This shows that our notion of convexity plays a role similar to that of Euclidean convexity.

Next, we investigate the power of queries. First, we show that, without seed nodes, any algorithm using only the SCQ oracle needs $\Omega(n)$ queries to recover the clusters. To circumvent this lower bound, we add a more powerful query, called SEED. Given a partition of X and the id of a cluster, the SEED query provides a certificate (i.e., a point of X) that the cluster is cut by the partition, or answers negatively if the cluster is not cut. We show that, if we can use $\mathcal{O}\big(k^2 \log n + k^2 \left(6/\beta\gamma\right)^{\mathrm{dens}(X)}\big)$ SCQ queries plus only $\mathcal{O}(k^2)$ SEED queries, then we can recover clusters with *different* radii $\epsilon_1,\dots,\epsilon_k$, a case that we model by generalizing the (β,γ)-convex definition in a natural way. This allows us to capture clusters with different "scales", as shown in Figure 4.1.2. If the radii $\epsilon_1,\dots,\epsilon_k$ are unknown, we show that $\mathcal{O}(k\log n)$ SEED queries are sufficient to learn them in time $\mathcal{O}\big(m\alpha(m,n)+kn\log n\big)$, where α is the inverse of the Ackermann function, and that no algorithm can learn the radii with less than $\Omega(k\log\frac{n}{k})$ queries. Furthermore, if one of β and γ is unknown, we show that we can still learn the clusters by paying a small overhead.

4.2 Related work

Exact reconstruction of clusters with same-cluster queries was introduced by Ashtiani et al. [2016], who showed how to recover exactly the optimal k-means clustering with $\mathcal{O}(k^2\log n)$ queries when each cluster lies inside a sphere centered in the cluster's centroid and well separated from the spheres of other clusters. Bressan et al. [2020b] extend these results to clusters separated by arbitrary ellipsoids with arbitrary centers, and Bressan et al. [2021b] to arbitrary convex clusters with margin. Both results assume the standard Euclidean metric.

SEED queries have been used by Hanneke [2009] as "positive example queries", by Balcan and Hanneke [2012] as "conditional class queries", by Beygelzimer et al. [2016]; Attenberg and Provost [2010] as "search queries", and, implicitly, also by Tong and Chang [2001]; Doyle et al. [2011]. Previous works also show that SEED queries are well justified in practice, as noted by Beygelzimer et al. [2016].

As with $O(k)$ SCQ queries one can learn the cluster id of any point (up to a relabeling of the clusters), we could reduce our problem to the problem of classifying the nodes of $G_X(\epsilon)$. Dasarathy et al. [2015] develop a probabilistic active classification algorithm, called S^2 ($s$hortest-$s$hortest-path), whose label complexity depends on the graph's structure. In particular, the label complexity is linear in the size of the boundary of the cut between nodes with different labels. Unfortunately, even under (β,γ)-convexity, in $G_X(\epsilon)$ this boundary can have size $\Omega(n)$. Thiessen and Gärtner [2020] show a deterministic algorithm with label complexity proportional to the size of the shortest path cover of the graph (the smallest set of shortest paths that cover all nodes). Again, in $G_X(\epsilon)$ this cover could have size $\Omega(n)$ even under (β,γ)-convexity. Active classification on unweighted graph has been also studied by Afshani et al. [2007]; Cesa-Bianchi et al. [2010]; Guillory and Bilmes [2011], but only with approximate reconstruction guarantees (i.e., $\Omega(n)$ queries may be needed for exact recovery).

Mazumdar and Saha [2017a] study exact cluster reconstruction with a SCQ oracle on weighted graphs, where weights express similarities. They prove a logarithmic query bound using a Monte-Carlo algorithm and a log-linear bound using a Las-Vegas algorithm. However, similarly to a stochastic block model, they assume that the weights are drawn from some latent distribution that depends on the clustering. Stochastic block models [Zhang et al., 2014; Gadde et al., 2016] and geometric block models [Chien et al., 2020] have been also considered as generative models for active clustering on graphs.

Center-based [Balcan and Long, 2013b], density-based [Mai et al., 2013], spectral [Wang and Davidson, 2010; Shamir and Tishby, 2011], and hierarchical [Eriksson et al., 2011; Krishnamurthy et al., 2012] clustering have been also studied in a more restricted active learning setting, where the algorithm has access to the pairwise distances through an oracle.

4.3 Preliminaries and notation

Our algorithms receive in input a semimetric represented by an undirected weighted graph $\mathcal{G} = (X, \mathcal{E}, d)$, where $d(x, y) > 0$ is the weight[3] of the edge $(x, y) \in \mathcal{E}$ and $|X| = n$. For $\epsilon > 0$, we let $G_X(\epsilon)$ be the undirected graph with vertex set X where $x, y \in X$ are connected if and only if $d(x, y) \le \epsilon$. Given a graph $G = (V, E)$ and two vertices $x, y \in V$, we denote by $d_G(x, y)$ the shortest-path distance between x and y in G and by $\rho(G)$ the number of connected components of G. Given any subset $U \subseteq V$, we use $G[U]$ to denote the subgraph of G induced by U, and we use $\Gamma(U)$ to denote the set of edges having exactly one endpoint in U. An edge $(x, y) \in \Gamma(U)$ is called a *cut edge* of U.

Let $x \in X$ and $r > 0$, the ball of center x and radius r is $B(x, r) = \{y \in X : d(x, y) \le r\}$. For any set $K \subseteq X$, we denote by $\mathcal{M}(K, r)$ the maximum cardinality of any subset A of K such that all distinct $x, y \in A$ satisfy $d(x, y) > r$. Following Gottlieb et al. [2017], we define the *density constant* of X as:[4]

$$\mu(X) = \min \{\mu \in \mathbb{N} : (x \in X) \wedge (r > 0) \Rightarrow \mathcal{M}(B(x, r), {}^{r}\!/_{2}) \le \mu\} \tag{4.1}$$

Therefore, in X, any ball of radius r contains at most $\mu(X)$ points at pairwise distance larger than $\frac{r}{2}$. The *density dimension* of X is $\mathrm{dens}(X) = \log_2 \mu(X)$. It is easy to see that, for any ball $B(x, r)$ and for any $\eta \in (0, 1)$ we have:

$$\mathcal{M}(B(x, r), \eta r) \le \mu(X)^{\left\lceil \log_2 \frac{1}{\eta} \right\rceil} \le ({}^{2}\!/_{\eta})^{\mathrm{dens}(X)} \tag{4.2}$$

When d is a metric, we have $\mathrm{dbl}(X) \le \mathrm{dens}(X) \le 2\,\mathrm{dbl}(X)$ where $\mathrm{dbl}(X)$ is the doubling dimension of X, see Lemma 4.17 in Appendix 4.A. We denote by $\mathcal{M}^*(\eta)$ the maximum of $\mathcal{M}(B(x, r), \eta r)$ over all $x \in X$ and $r > 0$. The quantity $\mathcal{M}^*(\eta)$ appears in our bounds, with η being a function of β and γ. Note that $\mathcal{M}^*(\eta) \le \left({}^{2}\!/_{\eta}\right)^{\mathrm{dens}(X)}$, by (4.2).

A k-clustering of X is a partition $\mathcal{C} = (C_1, \dots, C_k)$ of X. We denote by $\mathcal{C}(x)$ the cluster id of x, so that $x \in C_{\mathcal{C}(x)}$. We now introduce the characterization of the clusterings considered in this work. The following definition is for clusters with identical radii; a more complex version will be needed in the case with different radii, see Section 4.5.

Definition 4.1 ((β, γ)-convex clustering). *For any $\beta, \gamma \in (0, 1]$, a clustering $\mathcal{C} = (C_1, \dots, C_k)$ of X is (β, γ)-convex if $\exists\, \epsilon > 0$ such that for each $i \in [k]$ the following properties hold:*

1. connectedness*: the subgraph induced by C_i in $G_X(\epsilon)$ is connected*
2. local metric margin*: for all $x, y \in X$, if $x \in C_i$ and $y \notin C_i$, then $d(x, y) > \beta\epsilon$*
3. geodesic convexity with margin*: if $x, y \in C_i$, then in $G_X(\epsilon)$ any simple path between x and y of length at most $(1 + \gamma) d_{G_X(\epsilon)}(x, y)$ lies entirely in C_i*

The smallest value of ϵ satisfying the three properties is called the *radius* of the clusters.[5]

In this work, we assume the algorithm obtains information about the unknown target clustering $\mathcal{C}$ through *same-cluster queries* [Ashtiani et al., 2016]:

SCQ: for any $x, y \in X$, SCQ(x, y) returns $\{\mathcal{C}(x) = \mathcal{C}(y)\}$, where $\{\cdot\}$ is the indicator function

If only same-cluster queries are available, we assume that, together with X, the recovery algorithm is given a *seed node* $s_i \in C_i$ for each cluster C_i. Seed nodes are collected in a vector

[3] Our query bounds do not depend on the size of the weights.

[4] While [Gottlieb et al., 2017] uses open balls, we use closed balls.

[5] Actually, our algorithm of Section 4.4 works with *any* such ϵ. We use the minimum to disambiguate the radius.

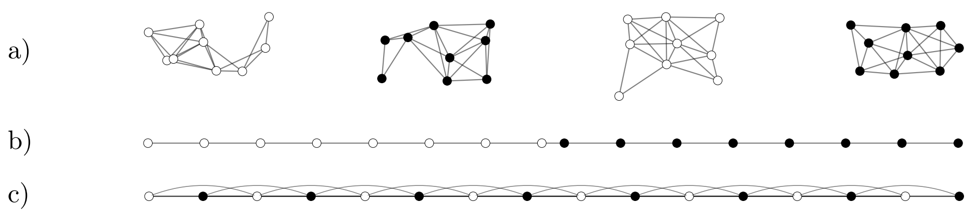


FIGURE 4.3.1: a) a clustering violating connectivity, b) a clustering violating local metric margin, c) a clustering violating geodesic convexity with margin. Empty nodes are in C_1, filled nodes are in C_2. Shown are the edges of $G_X(\epsilon)$.

$\boldsymbol{s} = (s_1, \dots, s_k)$. Without seed nodes, $\Omega(n)$ same-cluster queries are needed to recover $\mathcal{C}$ (see Section 4.8)[6].

When the cluster radii are different and/or unknown, or the seed nodes are not available, we show that there are instances where $\Omega(n)$ same cluster queries are needed. To overcome this limitation, we allow the algorithm to use another type of queries, called *seed queries*:

SEED: for any $S \subseteq X$ and $i \in [k]$, SEED(S, i) returns an arbitrary point $s_i \in C_i \cap S$, or NIL if $C_i \cap S = \emptyset$.

The SEED query is a kind of separation oracle: given a partition $(S, X \setminus S)$ of X and a cluster label i, the query can be used to check whether C_i is cut by this partition. Indeed, if C_i is cut, then SEED(S, i) and SEED$(X \setminus S, i)$ return a point of C_i belonging to, respectively, S and $X \setminus S$. Note that these queries are very natural. In a crowdsourcing setting, they correspond to asking the rater to identify an entity (say a picture of a car) within a set (say a set of pictures).

4.3.1 Necessity of the properties of Definition 4.1

We give some representative examples of degenerate clusterings resulting from dropping any of the properties of Definition 4.1. We assume that $k = 2$ and $X \subseteq \mathbb{R}^2$ with d being the Euclidean metric. The examples are depicted in Figure 4.3.1 below.

Removing connectivity. Choose any $\beta, \gamma \le 1$. Let X consist of disjoint subsets, each one with Euclidean diameter $\le \epsilon$, and sufficiently far from each other. Label any subsets as C_1 and the rest as C_2. Note that the second and third property of Definition 4.1 are satisfied.

Removing local metric margin. Choose any $\gamma \le 1$. Let C_1 be formed by collinear points equally spaced by ϵ, and the same for C_2, so that two extremal points of C_1 and C_2 are at arbitrarily small distance $\delta < \epsilon$. Note that the first and third property of Definition 4.1 are satisfied.

Removing geodesic convexity with margin. Choose any $\beta < \frac{1}{2}$. The set X is formed by collinear points equally spaced by $\frac{\epsilon}{2}$, with the points of cluster C_1 interleaved with those of cluster C_2. Note that the first and second property of Definition 4.1 are satisfied, but the third is violated for any $\gamma = \omega(1/n)$: take the two extreme nodes of C_1 and change their shortest path to use a node of C_2.

4.3.2 Relationship with other clustering notions

As noted, (β, γ)-convexity is meant to capture density-based clusterings produced by popular algorithms such as Single Linkage and DBSCAN. Those clusterings, however, are in general *not* recoverable with less than $\Omega(n)$ queries, since they allow for $\Omega(n)$ ties (points that can be assigned to one of two clusters in an arbitrary way). Therefore, (β, γ)-convexity should be thought of as an *additional* property, to be requested *on top of* existing notions of clustering in order to obtain efficient exact recovery. Here, we give two examples of how existing notions of clustering yield (β, γ)-convexity in particular cases.

[6] As we already said, if the sizes of the clusters are "roughly" balanced we can remove this assumption and sample $\widetilde{\mathcal{O}}(k)$ random nodes and use $\widetilde{\mathcal{O}}(k^2)$ same-cluster queries to obtain the seed nodes with high probability.

The first example is DBSCAN, whose parameters are the connectivity radius r and the density parameter κ. The clustering is defined by looking at $G_X(r)$, clustering together any maximal (sub)tree whose vertices all have degree at least κ, and assigning each remaining vertex to the same cluster as some of its neighbors (if it has a neighbor). Now, suppose that $G_X(r)$ is formed by k connected components, and each one of them is spanned by a tree on vertices that have degree at least κ. Then, by taking $\epsilon = r$, one can see that each such component is a cluster that is (β, γ)-dense for $\beta \geq 1$ and every $\gamma > 0$.

The second example is that of spherical clusters with margin [Ashtiani et al., 2016]. In this case we use the generalized (β, γ)-convexity, see Definition 4.10. Let $X \subset \mathbb{R}^d$, and suppose that every cluster C_i is contained in some ball $B_i = B(c_i, r_i)$, so that, for some $\delta > 0$, $B(c_i, r_i(1 + \delta)) \cap C_j = \emptyset$ for all $j \neq i$. Suppose that each C_i is a set of $(\frac{a}{\delta})^d$ points drawn independently and uniformly from B_i, for some constant $a > 0$. We claim that the resulting clustering is (β, γ)-convex with high probability, for $\beta = 1$ and any $\gamma > 0$. Indeed, $(\frac{a}{\delta})^d$ points draws uniformly at random from B_i are with high probability a $\frac{\delta r_i}{2}$-net for B_i [Vershynin, 2018] when a is sufficiently large. In this case, it is easy to see that C_i is connected in $G_X(\epsilon_i)$, where $\epsilon_i = \delta r_i$, thus satifying condition (1) of Definition 4.10. Moreover, by assumption, for any $x \in C_i$ and any $y \in C_j$ with $j \neq i$ we have $d(y, x) > \epsilon_i$, thus satisfying conditions (2) and (3) of Definition 4.10 for $\beta = 1$ and any $\gamma > 0$.

Chapter organization. Section 4.4 gives the step-by-step construction of our algorithm for recovering (β, γ)-convex clusterings. Section 4.5 extends (β, γ)-convexity to capture clusters with different radii, and shows how, by introducing SEED queries, our algorithm can be extended to this case. Section 4.6 shows how we can learn the radii as well as β or γ, using again SEED queries. Section 4.7 discusses an efficient implementation of our algorithms. Section 4.8 presents our query complexity lower bounds.

4.4 Exact recovery of (β, γ)-convex clusters

Throughout this section we assume that ϵ, β, γ are known, and that we know a vector of seed nodes $\boldsymbol{s} = (s_1, \dots, s_k)$, so that $s_i \in C_i$ for all i. Under these assumptions, our goal is to construct an algorithm, called RecoverCluster, that for any i returns C_i using $\mathcal{O}\big(k \log n + k\, \mathcal{M}^*\big(\frac{\beta\gamma}{2+\gamma}\big)\big)$ SCQ queries and time $\mathcal{O}(k(n + m))$. By running RecoverCluster once for each i, it is immediate to obtain a full cluster recovery algorithm, RecoverClustering, with the following guarantees. Note that $\mathcal{M}^*\big(\frac{\beta\gamma}{2+\gamma}\big) \leq (6/\beta\gamma)^{\mathrm{dens}(X)}$ since $\mathcal{M}^*(\eta) \leq (2/\eta)^{\mathrm{dens}(X)}$ and $\gamma \leq 1$.

Theorem 4.2. *Suppose $\mathcal{C}$ is (β, γ)-convex (Definition 4.1). Then* RecoverClustering$(X, \epsilon, \boldsymbol{s})$ *deterministically returns $\mathcal{C}$ in time $\mathcal{O}(k^2(n + m))$ using $\mathcal{O}\big(k^2 \log n + k^2\, (6/\beta\gamma)^{\mathrm{dens}(X)}\big)$* SCQ *queries.*

The rest of this section describes RecoverCluster and proves Theorem 4.2, with the exception of the running time bound, which is proven in Section 4.7. Unless otherwise specified, from now on G stands for $G_X(\epsilon)$.

4.4.1 Margin-based separation, and binary search on shortest paths.

We start with a subroutine MBS (Margin-Based Separator) that, given an input set Z and the cluster index i, computes $Z \cap C_i$. The routine uses the local metric margin and is efficient when the metric radius of Z is small.

Algorithm 6 MBS(Z, ϵ, u)

```
1: $U := \emptyset$
2: for each connected component $H$ of $G_Z(\beta\epsilon)$ do
3:     choose any $x \in V(H)$
4:     if SCQ$(x, u) = 1$ then add $V(H)$ to $U$
5: return $U$
```

Lemma 4.3. *For any $u \in X$ and any $Z \subseteq X$ such that $Z \subseteq B(z, r)$ for some $z \in X$ and $r < \infty$, $\mathrm{MBS}(Z, \epsilon, u)$ returns $Z \cap C_i$ using $\mathcal{M}^*\left(\frac{\beta\epsilon}{r}\right)$ SCQ queries, where $i = \mathcal{C}(u)$.*

Proof. Let $G_Z = G_Z(\beta\epsilon)$. If H is any connected component of G_Z, then $d(x, y) \le \beta\epsilon$ for all $x, y \in H$. Thus, a repeated application of the local metric margin implies that all nodes of H belong to the same cluster. Therefore, either $V(H) \cap C_i = \emptyset$, or $V(H) \subseteq C_i$. This shows that x is added to U if and only if $x \in C_i$, proving the correctness. For the query complexity, let P be the set of points x queried by the algorithm at line 4. Clearly P is an independent set in G_Z and, thus, $d(x, y) > \beta\epsilon$ for all distinct $x, y \in P$. By the definition of $\mathcal{M}^*$ this implies that $|P| \le \mathcal{M}^*(\frac{\beta\epsilon}{r})$. □

Next, we introduce a condition for finding efficiently a cut edge of C_i. A path $\pi = (x_1, \ldots, x_{|\pi|})$ is *C_i-prefixed* if there exists an index $j^* \in \left[|\pi|\right]$ such that $x_j \in C_i$ if and only if $j \in \{1, \ldots, j^*\}$

Lemma 4.4. *Let $C_i \subseteq R \subseteq X$ such that $G[R]$ is connected. Then, in $G[R]$, any shortest path between any $s_i \in C_i$ and any $s \in R \setminus C_i$ is C_i-prefixed.*

Proof. Suppose by contradiction that in $G[R]$ there exists a shortest path π between $s_i \in C_i$ and $s \in R \setminus C_i$ that is not C_i-prefixed. Then some prefix π' of π is a shortest path between $s_i \in C_i$ and $s'_i \in C_i$ that intersects $R \setminus C_i$. Now observe that π' is a shortest path in G, too. This holds because any shortest path between s_i and s'_i in G lies inside $G[C_i]$ by geodesic convexity, and thus inside $G[R]$ as $C_i \subseteq R$. By geodesic convexity this implies that $\pi' \subseteq G[C_i]$, a contradiction. □

Finally, we observe that, in a C_i-prefixed simple path, a cut edge of C_i can be found via binary search from the endpoints of the path. This yields a subroutine FindCutEdge with the following guarantees (pseudocode in Appendix 4.B):

Remark 4.1. *Given a simple C_i-prefixed path π, $\mathrm{FindCutEdge}(\pi)$ returns the unique cut edge of C_i in π using $\mathcal{O}(\log n)$ SCQ queries.*

4.4.2 Cluster separators

We introduce the notion of cluster separator, which is at the heart of our algorithm.

Definition 4.5. *A partition (S_i, S_j) of X is an (i, j)-separator of X if $S_i \cap C_j = S_j \cap C_i = \emptyset$.*

This is similar to half-spaces in abstract closure systems [Seiffarth et al., 2019], where we would have $C_i \subseteq S_i$ and $C_j \subseteq S_j$. We use the weaker condition $S_i \cap C_j = S_j \cap C_i = \emptyset$ because in some of our algorithms X will be a subset of the input set, in which case $C_j \subseteq X$ might not hold. On the other hand, we will always make sure that $C_i \subseteq X$ holds.

Now, if $C_i \subseteq X$ and (S_i, S_j) is an (i, j)-separator for X, then $C_i \subseteq S_i$ but $C_j \cap S_i = \emptyset$. Thus, if we could compute an (i, j)-separator for all $j \neq i$, then we could compute C_i by a simple set intersection. Unfortunately, it is not clear how to compute (S_i, S_j) for an arbitrary j, even given the seed node s_j. However, as we shall see, we can compute (S_i, S_j) if we know a cut edge (u_i, u_j) between C_i and C_j. The trick is to take each $x \in X$ and look at its distance $d_G(x, u_i)$ from u_i. If $d_G(x, u_i) < \frac{1}{\gamma}$, then we learn whether $u_i \in C_i$ using MBS. If instead $d_G(x, u_i) \ge \frac{1}{\gamma}$, then the comparison $d_G(x, u_i) \le d_G(x, u_j)$ tells us whether x should be in S_i or in S_j, without using any query. This is implied by geodesic convexity through the following lemma, proven in Appendix 4.B:

Lemma 4.6. *Let $(u_i, u_j) \in G$ be a cut edge between C_i and C_j. For all $x \in X$ with $\frac{1}{\gamma} \le d_G(u_i, x) < \infty$, if $d_G(u_i, x) \le d_G(u_j, x)$ then $x \notin C_j$, and if $d_G(u_i, x) \ge d_G(u_j, x)$ then $x \notin C_i$.*

The intuition is that $d_G(u_i, x) \le d_G(u_j, x)$ and $x \in C_j$ cannot both hold, because this would violate the geodesic convexity of C_j, and the same holds when i and j are exchanged.

The above discussion leads to CSeparator (Algorithm 7), whose correctness and query cost are proven in Lemma 4.7. Clearly enough, to use CSeparator we need to compute the cut edge (u_i, u_j), and we show how to do so in the next sections.

Lemma 4.7. *Let $(u_i, u_j) \in G$ be a cut edge between C_i and C_j. Then, $\mathrm{CSeparator}(G, u_i, u_j)$ returns a pair (S_i, S_j) that is an (i, j)-separator of $V(G)$, using $\mathcal{O}(\mathcal{M}^*(\beta\gamma))$ SCQ queries.*

Algorithm 7 CSeparator(G, u_i, u_j)

```
1: $Z := \{x \in V(G) : d_G(u_i, x) < 1/\gamma\}$
2: $Z_i :=$ MBS$(Z, \epsilon, u_i)$,   $Z_j := Z \setminus Z_i$
3: $U_i := \emptyset$,   $U_j := \emptyset$
4: for each $x \in V(G) \setminus Z$ do
5:     choose any $x \in V(H)$
6:     if $d_G(x, u_i) \le d_G(x, u_j)$ then add $x$ to $U_i$
7:     else add $x$ to $U_j$
8: return $(Z_i \cup U_i,\ Z_j \cup U_j)$
```

Proof. The query bound follows from Lemma 4.3 by observing that $Z \subseteq B(u_i, \epsilon/\gamma)$. For the correctness, we show that (Z_i, Z_j) is an (i,j)-separator of Z and (U_i, U_j) is an (i,j)-separator of $V(G) \setminus Z$. For Z, Lemma 4.3 guarantees that $Z_i = C_i \cap Z$, and the algorithm sets $Z_j = Z \setminus Z_i$. So $Z_i \cap C_j = Z_j \cap C_i = \emptyset$, and (Z_i, Z_j) is an (i,j)-separator of Z. Consider now any $x \in V(G) \setminus Z$. By definition of Z, we have $d_G(x, u_i) \ge \frac{1}{\gamma}$. Therefore, by Lemma 4.6, if the algorithm assigns x to U_i then $x \notin C_j$, and if the algorithm assigns x to U_j then $x \notin C_i$. Therefore $U_i \cap C_j = U_j \cap C_i = \emptyset$, and (U_i, U_j) is an (i,j)-separator of $V(G) \setminus Z$. □

4.4.3 Recovering a single cluster

We can now describe RecoverCluster (Algorithm 8). The algorithm starts by computing R_i, the set of nodes reachable from s_i in G, and the corresponding induced subgraph $G_i = G[R_i]$. Note that, by the connectedness of the clusters, initially R_i is simply the union of C_i and zero or more other clusters. Now, we search for some seed node $s_h \in \boldsymbol{s}$, such that $s_h \in R_i$ but $s_h \neq s_i$. If no such node is found, then $R_i = C_i$ and we are done. Otherwise, we compute the shortest path π between s_h and s_i in G_i. By Lemma 4.4, π is C_i-prefixed, and so by Remark 4.1 we can find a cut edge (u_i, u_j) of C_i with $\mathcal{O}(\log n)$ SCQ queries. With the cut edge (u_i, u_j), we can compute an (i,j)-separator (S_i, S_j) of $X = V(G)$ using CSeparator. Finally, we update G_i to be the connected component of s_i in $G[R_i \cap S_i]$, and R_i to be its node set. By definition of (S_i, S_j), this update removes from G_i all points of C_j, so we have reduced by at least one the number of clusters other than C_i intersected by R_i. After at most $k-1$ of these rounds, we will be left with $R_i = C_i$.

Unfortunately, this process has a problem: we can run out of seeds in R_i. Indeed, by taking $R_i \cap S_i$, we could remove every seed node s_h, even if $R_i \cap S_i$ still contains points of C_h. If this is the case, then, even though $R_i \neq C_i$, RecoverCluster would be stuck, unable to compute a new cut edge to remove some cluster from R_i. One is tempted to ignore the fact that $s_h \notin R_i$, and simply compute the shortest path between s_h and s_i for all $h \neq i$, obtaining a set of different cut edges. This however does not work, as all those shortest paths could use the same cut edge (u_i, u_j).

We bypass this obstacle by exploiting the separators found by RecoverCluster. By carefully analysing the cuts induced by those separators, we devise an algorithm, FindNewSeed, that either finds some new seed $s_h \in R_i \setminus C_i$ or certifies that $R_i = C_i$. FindNewSeed is invoked by RecoverCluster at the beginning of each round, and we describe it in Section 4.4.4.

Algorithm 8 RecoverCluster$(G, \epsilon, \boldsymbol{s}, i)$

```
1: $G_i := \{$the connected component of $s_i$ in $G\}$,  $R_i := V(G_i)$
2: $\boldsymbol{u} :=$ an empty vector
3: while True do
4:     $s_h :=$ FindNewSeed$(G, R_i, \epsilon, \boldsymbol{s}, \boldsymbol{u}, i)$
5:     if $s_h =$ NIL then stop and return $R_i$
6:     $\pi(s_i, s_h) :=$ ShortestPath$(G[R_i], s_i, s_h)$
7:     $(u_i, u_j) :=$ FindCutEdge$(\pi(s_i, s_h))$
8:     add $u_i$ to $\boldsymbol{u}$
9:     $(S_i, S_j) :=$ CSeparator$(G, u_i, u_j)$
10:    $G_i := \{$the connected component of $s_i$ in $G[R_i \cap S_i]\}$,  $R_i := V(G_i)$
```

Lemma 4.8. RecoverCluster$(G, \epsilon, \boldsymbol{s}, i)$ *returns* C_i *using* $\mathcal{O}\big(k \log n + k\,\mathcal{M}^*(\frac{\beta\gamma}{2+\gamma})\big)$ SCQ *queries.*

The proof is along the lines of the discussion above, see Appendix 4.B.

4.4.4 Finding new seed nodes

We describe FindNewSeed, which finds a node $s_h \in R_i \setminus C_i$ if $R_i \setminus C_i \neq \emptyset$, and otherwise detects that $R_i = C_i$ and returns NIL. The key idea behind FindNewSeed is the following: if R_i does not contain any seed $s_h \in \boldsymbol{s}$ other than s_i, then for each $h \neq i$ either $C_h \cap R_i = \emptyset$, or, by the connectedness of $G[C_h]$, the cut $\Gamma(R_i)$ must contain some edge of $G[C_h]$. Therefore, the task boils down to finding such an edge, or deciding that no such edge exists. Clearly, we cannot just check all edges in $\Gamma(R_i)$, as this would require too many queries. Thus, we proceed as follows.

Consider the beginning of a generic iteration of RecoverCluster, and let $\boldsymbol{u}$ be the set of all nodes u_i that appeared in a cut edge used in some previous iteration. First, for every $u \in \boldsymbol{u}$, we consider the set Z of nodes $x \in R_i$ such that $d_G(u,x) < \frac{2}{\gamma} + 1$. Then, like we did in CSeparator, we use MBS to recover efficiently the subset $Z \setminus C_i$. If this subset is nonempty, then we just return any $x \in Z \setminus C_i$ and we are done. If after considering every $u \in \boldsymbol{u}$ we have not found a seed, then we turn to the remaining nodes, that is, all nodes x such that $d_G(u,x) \geq \frac{2}{\gamma} + 1$ for all $u \in \boldsymbol{u}$. In this case, as a consequence of geodesic convexity with margin we prove the following structural result: if x has an edge $(x,y) \in \Gamma(R_i)$, then $x \notin C_i$. So, if any such x exists, which we can check without making any query, then we can again return x. If both attempts to find $x \in R_i \setminus C_i$ fail, we can show that necessarily $R_i = C_i$.

Lemma 4.9 below states the guarantees of FindNewSeed. Its proof is found in Appendix 4.B.

Algorithm 9 FindNewSeed$(G, R_i, \epsilon, \boldsymbol{s}, \boldsymbol{u}, i)$

```
1: if $\boldsymbol{s} \cap R_i \neq \{s_i\}$ then return any $s \in \boldsymbol{s} \cap R_i \setminus \{s_i\}$;
2: for each $u \in \boldsymbol{u}$ do
3:     $Z = \{x \in R_i \,:\, d_G(u,x) < 2/\gamma + 1\}$
4:     $Z_i := \text{MBS}(Z, \epsilon, u)$ ;
5:     if $Z \setminus Z_i \neq \emptyset$ then return any $x \in Z \setminus Z_i$;
6: if $\exists\,(x,y) \in \Gamma(R_i)$ such that $\forall u \in \boldsymbol{u} \,:\, d_G(u,x) \geq 2/\gamma + 1$ then return any such $x$;
7: else return NIL;
```

Lemma 4.9. *At each iteration of* RecoverCluster$(G, \epsilon, \boldsymbol{s}, i)$, FindNewSeed$(G, R_i, \epsilon, \boldsymbol{s}, \boldsymbol{u})$ *returns an* $x \in R_i \setminus C_i$ *if* $R_i \neq C_i$*, or* NIL *if* $R_i = C_i$*, using at most* $k\,\mathcal{M}^*(\frac{\beta\gamma}{2+\gamma})$ SCQ *queries. Moreover,* FindNewSeed *can be adapted to make at most* $k\,\mathcal{M}^*(\frac{\beta\gamma}{2+\gamma})$ SCQ *queries over the entire execution of* RecoverCluster.

4.5 Extension to clusters with different radii

In this section we generalize the notion of (β,γ)-convexity (Definition 4.1) so to allow each cluster C_i to have its own radius, denoted by ϵ_i. Then, by using SEED queries, we will extend our cluster recovery algorithm to this generalized setting. For technical reasons, we need to strengthen geodesic convexity in a hereditary fashion.

Definition 4.10 (generalized (β,γ)-convex clustering)**.** *For any* $\beta, \gamma \in (0,1]$*, a* k*-clustering* $\mathcal{C} = (C_1, \dots, C_k)$ *of* X *is* (β,γ)*-convex if* $\forall\, i \in [k] : \exists\, \epsilon_i > 0$ *such that the following properties hold:*

1. connectedness*: the subgraph induced by* C_i *in* $G_X(\epsilon_i)$ *is connected*
2. local metric margin*: for all* $x, y \in X$*, if* $x \in C_i$ *and* $y \notin C_i$*, then* $d(x,y) > \beta\epsilon_i$
3. geodesic convexity with margin*: for any* $\epsilon \leq \epsilon_i$*, if* $x, y \in C_i$ *and* $d_{G_X(\epsilon)}(x,y) < \infty$*, then in* $G_X(\epsilon)$ *any simple path between* x *and* y *of length at most* $(1+\gamma)d_{G_X(\epsilon)}(x,y)$ *lies entirely in* C_i

In Lemma 4.20 in the Appendix, we show that if the conditions above are satisfied, then they are satisfied in particular by the smallest ϵ_i such that C_i is connected in $G_X(\epsilon_i)$.[7] Therefore, without loss of generality, we assume that each ϵ_i satisfies this minimality assumption.

We turn to the recovery of $\mathcal{C}$. We show that, with some care, the problem can be reduced to the case of identical radii, at the price of making $\mathcal{O}(k^2)$ SEED queries. This gives an algorithm RecoverClustering2 with the following guarantees, where $\boldsymbol{\epsilon} = (\epsilon_1, \dots, \epsilon_k)$ is the vector of the radii:

Theorem 4.11. *Suppose $\mathcal{C}$ is (β, γ)-convex. Then,* RecoverClustering2$(X, \boldsymbol{\epsilon}, \boldsymbol{s})$ *deterministically returns $\mathcal{C}$, has the same runtime as* RecoverClustering$(X, \boldsymbol{\epsilon}, \boldsymbol{s})$*, and uses the same number of* SCQ *queries as* RecoverClustering$(X, \boldsymbol{\epsilon}, \boldsymbol{s})$*, plus at most $\mathcal{O}(k^2)$* SEED *queries.*

The basic idea of RecoverClustering2 is to invoke RecoverCluster for each $i \in [k]$, as we did for the case of identical radii, using the graph $G_X(\epsilon_i)$ when we want to recover C_i. This does not work straight away, however: in $G_X(\epsilon_i)$, any cluster C_j with $\epsilon_j < \epsilon_i$ is by definition not required to satisfy geodesic convexity, which means that RecoverCluster can fail. However, we can show that this approach works if we adopt the following precautions:

1. recover the clusters in nondecreasing order of radius
2. when recovering C_i, restrict $G_X(\epsilon_i)$ to its connected component containing s_i
3. after recovering C_i, delete it from X.

Note that, for each i, this procedure works on a potentially different graph — thresholded at a different radius, and containing only a subset of the original points. Thus, its correctness may not be obvious. In particular, it may not be obvious that the clustering induced by the sub-instance used at the i-th iteration is (β, γ)-convex, which is necessary for RecoverCluster to work. We show that it is: for every i, at the i-th iteration, the residual clustering is (β, γ)-convex. Thus the input to RecoverCluster satisfies the hypotheses of Lemma 4.8, and by that lemma, RecoverCluster will return C_i using $\mathcal{O}\big(k \log n + k\, \mathcal{M}^*(\frac{\beta\gamma}{2+\gamma})\big)$ SCQ queries, as desired. In all this, the role of SEED queries is to find one seed node s_h for each cluster in the connected component of $G_X(\epsilon_i)$ containing s_i, as required by RecoverCluster.

The formal construction of RecoverClustering2 and the proof of Theorem 4.11 are given in Appendix 4.C.1.

4.6 Learning the radii and the convexity parameters

In this section we show how to use SEED queries to learn the cluster radii and to deal with the case where one of β or γ is unknown. For learning the radii, we prove:

Theorem 4.12. *Suppose $\mathcal{C}$ is (β, γ)-convex. Then, the cluster radii $\epsilon_1, \dots, \epsilon_k$ can be learned using $\mathcal{O}(k \log n)$* SEED *queries in time $\mathcal{O}(m\, \alpha(m, n) + kn \log n)$, where $\alpha(m, n)$ is the functional inverse of the Ackermann function.*[8]

This result hinges on three observations. First, as said above, ϵ_i is actually the smallest ϵ such that all nodes of C_i belong to a single connected component of $G_X(\epsilon)$. Second, with $\mathcal{O}(1)$ SEED queries, we can test whether C_i belongs to a single connected component of G, for any given graph G, see Claim 4.1. Third, if T is any minimum spanning tree of $\mathcal{G}$, then the connected components of $G_X(\epsilon)$ are exactly the connected components of $T(\epsilon)$, see Claim 4.2. Our algorithm starts by computing T, which takes time $\mathcal{O}(m\, \alpha(m, n))$ where α is inverse Ackermann, see [Chazelle, 2000]. Then, for each cluster C_i, we perform a binary search to find the smallest edge weight ϵ such that C_i is connected in $T(\epsilon)$. All details are given in Appendix 4.D. In Section 4.8, we also prove a nearly-matching lower bound of $\Omega(k \log \frac{n}{k})$ queries.

We conclude by discussing the case where one among β and γ is unknown. Equipped with the SEED queries, we make a series of halving guesses for β or γ, until we detect that the clustering is correct. This yields the following result (proof in Appendix 4.D):

[7] Note that this is different from requiring that $G_X(\epsilon_i)[C_i]$ is connected; here we are only requiring that any two points of C_i have a connecting path in $G_X(\epsilon_i)$. Lemma 4.20 however shows that the smallest ϵ_i that satisfies one condition is also the smallest ϵ_i that satisfies the other condition.

[8] For all practical purposes, α can be considered constant. For instance, $\alpha(m, m) \leq 4$ for all $m \leq \frac{1}{8} 2^{2^{2^{65536}}}$.

Theorem 4.13. *Suppose $\mathcal{C}$ is (β,γ)-convex, and let $R = \log(\frac{4}{\beta\gamma})\,\mathrm{dens}(X)$. If only one between β and γ is unknown, then we can recover $\mathcal{C}$ with a multiplicative overhead of $\mathcal{O}(R)$ in both query cost and running time, plus $\mathcal{O}(k^2R)$ SCQ queries and $\mathcal{O}(kR)$ SEED queries. This applies to each one of our algorithms (i.e., with radii that are identical or not, known or unknown).*

4.7 Bounds on the running time

All our algorithms admit efficient implementations, with a running time linear in the size of $\mathcal{G}$ (or essentially linear, see Section 4.6). Here, we give a quick overview of these implementations; for a more complete discussion, see Appendix 4.E.

Recall that our input is the weighted graph $\mathcal{G} = (X, \mathcal{E}, d)$, where $(u, v) \in \mathcal{E}$ if and only if $d(u, v) < \infty$. Recall also that $n = |X|$ and $m = |\mathcal{E}|$. We assume that d can be accessed in constant time, which can be achieved with a hash map, whose construction takes time $O(m)$ [Fredman et al., 1982]. We further assume that, for any graph G and for any $x \in V(G)$, the edges incident to x can be listed in constant time per edge. Under these assumptions, the following basic fact holds:

Remark 4.2. *Given $\mathcal{G}$, for any ϵ, the graph $G_X(\epsilon)$ can be computed in time $\mathcal{O}(n+m)$. This holds in general for thresholding any subgraph of $\mathcal{G}$.*

Using Observation 4.2, we can easily implement RecoverCluster so that it runs in time $\mathcal{O}(k^2(n+m))$: essentially, for $\mathcal{O}(k^2)$ times the algorithm computes the distances of all nodes of $G_X(\epsilon)$ from some given node u, which takes time $\mathcal{O}(n+m)$ via breadth-first search. Since RecoverCluster is invoked once per each cluster, this would give a total running time of $\mathcal{O}(k^3(n+m))$ for both RecoverClustering and RecoverClustering2. With some extra effort, however, we show how to adapt RecoverCluster so that it runs $\mathcal{O}(k(n+m))$, by amortizing in particular the cost of its subroutine FindNewSeed. In the end, we obtain a total running time of $\mathcal{O}(k^2(n+m))$ for both our cluster recovery algorithms, RecoverClustering and RecoverClustering2.

4.8 Lower Bounds

In this section we show that some of our parameters and assumptions are necessary, and in particular: (1) in general, to recover a (β,γ)-convex clustering, any algorithm needs $\Omega(2^{\mathrm{dens}(X)})$ queries; (2) to recover a (β,γ)-convex clustering without initial seed nodes, any algorithm needs $\Omega(n)$ SCQ queries; (3) to learn the radii of k clusters, any algorithm needs $\mathcal{O}(k \log \frac{n}{k})$ SCQ and/or SEED queries. The full proofs are deferred to Appendix 4.F.

Theorem 4.14 (Dependence on dens(X).). *Choose any $\beta, \gamma \in (0, 1)$. There is a distribution of (β,γ)-convex 2-clusterings $\mathcal{C}$ (Definition 4.1 or Definition 4.10), where $n = |X| = 2^{\mathrm{dens}(X)}$ is arbitrarily large, such that any algorithm (even randomized) needs $\Omega(2^{\mathrm{dens}(X)})$ SCQ and/or SEED queries to recover $\mathcal{C}$ with constant probability. This holds even if β, γ, ϵ are known.*

Theorem 4.15 (Necessity of seeds.). *Choose any $\beta, \gamma \in (0, 1]$. There is a distribution of (β,γ)-convex 2-clusterings $\mathcal{C}$ (Definition 4.1 or Definition 4.10), where $X \subseteq \mathbb{R}^2$ with $n = |X|$ arbitrarily large and d the Euclidean distance, such that any algorithm (even randomized) needs $\Omega(n)$ SCQ queries to recover $\mathcal{C}$ with constant probability if no seed nodes are given. This holds even if γ, α, ϵ are known.*

Theorem 4.16 (Cost of learning the radii.). *For any $k \geq 2$, and any sufficiently large n, there exists a distribution of $(1/2, 1)$-convex k-clusterings (Definition 4.1 or Definition 4.10) on n points such that any algorithm (even randomized) needs $\Omega(k \log \frac{n}{k})$ SEED and/or SCQ queries to learn the radii of all clusters with constant probability.*

Appendix

4.A Supplementary material for Section 4.3

Lemma 4.17. *If X is a metric space, then $\mathrm{dbl}(X) \leq \mathrm{dens}(X) \leq 2\,\mathrm{dbl}(X)$ where $\mathrm{dbl}(X)$ and $\mathrm{dens}(X)$ are respectively the doubling dimension and the density dimension of X.*

Proof. Recall that $\mathrm{dbl}(X) = \log_2 D(X)$, where $D(X)$ is the doubling constant of X:

$$D(X) = \min\left\{D \in \mathbb{N} \,:\, (x \in X) \wedge (r > 0) \Rightarrow \mathcal{N}\Big(B(x,r), \frac{r}{2}\Big) \leq D\right\} \tag{4.3}$$

where $\mathcal{N}(K,\eta)$ is the covering number of K, that is, the smallest number of closed balls of radius η whose union contains K. We recall from Section 4.3 that $\mathrm{dens}(X) = \log_2 \mu(X)$, where:

$$\mu(X) = \min\left\{\mu \in \mathbb{N} \,:\, (x \in X) \wedge (r > 0) \Rightarrow \mathcal{M}\Big(B(x,r), \frac{r}{2}\Big) \leq \mu\right\} \tag{4.4}$$

Now, since we are in a metric space, we have the well-known relationship:

$$\mathcal{M}(K, 2\eta) \leq \mathcal{N}(K,\eta) \leq \mathcal{M}(K,\eta) \tag{4.5}$$

On the one hand, $\mathcal{N}(K,\eta) \leq \mathcal{M}(K,\eta)$ implies $D(X) \leq \mu(X)$, and therefore $\mathrm{dbl}(X) \leq \mathrm{dens}(X)$. On the other hand, $\mathcal{M}(K,2\eta) \leq \mathcal{N}(K,\eta)$ implies:

$$\begin{aligned}
\mathcal{M}(B(x,r), {}^{r}/_{2}) &\leq \mathcal{N}(B(x,r), {}^{r}/_{4}) && (4.6)\\
&\leq \mathcal{N}(B(x,r), {}^{r}/_{2}) \cdot D(X) && (4.7)\\
&\leq D(X) \cdot D(X) && (4.8)
\end{aligned}$$

Therefore, $\mu(X) \leq D(X)^2$, and $\mathrm{dens}(X) \leq 2\,\mathrm{dbl}(X)$. □

4.B Supplementary material for Section 4.4

4.B.1 Pseudocode of FindCutEdge

Algorithm 10 FindCutEdge($\pi(s_i, s_h)$)

```
1: if $|\pi(s_i, s_h)| = 1$ thenreturn $\pi(s_i, s_h)$
2: choose a median node $x \in \pi(s_i, s_h)$
3: if $\textsc{scq}(s_i, x) = +1$ thenreturn FindCutEdge($\pi(x, s_h)$)
4: elsereturn FindCutEdge($\pi(s_i, x)$)
```

4.B.2 Proof of Lemma 4.6

Lemma 4.6. *Let $(u_i, u_j) \in G$ be a cut edge between C_i and C_j. For all $x \in X$ with $\frac{1}{\gamma} \leq d_G(u_i, x) < \infty$, if $d_G(u_i, x) \leq d_G(u_j, x)$ then $x \notin C_j$, and if $d_G(u_i, x) \geq d_G(u_j, x)$ then $x \notin C_i$.*

Proof. Suppose $d_G(u_i, x) \leq d_G(u_j, x)$ and $x \in C_j$; we show this leads to a contradiction. Consider the path $\pi = \pi(x, u_i) + (u_i, u_j)$. First, π is a simple path. If this was not the case, then we would have $u_j \in \pi(x, u_i)$; but this would imply $d_G(u_j, x) \leq d_G(u_i, x) - 1$, contradicting our assumption $d_G(u_i, x) \leq d_G(u_j, x)$. Second, we have:

$$\begin{aligned}
|\pi| &= d_G(x, u_i) + 1 && && (4.9)\\
&\leq d_G(x, u_j) + 1 && \text{since } d_G(u_i, x) \leq d_G(u_j, x) && (4.10)\\
&\leq d_G(x, u_j)\,(1+\gamma) && \text{since } d_G(x, u_j) \geq d_G(x, u_i) \geq \frac{1}{\gamma} && (4.11)
\end{aligned}$$

Thus, π is a simple path between two nodes of C_j, with length at most $(1+\gamma)$ times their distance in G, and containing a node of C_i. This violates the geodesic convexity of C_j. We conclude that $x \notin C_j$. The other case is symmetric. □

4.B.3 Proof of Lemma 4.8

Lemma 4.8. RecoverCluster$(G, \epsilon, \boldsymbol{s}, i)$ *returns* C_i *using* $\mathcal{O}\big(k \log n + k\,\mathcal{M}^*(\frac{\beta\gamma}{2+\gamma})\big)$ SCQ *queries.*

Proof. First, we prove the correctness. For each $\ell = 1, 2, \ldots$, we denote by R_i^ℓ the set R_i at the beginning of the ℓ-th iteration, and by $\kappa(R_i^\ell)$ the number of distinct clusters that R_i^ℓ intersects. We show that, at the beginning of the ℓ-th iteration, the following invariants hold:

I1. $G[R_i^\ell]$ is connected

I2. $C_i \subseteq R_i^\ell$

I3. $\kappa(R_i^\ell) \leq \kappa(R_i^1) - (\ell - 1)$

Note that I2 and I3 imply that the algorithm terminates by returning C_i (line 5) after at most k iterations. Invariant I1 holds by the construction of R_i^ℓ, so we focus on proving I2 and I3.

Suppose first $\ell = 1$. Then, I2 holds by the assumptions of the lemma, and I3 is trivial. Suppose then I1, I2, I3 hold for some $\ell \geq 1$, and that iteration $\ell + 1$ exists; we show that I2, I3 hold at iteration $\ell + 1$ as well. First, if iteration $\ell + 1$ exists, then $C_i \subsetneq R_i^\ell$. In this case, by Lemma 4.9, FindNewSeed will return a node $s_h \in R_i^\ell \setminus C_i$. Since $G[R_i^\ell]$ is connected by I1, the shortest path $\pi(s_i, s_h)$ exists in $G[R_i^\ell]$. Because of Lemma 4.4 applied to $R = R_i^\ell$, such a path is also C_i-prefixed. Therefore, FindCutEdge$(\pi(s_i, s_h))$ returns a cut edge (u_i, u_j) of C_i in $\pi(s_i, s_h)$. At this point the hypotheses of Lemma 4.7 are satisfied, and therefore the output (S_i, S_j) of CSeparator(G, u_i, u_j) is an (i, j)-separator of $V(G)$. Hence, $C_i \subseteq S_i$ and $C_j \cap S_i = \emptyset$. Therefore, $C_i \subseteq R_i^\ell \cap S_i$. This implies that the connected component of s_i in $G[R_i^\ell \cap S_i]$ still contains C_i. The vertex set of this connected component is precisely $R_i^{\ell+1}$ (line 10). Therefore, I2 holds at iteration $\ell + 1$. Moreover, observe that $u_j \in R_i^\ell$, since $u_j \in \pi(s_i, s_h) \subseteq R_i^\ell$. Therefore $R_i^\ell \cap C_j \neq \emptyset$. However, by construction, $R_i^{\ell+1} \subseteq R_i^\ell \cap S_i$ and $S_i \cap C_j = \emptyset$ since (S_i, S_j) is an (i, j)-separator of $V(G)$. Therefore, $\kappa(R_i^{\ell+1}) \leq \kappa(R_i^\ell) - 1$. Because I3 holds at ℓ, $\kappa(R_i^{\ell+1}) \leq \kappa(R_i^1) - (\ell - 1) - 1 = \kappa(R_i^1) - ((\ell + 1) - 1)$. So, I3 holds at iteration $\ell + 1$, too.

Finally, we bound the number of queries. First, by Lemma 4.9, FindNewSeed makes at most $k\,\mathcal{M}^*(\frac{\beta\gamma}{2+\gamma})$ SCQ in total across all iterations. Second, FindCutEdge makes $\mathcal{O}(\log n)$ queries at each iteration, see Observation 4.1. Third, CSeparator makes at most $\mathcal{M}^*(\beta\gamma)$ queries at each iteration, see Lemma 4.7. Summing the three terms we obtain the claimed bound. □

4.B.4 Proof of Lemma 4.9

Lemma 4.9. *At each iteration of* RecoverCluster$(G, \epsilon, \boldsymbol{s}, i)$, FindNewSeed$(G, R_i, \epsilon, \boldsymbol{s}, \boldsymbol{u})$ *returns an* $x \in R_i \setminus C_i$ *if* $R_i \neq C_i$, *or* NIL *if* $R_i = C_i$, *using at most* $k\,\mathcal{M}^*(\frac{\beta\gamma}{2+\gamma})$ SCQ *queries. Moreover,* FindNewSeed *can be adapted to make at most* $k\,\mathcal{M}^*(\frac{\beta\gamma}{2+\gamma})$ SCQ *queries over the entire execution of* RecoverCluster.

We need to prove two technical results, Lemma 4.18 and Lemma 4.19. Then, we will prove Lemma 4.9.

Lemma 4.18. *Let* G *be such that* $C_i \subseteq V(G)$, *and let* (S_i, S_j) *be an* (i, j)*-separator for* $V(G)$ *obtained from* CSeparator$(V(G), u_i, u_j)$. *If* $(x, y) \in \Gamma(S_i)$ *and* $d_G(u_i, x) \geq \frac{2}{\gamma} + 1$, *then* $x \notin C_i$.

Proof. We show that $x \in C_i$ violates the geodesic convexity of C_i. First $(x, y) \in \Gamma(S_i)$ implies $y \in S_j$. Moreover, $d_G(u_i, y) \geq d_G(u_i, x) - 1 > \frac{1}{\gamma}$. Now recall the code of CSeparator. Since $d_G(u_i, y) > \frac{1}{\gamma}$, then $y \notin Z_j$. But $y \in S_j = Z_j \cup U_j$, and therefore $y \in U_j$. This means that CSeparator executed line 7, which happens only if:

$$d_G(u_j, y) < d_G(u_i, y) \tag{4.12}$$

Now consider the path $\pi = (u_i, u_j) + \pi(u_j, y) + (y, x)$ from u_i to x where $\pi(u_j, y)$ has length $d_G(u_j, y)$. First, we observe that π is a simple path. Suppose indeed by contradiction that π is not simple. Since $\pi(u_j, y)$ is simple (it is a shortest path), and since $u_i \neq x$ (because $d_G(u_i, x) \geq 1$ by assumption), we must have $u_i \in \pi(u_j, y)$ or $x \in \pi(u_j, y)$. If $u_i \in \pi(u_j, y)$, then $d_G(u_j, y) > d_G(u_i, y)$, which contradicts (4.12). If instead $x \in \pi(u_j, y)$, then $d_G(u_j, y) > d_G(u_j, x)$, which gives:

$$\begin{aligned}
d_G(u_j, y) &> d_G(u_j, x) && && (4.13)\\
&\geq d_G(u_i, x) && \text{since } x \in S_i && (4.14)\\
&\geq d_G(u_i, y) - 1 && \text{since } (x, y) \in E(G) && (4.15)
\end{aligned}$$

which, since d_G is integral, implies $d_G(u_j, y) \geq d_G(u_i, y)$. This contradicts again (4.12). Thus, π is a simple path.

Now we show that $|\pi| \leq d_G(u_i, x)\,(1 + \gamma)$. From (4.12), we have:

$$\begin{aligned}
d_G(u_j, y) &\leq d_G(u_i, y) - 1 && \text{since } d_G \in \mathbb{N} && (4.16)\\
&\leq d_G(u_i, x) + d_G(x, y) - 1 && \text{since } (x, y) \in E(G) && (4.17)\\
&= d_G(u_i, x) && && (4.18)
\end{aligned}$$

And therefore:

$$\begin{aligned}
|\pi| &= d_G(u_j, y) + 2 && && (4.19)\\
&\leq d_G(u_i, x) + 2 && \text{since } d_G(u_j, y) \leq d_G(u_i, x) && (4.20)\\
&\leq d_G(u_i, x)\,(1 + \gamma) && \text{since } d_G(u_i, x) \geq \frac{2}{\gamma} && (4.21)
\end{aligned}$$

Therefore π is a simple path between two nodes of C_i that violates the geodesic convexity of C_i. So $x \notin C_i$, as claimed. □

Now recall RecoverCluster$(G, \epsilon, \boldsymbol{s}, i)$. Let R_i^ℓ be the value of R_i at the beginning of the ℓ-th iteration, and let (S_i^ℓ, S_{j_ℓ}) be the separator computed by CSeparator at the ℓ-th iteration (note: the first cluster is always i, but the second cluster varies with ℓ).

Lemma 4.19. *If $(x, y) \in \Gamma(R_i^\ell)$, then $(x, y) \in \Gamma(S_i^\tau)$ for some $\tau \in \{1, \ldots, \ell - 1\}$.*

Proof. Let:

$$\tau = \min\left\{1 \leq t \leq \ell - 1 \,:\, (x, y) \in \Gamma(R_i^{t+1})\right\} \tag{4.22}$$

First, we have $x \in S_i^\tau$. Indeed, by construction $R_i^{\tau+1} \subseteq R_i^\tau \cap S_i^\tau$, and we know $x \in R_i^{\tau+1}$. Second, we have $y \notin S_i^\tau$. Suppose indeed by contradiction that $y \in S_i^\tau$. Since $x \in R_i^\tau$, and since $(x, y) \notin \Gamma(R_i^\tau)$, then $y \in R_i^\tau$. Therefore, $y \in S_i^\tau \cap R_i^\tau$. So y would be connected to x in $G[S_i^\tau \cap R_i^\tau]$, and therefore we would have $y \in R_i^{\tau+1}$ as well, by construction of $R_i^{\tau+1}$ as a connected component. But then, $y \in R_i^{\tau+1}$ would imply $(x, y) \notin \Gamma(R_i^{\tau+1})$ which contradicts our hypothesis. Therefore $x \in S_i^\tau$ and $y \notin S_i^\tau$, which implies $(x, y) \in \Gamma(S_i^\tau)$, as claimed. □

of Lemma 4.9. The first bound on the number of queries follows by Lemma 4.3, by observing that $Z \subseteq B(u, \epsilon(\frac{2}{\gamma} + 1)) = B(u, \epsilon\frac{2+\gamma}{\gamma})$. Let us now prove the correctness; the claim on the adapted version will follow afterwards.

First, suppose that $R_i = C_i$. Then $\boldsymbol{s} \cap R_i = \{s_i\}$. Moreover, at each iteration of the loop, by Lemma 4.3 $Z_i = Z$, so $Z \setminus Z_i = \emptyset$. Finally, no edge $(x, y) \in \Gamma(R_i)$ exists such that $d_G(u, x) \geq \frac{2}{\gamma} + 1$ for all $u \in \boldsymbol{u}$. Indeed, if such an edge (x, y) existed, by Lemma 4.19 we would have $(x, y) \in \Gamma(S_i)$ at some previous round of RecoverCluster, which by Lemma 4.18 implies $x \notin C_i$ — a contradiction, since $x \in R_i = C_i$. Hence, FindNewSeed reaches the last line and returns NIL.

Suppose now that $R_i \neq C_i$. If $s_h \in R_i$ for some $s_h \neq s_i$, then FindNewSeed returns s_h at line 1. Otherwise, we must have $C_h \cap R_i \neq \emptyset$ but $C_h \not\subseteq R_i$ for some $h \neq i$. By the connectedness of C_h in G, this implies the existence of an edge $(x, y) \in \Gamma(R_i)$ with $x \in C_h$. If any such edge exists with $d_G(u, x) < \frac{2}{\gamma} + 1$ for some $u \in \boldsymbol{u}$, then line 5 will find such an x and return it. Otherwise, any such edge has $d_G(u, x) \geq \frac{2}{\gamma} + 1$ for all $u \in \boldsymbol{u}$. In this case, line 6 will return some x such that $(x, y) \in \Gamma(R_i)$ and $d_G(u, x) \geq \frac{2}{\gamma} + 1$ for all $u \in \boldsymbol{u}$, which is correct since by Lemma 4.19 and Lemma 4.18 we have $x \notin C_i$.

To make FindNewSeed use at most $k\,\mathcal{M}^*(\frac{\beta\gamma}{2+\gamma})$ queries over the whole execution of RecoverCluster, we keep track of $Z \setminus Z_i$ in the following way. At each invocation, we compute the set $Z^{(u)} = \{x \in R_i : d_G(u, x) < \frac{2}{\gamma} + 1\}$, where u is the last node added to $\boldsymbol{u}$. Then we invoke MBS only on $Z^{(u)}$, obtaining $Z_i^{(u)}$, and we then add $Z^{(u)} \setminus Z_i^{(u)}$ to $Z \setminus Z_i$. Finally, we remove from $Z \setminus Z_i$ all points not in R_i. This sequence of operations costs $\mathcal{M}^*(\frac{\beta\gamma}{2+\gamma})$ SCQ queries, and the resulting set $Z \setminus Z_i$ will be exactly the one computed by FindNewSeed above. Hence the behavior of the algorithm is unchanged, but the total number of queries is at most $k\,\mathcal{M}^*(\frac{\beta\gamma}{2+\gamma})$. □

4.C Supplementary material for Section 4.5

4.C.1 Lemma 4.20

Lemma 4.20. *Let $\mathcal{C}$ be a (β, γ)-convex k-clustering of X (Definition 4.10), and for $i \in \{1, \ldots, k\}$ let:*

$$\zeta_i = \min\{\zeta : \forall\, x, y \in C : d_{G_X(\zeta)}(x, y) < \infty\} \tag{4.23}$$

$$\zeta_i^* = \min\{\zeta : \rho(G_{C_i}(\zeta)) = 1\} \tag{4.24}$$

Then $\zeta_i = \zeta_i^*$, and all properties of Definition 4.10 hold when $\epsilon_i = \zeta_i = \zeta_i^*$.

Proof. First, observe that $\zeta_i \leq \zeta_i^*$, since $\rho(G_{C_i}(\zeta)) = 1$ implies $\forall\, x, y \in C : d_{G_X(\zeta)}(x, y) < \infty$, and thus the minimum in (4.23) is taken over a superset of that of (4.24). Now, consider the graph $G_X(\zeta_i)$. For any two nodes $x, y \in C_i$, since $\zeta_i \leq \zeta_i^*$ and $d_{G_X(\zeta_i)}(x, y) < \infty$, the geodesic convexity implies that any shortest path in $G_X(\zeta_i)$ between x and y lies in C_i. But this means that the subgraph induced by C_i in $G_X(\zeta_i)$ is connected. By definition of ζ_i^* this implies that $\zeta_i^* \leq \zeta_i$. We conclude that $\zeta_i = \zeta_i^*$.

For the second claim, we consider each property in turn when $\epsilon_i = \zeta_i^*$. The connectedness of $G_{C_i}(\zeta_i^*)$ holds by definition of ζ_i^*. Now let ζ be any value such that the three properties hold when $\epsilon_i = \zeta$. Then $\zeta \geq \zeta_i^*$, because for $\epsilon_i < \zeta_i^*$ the connectedness fails, by definition of ζ_i^*. This implies that the local metric margin and the geodesic convexity with margin hold for $\epsilon_i = \zeta_i^*$, since they hold for $\epsilon_i = \zeta \geq \zeta_i^*$. □

4.C.2 Pseudo-code of RecoverClustering2 and proof of Theorem 4.11

Theorem 4.11. *Suppose $\mathcal{C}$ is (β, γ)-convex. Then,* RecoverClustering2$(X, \boldsymbol{\epsilon}, \boldsymbol{s})$ *deterministically returns $\mathcal{C}$, has the same runtime as* RecoverClustering$(X, \boldsymbol{\epsilon}, \boldsymbol{s})$*, and uses the same number of* SCQ *queries as* RecoverClustering$(X, \boldsymbol{\epsilon}, \boldsymbol{s})$*, plus at most $\mathcal{O}(k^2)$* SEED *queries.*

To prove the theorem, we need a technical lemma. It guarantees that, if we take the connected component of the cluster with smallest radius ϵ^* in $G_X(\epsilon^*)$, then the clustering induced by $\mathcal{C}$ on that subgraph is (β, γ)-convex (Definition 4.1) with radius ϵ^*.

Lemma 4.21. *Let $\mathcal{C}$ be a (β, γ)-convex κ-clustering of X (Definition 4.10). Let ϵ^* be the smallest radius of any cluster of $\mathcal{C}$, and let G^* be the connected component of $G_X(\epsilon^*)$ that contains a cluster with radius ϵ^*. Finally, let $X^* = V(G^*)$, and let $\mathcal{C}^* = \big(C_1 \cap X^*, \ldots, C_\kappa \cap X^*\big)$. Then, $\mathcal{C}^*$ is a (β, γ)-convex clustering of X^* with radius ϵ^* (Definition 4.1).*

Proof. We verify that the three properties of Definition 4.1 hold with radius ϵ^*. This implies that ϵ^* is also the smallest such value, since the corresponding cluster becomes disconnected in $G^*(\epsilon)$ for any $\epsilon < \epsilon^*$ — and thus ϵ^* is indeed the radius of the clustering. We define $C_i^* = C_i \cap X^*$ for all $i \in [\kappa]$. Note that the graph on which we verify the properties is $G_{X^*}(\epsilon^*)$, which is precisely G^* by the maximality of X^* as a connected component. Hence, from now on we write G^* for $G_{X^*}(\epsilon^*)$.

Connectivity: the subgraph induced by C_i^* in G^* is connected.
Proof. Consider two points $x, y \in C_i^*$; obviously $x, y \in C_i$. Since G^* is connected, $d_{G^*}(x,y) < \infty$. Moreover, $d_{G^*}(x,y) = d_{G_X(\epsilon^*)}(x,y)$, since by construction G^* is the connected component of $G_X(\epsilon^*)$ containing x and y. Thus, $d_{G_X(\epsilon^*)}(x,y) < \infty$. Moreover, $\epsilon^* \le \epsilon_i$ by assumption. Thus, $x, y \in C_i$, and $d_{G_X(\epsilon^*)}(x,y) < \infty$ with $\epsilon^* \le \epsilon_i$. Then, by the geodesic convexity of $\mathcal{C}$ on X (Definition 4.10), any shortest path π between x and y in $G_X(\epsilon^*)$ lies entirely in C_i. Since again G^* is the connected component of $G_X(\epsilon^*)$ containing x, y, then π must lie in X^*. We conclude that $\pi \in C_i \cap X^* = C_i^*$. This holds for any choice of x, y. Therefore, $G^*[C_i^*]$ is connected.

Local metric margin: for all $x, y \in X^*$, if $x \in C_i^*$ and $y \notin C_i^*$, then $d(x,y) > \beta\epsilon^*$.
Proof. Since $x \in C_i^*$ and $y \notin C_i^*$, then $x \in C_i$ and $y \notin C_i$. By the local metric margin of $\mathcal{C}$, and since $\epsilon_i \ge \epsilon^*$, we have $d(x,y) > \beta\epsilon_i \ge \beta\epsilon^*$.

Geodesic convexity with margin: if $x, y \in C_i^*$, then in G^* any simple path between x and y of length at most $(1+\gamma)d_{G^*}(x,y)$ lies entirely in C_i^*.
Proof. By the same argument of connectivity, we invoke the geodesic convexity (Definition 4.10) for $x, y \in C_i$, for $\epsilon = \epsilon^* \le \epsilon_i$. We obtain that $G_X(\epsilon^*)$ contains no simple path of length at most $(1+\gamma)d_{G_X(\epsilon^*)}(x,y)$ between x and y that leaves C_i. This implies that no such path exists in G^* as well, since $G^* \subseteq G_X(\epsilon^*)$. Moreover, since $C_i^* = C_i \cap X^*$, no such path exists in G^* that leaves C_i^* (otherwise it would leave C_i). Recalling that $(1+\gamma)d_{G_X(\epsilon^*)}(x,y) = (1+\gamma)d_{G^*}(x,y)$, we deduce that in G^* there is no path of length at most $(1+\gamma)d_{G^*}(x,y)$ between x and y that leaves C_i^*. This is the geodesic convexity of C_i^* in G^* (Definition 4.1). □

We can now present the algorithm for recovering (β,γ)-convex clusters with different radii.

Algorithm 11 RecoverClustering2($X, \boldsymbol{\epsilon}, \boldsymbol{s}$)

```
1: assume $\epsilon_1 \le \ldots \le \epsilon_k$
2: for $i = 1, \ldots, k$ do
3:     $G^* :=$ the connected component of $s_i$ in $G_X(\epsilon_i)$,   $X^* := V(G^*)$
4:     for $j = i+1, \ldots, k$ do$s_j^* :=$ SEED$(X^*, j)$
5:         $s_j^* :=$ SEED$(X^*, j)$;
6:     $\boldsymbol{s}^* := (s_i, s_{i+1}^*, \ldots, s_k^*)$
7:     $\hat{C}_i :=$ RecoverCluster$(G^*, \epsilon_i, \boldsymbol{s}^*, i)$
8:     output $\hat{C}_i$
9:     $X := X \setminus \hat{C}_i$
```

of Theorem 4.11. For each $i = 1, \ldots, k$ let X_i be the value of X at the beginning of the i-th iteration of RecoverClustering2($X, \boldsymbol{\epsilon}, \boldsymbol{s}$). We show that the following three invariants holds:

1. $X_i = C_i \cup \ldots \cup C_k$
2. $(C_i, \ldots, C_k)$ is a (β,γ)-convex clustering of X_i (Definition 4.10)
3. if $i > 1$ then the algorithm has output $C_1, \ldots, C_{i-1}$ so far

When $i = 1$ we have $X_i = X$, and all invariants clearly hold. Now assume that they hold at the beginning of the i-th iteration for some $i \ge 1$. We will show that the algorithm sets $\hat{C}_i = C_i$. This will imply that the three invariants hold at iteration $i+1$ as well. For the first and third invariant, this is trivial. For the second, simply observe that deleting a cluster never invalidates the three properties of Definition 4.10; thus, $(C_{i+1}, \ldots, C_k)$ will be a (β,γ)-convex clustering of $X_{i+1} = C_{i+1} \cup \ldots \cup C_k$.

Thus, we prove that $\hat{C}_i = C_i$. To this end, consider the subgraph G^* and its node set V^* computed at line 3. Let $\mathcal{C}_i = (C_i, \dots, C_k)$, and let $\mathcal{C}_i^* = (C_i \cap X^*, \dots, C_k \cap X^*)$. Since ϵ_i is the smallest radius of all clusters in $\mathcal{C}_i$, then by Lemma 4.21 with $\epsilon^* = \epsilon_i$, $\mathcal{C}_i^*$ is a (β, γ)-convex clustering for X_i with radius ϵ_i (Definition 4.1). Furthermore, by construction $\boldsymbol{s}^*$ contains one seed for each nonempty cluster in $\mathcal{C}^*$. Therefore, by Lemma 4.8, RecoverCluster$(G^*, \epsilon, \boldsymbol{s}^*, i)$ returns C_i, so $\hat{C}_i = C_i$.

The bound on the number of queries is straightforward. □

4.D Supplementary material for Section 4.6

4.D.1 GetEpsilons and proof of Theorem 4.12

Theorem 4.12. *Suppose $\mathcal{C}$ is (β, γ)-convex. Then, the cluster radii $\epsilon_1, \dots, \epsilon_k$ can be learned using $\mathcal{O}(k \log n)$ SEED queries in time $\mathcal{O}(m\, \alpha(m,n) + kn \log n)$, where $\alpha(m,n)$ is the functional inverse of the Ackermann function.*[9]

We start with a simple routine for testing the connectedness of a cluster using SEED queries.

Algorithm 12 IsConnected(G, i)

```
1: $u := \text{SEED}(V(G), i)$
2: $U :=$ the connected component of $u$ in $G$
3: return $(\text{SEED}(V(G) \setminus U, i) = \text{NIL})$;
```

Claim 4.1. *If $V(G) \cap C_i \neq \emptyset$, then* IsConnected$(G, i)$ *uses two* SEED *queries and returns* TRUE *if and only if $d_G(x, y) < \infty$ for all $x, y \in C_i$.*

Let T be a minimum spanning tree of the weighted graph $\mathcal{G}$. For any $\epsilon > 0$, let $T(\epsilon)$ be the forest obtained by keeping only the edges (x, y) of T such that $d(x, y) \leq \epsilon$. Recall the following basic fact:

Claim 4.2. *The connected components of $T(\epsilon)$ are the connected components of $G_X(\epsilon)$.*

As a consequence, we have:

Claim 4.3. IsConnected$(T(\epsilon), i) =$ IsConnected$(G_X(\epsilon), i)$*, for any $i \in [k]$ and any $\epsilon > 0$.*

We introduce the algorithm for learning the radius of a single cluster.

Algorithm 13 GetEpsilon(T, i)

```
1: $\boldsymbol{w} := (w_0, w_1, \dots, w_\ell)$, the distinct edge weights of $T$ in increasing order, with $w_0 = 0$
2: $lo := 0$,    $hi := \ell$
3: while $w_{lo} < w_{hi}$ do
4:     $mid := \lfloor \frac{lo+hi}{2} \rfloor$
5:     if IsConnected$(T(w_{mid}), i)$ then $hi := mid$ else $lo := mid + 1$
6: return $w_{hi}$
```

Lemma 4.22. *If T is a MST of $\mathcal{G} = (X, \mathcal{E}, d)$, then* GetEpsilon$(T, i)$ *returns ϵ_i in time $\mathcal{O}(n \log n)$ using $\mathcal{O}(\log n)$* SEED *queries.*

Proof. It is straightforward to see that the algorithm stops within $\mathcal{O}(\log n)$ iterations, since $\boldsymbol{w}$ has at most $m = \mathcal{O}(n^2)$ entries and $(hi - lo)$ decreases by a constant factor at each iteration. For the running time, since T has $\mathcal{O}(n)$ edges, every call to IsConnected$(T(w_{mid}), i)$ takes time $\mathcal{O}(n)$. This gives the time bound of $\mathcal{O}(n \log n)$.

Now we show that the algorithm returns ϵ_i. By Lemma 4.20, this is equivalent to prove that the algorithm returns $w^* = \min\{w \in \boldsymbol{w} : C_i \text{ is connected in } G_X(w)\}$. Consider the

[9]For all practical purposes, α can be considered constant. For instance, $\alpha(m, m) \leq 4$ for all $m \leq \frac{1}{8} 2^{2^{2^{65536}}}$.

beginning of a generic iteration, when the test $w_{lo} < w_{hi}$ is performed. We claim that $w^* \leq w_{hi}$. To this end, observe that C_i is connected in $G_X(w_{hi})$. This is true since it holds at the beginning of the first iteration, when $w_{hi} = w_\ell$, and because at each iteration hi is set to mid only if IsConnected$(T(w_{mid}), i) =$ TRUE. We now claim that $w^* \geq w_{lo}$. This holds since $w^* \geq w_0 = 0$ at the first iteration, and because at each iteration lo is set to $mid + 1$ only if IsConnected$(T(w_{mid}), i) =$ FALSE. Therefore, when the algorithm stops, we have $w_{lo} = w_{hi} = w^*$, as claimed. □

We conclude with the algorithm to learn all the radii. We denote by MST(m) the time needed for computing the MST of a connected graph (note that we can always assume $\mathcal{G}$ is connected, otherwise we can just compute its connected components in time $\mathcal{O}(m)$ and use each one of them in turn). It is known that MST$(m) = O(m\,\alpha(m, m))$, where $\alpha(m, m)$ is the classic functional inverse of Ackermann's function [Chazelle, 2000]. Lemma 4.23 below follows immediately from these observations.

Algorithm 14 GetEpsilons$(\mathcal{G} = (X, \mathcal{E}, d), k)$

```
1: $T :=$ MST$(\mathcal{G})$
2: for $i = 1, \ldots, k$ do
3:     $\hat{\epsilon_i} :=$ GetEpsilon$(T, i)$
4: return $\hat{\epsilon_1}, \ldots, \hat{\epsilon_k}$
```

Lemma 4.23. GetEpsilons$(\mathcal{G}, k)$ *returns the radii* $\epsilon_1, \ldots, \epsilon_k$ *in time* $\mathcal{O}(m\,\alpha(m, n) + kn \log n)$ *using* $\mathcal{O}(k \log n)$ SEED *queries.*

4.D.2 Proof of Theorem 4.13

Theorem 4.13. *Suppose* $\mathcal{C}$ *is* (β, γ)*-convex, and let* $R = \log(\frac{4}{\beta\gamma})\,\text{dens}(X)$*. If only one between* β *and* γ *is unknown, then we can recover* $\mathcal{C}$ *with a multiplicative overhead of* $\mathcal{O}(R)$ *in both query cost and running time, plus* $\mathcal{O}(k^2 R)$ SCQ *queries and* $\mathcal{O}(kR)$ SEED *queries. This applies to each one of our algorithms (i.e., with radii that are identical or not, known or unknown).*

Proof. Suppose first β is unknown and γ is known. Recall that $\beta \leq 1$. We make a succession of guesses $\hat{\beta} = 2^{-j}$ for $j = 0, 1, \ldots$. For each guess, we run our algorithm with $\beta = \hat{\beta}$ and look at the output clustering $\hat{\mathcal{C}}$. Clearly, if $\mathcal{C}$ is (β, γ)-convex, then $\mathcal{C}$ is $(\hat{\beta}, \gamma)$-convex for any $\hat{\beta} \leq \beta$ as well. Thus, as soon as $\hat{\beta} \leq \beta$, our algorithm will return $\hat{\mathcal{C}} = \mathcal{C}$. So, after each run, we need only to check whether $\hat{\mathcal{C}} = \mathcal{C}$, and stop in the affirmative case.

To check whether $\hat{\mathcal{C}} = \mathcal{C}$, we do as follows. First, we check if $|\hat{\mathcal{C}}| \neq |\mathcal{C}|$. If this is the case, then the only possibility for $\hat{\mathcal{C}} \neq \mathcal{C}$ is that some cluster C_i intersects both $\hat{C}$ and $X \setminus \hat{C}$, for some cluster $\hat{C} \in \hat{\mathcal{C}}$. Therefore, we take each cluster $\hat{C} \in \hat{\mathcal{C}}$ in turn. We then take any node $x \in \hat{C}$, and we learn the label of i with $\mathcal{O}(k)$ SCQ queries. Then, we invoke SEED$(X \setminus \hat{C}, i)$. If we get a node in return, we know that C_i has points in $\hat{C}$ and $X \setminus \hat{C}$, and therefore $\hat{\mathcal{C}} \neq \mathcal{C}$. Otherwise, we continue to the next cluster. If the outputs of the SEED are all NIL, then we deduce that $\hat{\mathcal{C}} = \mathcal{C}$.

The process will stop with $\hat{\beta} \geq \frac{1}{2}\beta$, which happens after $R = \mathcal{O}(\log \mathcal{M}^*(\frac{\beta\gamma}{2}))$ rounds. Since $\mathcal{M}^*(\frac{\beta\gamma}{2}) \leq \left(\frac{4}{\beta\gamma}\right)^{\text{dens}(X)}$, see Section 4.3, then $\log \mathcal{M}^*\left(\frac{\beta\gamma}{2}\right) = \mathcal{O}(\text{dens}(X) \log\left(\frac{4}{\beta\gamma}\right))$. At each round, the algorithm uses k^2 SCQ queries plus $2k$ SEED queries. Thus, in total we use $\mathcal{O}(k^2 R)$ SCQ queries and $\mathcal{O}(kR)$ SEED queries. The case with γ is unknown and β is known is symmetric. □

4.E Supplementary material for Section 4.7

4.E.1 Running time with identical radii

We prove:

Theorem 4.24. RecoverClustering$(X, \epsilon, \boldsymbol{s})$ *runs in time* $\mathcal{O}(k^2(n + m))$.

Proof. First, recall from Section 4.7 that computing $G = G_X(\epsilon)$ takes time $\mathcal{O}(n+m)$. Then, each call to RecoverCluster$(G, \epsilon, \boldsymbol{s}, i)$ takes time $\mathcal{O}(k(n+m))$ by Lemma 4.29. Since there are k clusters, the claim follows. $\square$

In the rest of this appendix we prove Lemma 4.29, through a sequence of intermediate steps.

Lemma 4.25. MBS(Z, ϵ, u_i) *runs in time* $\mathcal{O}(n+m)$.

Proof. First, we construct $G(\beta\epsilon)$ by thresholding G, which takes time $\mathcal{O}(n+m)$. Then, we keep only the edges of $G(\beta\epsilon)$ which have both endpoints in Z, which takes again time $\mathcal{O}(n+m)$. Once we have $G_Z(\beta\epsilon)$, listing its connected components takes once again time $\mathcal{O}(n+m)$. $\square$

Lemma 4.26. *Given a simple C_i-prefixed path π,* FindCutEdge(π) *runs in time* $\mathcal{O}(\log n)$.

Proof. Straightforward, see Observation 4.1 and the code of FindCutEdge. $\square$

Lemma 4.27. CSeparator(G, u_i, u_j) *runs in time* $\mathcal{O}(n+m)$.

Proof. Computing $d_G(u_i, x)$ and $d_G(u_j, x)$ for all $x \in G$ takes time $\mathcal{O}(n+m)$ using a BFS from u_i and u_j. Thereafter, we can compute Z in time $\mathcal{O}(n)$. Running MBS(Z, ϵ, u_i) takes time $\mathcal{O}(n+m)$, see above. Finally, the loop at line 4 takes time $\mathcal{O}(n)$. $\square$

Lemma 4.28. *In* RecoverCluster$(G, \epsilon, \boldsymbol{s}, i)$*, each call to* FindNewSeed$(G, R_i, \epsilon, \boldsymbol{s}, \boldsymbol{u}, i)$ *takes time $\mathcal{O}(k(n+m))$. By adapting both algorithms, this can be reduced to $\mathcal{O}(n+m)$ while adding at most an additive $\mathcal{O}(n+m)$ to the running time of each iteration of* RecoverCluster.

Proof. Let us start with the $\mathcal{O}(k(n+m))$ bound given by a "naive" implementation of FindNewSeed. At line 1, we compute $\boldsymbol{s} \cap R_i$ in time $\mathcal{O}(k|R_i|) = \mathcal{O}(kn)$ and perform the check in constant time. At line 2, we make $|\boldsymbol{u}| \leq k$ iterations. At each iteration we compute Z in time $\mathcal{O}(n+m)$ with a BFS from u, then we run MBS(Z, ϵ, u) in time $\mathcal{O}(n+m)$ by Lemma 4.25, and possibly we search for $x \in Z \setminus Z_i$ which takes time $\mathcal{O}(n)$. Hence the entire loop of line 2 takes time $\mathcal{O}(k(n+m))$. Finally, at line 6 we compute the set $\{(x, y) \in \Gamma(R_i) : \forall u \in \boldsymbol{u} : d_G(u, x) \geq \frac{2}{\gamma} + 1\}$. To this end, we compute the set $\{x \in R_i : \forall u \in \boldsymbol{u} : d_G(u, x) \geq \frac{2}{\gamma} + 1\}$, which takes time $\mathcal{O}(k(n+m))$ using a BFS from u. For each such x in this set, we list all its edges $(x, y) \in E(G)$. If we find any such edge with $y \notin R_i$, we return y, else we return NIL. Thus, this part takes $\mathcal{O}(k(n+m))$. Therefore, a single call to FindNewSeed takes time $\mathcal{O}(k(n+m))$. Since FindNewSeed is called at most k times, this gives a total running time of $\mathcal{O}(k^2(n+m))$.

Let us now see how to reduce to $\mathcal{O}(n+m)$ the running time of FindNewSeed, by adding at most $\mathcal{O}(n+m)$ to each iteration of RecoverCluster. First, consider line 1 of FindNewSeed. We keep $\boldsymbol{s}$ updated so as to ensure that $\boldsymbol{s} \cap R_i = \boldsymbol{s} \setminus \{s_i\}$. In this way, we can run line 1 in constant time. Towards this end, we modify RecoverCluster as follows. First, we store s_i separately form $\boldsymbol{s}$ in a dedicated variable. Second, after updating G_i and R_i at line 10 of RecoverCluster, we replace $\boldsymbol{s}$ with $\boldsymbol{s} \cap R_i$. This is done by taking an empty dictionary $\boldsymbol{s}'$, taking every node $x \in R_i$, and adding x to $\boldsymbol{s}'$ if $x \in \boldsymbol{s}$. The whole operation takes time $\mathcal{O}(n)$ by using dictionaries with $O(1)$ time per lookup and update. Summarizing, we spend an additional $\mathcal{O}(n)$ time at each iteration of RecoverCluster, and line 1 of FindNewSeed will run in constant time.

Now consider the loop at line 2 of FindNewSeed. We modify RecoverCluster so as to keep track of the set of nodes:

$$\overline{Z(\boldsymbol{u})} = \{x \in R_i \setminus C_i : \exists u \in \boldsymbol{u} : d_G(u, x) < \frac{2}{\gamma} + 1\}$$

Note that line 2 of FindNewSeed detects precisely if $\overline{Z(\boldsymbol{u})} \neq \emptyset$, in which case it returns any $x \in \overline{Z(\boldsymbol{u})}$. Thus, if we have $\overline{Z(\boldsymbol{u})}$, we can replace the entire block at line 2 with an equivalent block that runs in time $\mathcal{O}(1)$. To keep track of $\overline{Z(\boldsymbol{u})}$, we initialize it to an empty set, using a dictionary with $O(1)$ lookup and access time. Then, after updating G_i and R_i at line 10, we perform the following operations. First, we make RecoverCluster compute $Z = Z(u_i)$. Second, we run MBS(Z, ϵ, u_i) to obtain $Z_i(u_i)$. Third, we compute $\overline{Z_i(u_i)} := Z(u_i) \setminus Z_i(u_i)$.

Fourth, we add $\overline{Z_i(u_i)}$ to $\overline{Z(\boldsymbol{u})}$. Finally, we keep in $\overline{Z(\boldsymbol{u})}$ only those $x \in \overline{Z(\boldsymbol{u})}$ such that $x \in R_i$. This can be done by creating a new dictionary, adding to it each $x \in R_i$ such that $x \in \overline{Z(\boldsymbol{u})}$, and overwriting $\overline{Z(\boldsymbol{u})}$ with that dictionary, which requires time $\mathcal{O}(n)$ in total. Therefore, we add $\mathcal{O}(n+m)$ to each iteration of RecoverCluster, and line 2 of FindNewSeed will take time $\mathcal{O}(n)$.

Finally, we have line 6 of FindNewSeed. Here, we want to perform the check in time $\mathcal{O}(n+m)$. To this end, at the beginning of the first iteration of RecoverCluster, we mark all nodes of R_i as *active*. Then, at each iteration, after RecoverCluster has computed (u_i, u_j), for all $x \in R_i$ we compute $d_G(u_i, x)$ and if $d_G(u_i, x) < \frac{2}{\gamma} + 1$ then we change the mark of x to *inactive*. This takes time $\mathcal{O}(n+m)$, using a BFS from u_i. Then, at line 6 of FindNewSeed, we only need to sweep over all $x \in R_i$ and, if x is active, list its edges (x, y) until finding $y \notin R_i$ (a check which takes again time $O(1)$ by storing R_i as a dictionary). This gives a total time bound of $\mathcal{O}(n+m)$ for the block at line 6, and once again we add only a $\mathcal{O}(n+m)$ to each iteration of RecoverCluster.

The proof is complete. □

Lemma 4.29. RecoverCluster$(G, \epsilon, \boldsymbol{s}, i)$ *runs in time* $\mathcal{O}(k(n+m))$.

Proof. Computing G_i and R_i at any point along the algorithm takes time $\mathcal{O}(n+m)$. Consider each iteration of the loop. By Lemma 4.28, we can make FindNewSeed$(G, R_i, \epsilon, \boldsymbol{s}, \boldsymbol{u}, i)$ run in time $\mathcal{O}(n+m)$ while increasing the overall running time of the iteration of RecoverCluster by $\mathcal{O}(n+m)$. ShortestPath$(G[R_i], s_i, s_h)$ runs in time $\mathcal{O}(n+m)$, as it is simply a BFS on $G[R_i]$. FindCutEdge$(\pi(s_i, s_h))$ runs in time $\mathcal{O}(\log n)$, see Observation 4.26. Adding u_i to $\boldsymbol{u}$ takes constant time. CSeparator(G, u_i, u_j) runs in time $\mathcal{O}(n+m)$, see Lemma 4.27. Therefore each iteration of RecoverCluster takes time $\mathcal{O}(n+m)$. At most k iterations are made, concluding the proof. □

4.E.2 Running time with different radii

Theorem 4.30. RecoverClustering2$(X, \boldsymbol{\epsilon}, \boldsymbol{s})$ *runs in time* $\mathcal{O}(k^2(n+m))$.

Proof. Let us consider each one of the k iterations of the algorithm. To compute G^*, we perform a BFS by ignoring any edge $(x, y) \in \mathcal{G}$ with $d(x, y) > \epsilon^*$. The resulting runtime is $\mathcal{O}(n+m)$. The SEED part takes time $\mathcal{O}(k) = \mathcal{O}(n)$, as does the construction of $\boldsymbol{s}^*$. The call to RecoverCluster takes time $\mathcal{O}(k(n+m))$ by Lemma 4.29. Writing $\hat{C}_i$ in the output takes time $\mathcal{O}(n)$. Summing over all iterations gives the bound. □

4.F Supplementary material for Section 2.7

4.F.1 Proof of Theorem 4.14

Theorem 4.14 (Dependence on dens(X).)**.** *Choose any* $\beta, \gamma \in (0, 1)$*. There is a distribution of* (β, γ)*-convex* 2*-clusterings* $\mathcal{C}$ *(Definition 4.1 or Definition 4.10), where* $n = |X| = 2^{\text{dens}(X)}$ *is arbitrarily large, such that any algorithm (even randomized) needs* $\Omega(2^{\text{dens}(X)})$ SCQ *and/or* SEED *queries to recover* $\mathcal{C}$ *with constant probability. This holds even if* β, γ, ϵ *are known.*

Proof. Let $\mathcal{G} = (X, \mathcal{E}, d)$ where $(X, \mathcal{E})$ is the complete graph on n nodes and $d = 1$, and let $\mathcal{C} = (C_1, C_2)$ be a uniform random partition of X. We claim that any such $\mathcal{C}$ is (β, γ)-convex according to both Definition 4.1 and Definition 4.10. Take indeed $\epsilon = 1$ and let $G = G_X(\epsilon)$. The connectivity of $G[C_1]$ and $G[C_2]$ holds trivially (they are complete graphs). The local metric margin holds as well, since any two distinct points $x, y \in X$ satisfy $d(x, y) = d > \beta d$, as $\beta < 1$. To see that geodesic convexity holds, too, note that for any $x, y \in C_1$ we have $d_G(x, y) \leq 1$ and any (simple) path between x and y that contains a point in $X \setminus C_1$ has length at least $2 > (1+\gamma) d_G(x, y)$. Finally, note that $G_X(\epsilon')$ is an independent set for any $\epsilon' < \epsilon$, proving that $\mathcal{C}$ is (β, γ)-convex according to Definition 4.10 as well.

Now, since $\mathcal{C}$ is chosen uniformly at random among all partitions of X, one can see that $\Omega(n)$ SCQ or SEED queries are necessary to recover $\mathcal{C}$ with constant probability. To see this, note that as long as x has not been returned by some SEED query or has not bee queried via SCQ, then x belongs to one of C_1 and C_2 with equal probability. Finally,

$\mathrm{dens}(X) = \log_2 \mu(X) \leq \log_2 n$ since $\mu(X) \leq |X|$. Thus $n = 2^{\mathrm{dens}(X)}$, which proves the thesis. □

4.F.2 Proof of Theorem 4.15

Theorem 4.15 (Necessity of seeds.)**.** *Choose any $\beta, \gamma \in (0, 1]$. There is a distribution of (β, γ)-convex 2-clusterings $\mathcal{C}$ (Definition 4.1 or Definition 4.10), where $X \subseteq \mathbb{R}^2$ with $n = |X|$ arbitrarily large and d the Euclidean distance, such that any algorithm (even randomized) needs $\Omega(n)$ SCQ queries to recover $\mathcal{C}$ with constant probability if no seed nodes are given. This holds even if γ, α, ϵ are known.*

Proof. Let $\mathcal{G} = (X, \mathcal{E}, d)$ where $(X, \mathcal{E})$ is the complete graph and $d(x, y)$ is the Euclidean distance in $\mathbb{R}^2$. We let $X = \mathrm{UP} \cup \mathrm{LOW}$, see Figure 4.F.1, where:

$$\mathrm{UP} = \bigcup_{j=1}^{n/3} \{(2j, 1)\}, \quad \mathrm{LOW} = \bigcup_{j=1}^{2n/3} \{(j, 0)\}$$

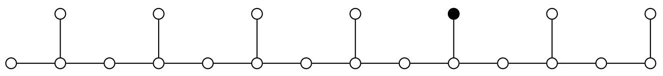


FIGURE 4.F.1: the graph $G_\epsilon(X)$ for $\epsilon = 1$. All points are in C_1, save for the filled point in C_2, chosen uniformly at random in UP.

Now choose a point z uniformly at random from UP, and set $C_1 = X \setminus \{z\}$ and $C_2 = \{z\}$. One can check that all properties of Definition 4.1 and Definition 4.10 are satisfied for $\epsilon = 1$. In particular, in $G_X(\epsilon)$, no simple path between two points of C_1 contains z. Hence, $\mathcal{C}$ is (β, γ)-convex. Clearly, $\Omega(n)$ queries are needed to find z (which is equivalent to recovering $\mathcal{C}$) with constant probability, even if γ, α and the ϵ are known. □

4.F.3 Proof of Theorem 4.16

Theorem 4.16 (Cost of learning the radii.)**.** *For any $k \geq 2$, and any sufficiently large n, there exists a distribution of $(1/2, 1)$-convex k-clusterings (Definition 4.1 or Definition 4.10) on n points such that any algorithm (even randomized) needs $\Omega(k \log \frac{n}{k})$ SEED and/or SCQ queries to learn the radii of all clusters with constant probability.*

Proof. We start with the simpler case $k = 2$. Let $\mathcal{G} = (X, \mathcal{E}, d)$ be a path with increasing edge weights, that is:

$$X = [n] \tag{4.25}$$
$$\mathcal{E} = \{(j, j+1) : j = 1, \ldots, n-1\} \tag{4.26}$$
$$d(j, j+1) = 1 + \frac{\beta j}{n}, \quad (j, j+1) \in \mathcal{E} \tag{4.27}$$

Choose j^* uniformly at random in $\{2, \ldots, n-1\}$. We let $C_1 = \{1, \ldots, j^*\}$, and $C_2 = \{j^*, \ldots, n\}$.

First, we prove that C_1 and C_2 are (β, γ)-convex with radii respectively $\epsilon_1 = d(j^* - 1, j^*)$ and $\epsilon_2 = d(n-1, n)$. Recall Definition 4.1. For the connectivity, clearly $\rho(G_{C_1}(\epsilon_1)) = \rho(G_{C_2}(\epsilon_2)) = 1$. For the local metric margin, note that, by the choice of β and d, we have $d \geq 1$ but $\epsilon_1, \epsilon_2 \leq \frac{3}{2}$. Thus, for any two distinct $x, y \in X$, we have $d(x, y) > \beta \max(\epsilon_1, \epsilon_2)$. Therefore the local metric margin is satisfied. For geodesic convexity, note that there is only one edge between C_1 and C_2 in $\mathcal{G}$, thus no simple path can exist between two points of one cluster that intersects the other cluster. Thus, the properties of Definition 4.1 are satisfied by ϵ_1, ϵ_2.

To show that Definition 4.10 is satisfied as well, we have to prove that ϵ_1, ϵ_2 are the smallest such values; by Lemma 4.20 this implies that ϵ_1 and ϵ_2 are the radii of respectively C_1 and C_2. To this end, simply note that $\epsilon_1 = \min\{\zeta : \rho(G_{C_1}(\zeta)) = 1\}$, since $d(j^* - 1, j^*) = \epsilon_1$

and $d(j, j+1) \leq \epsilon_1$ for all $j = 1, \dots, j^* - 1$. Similarly, $\epsilon_2 = \min\{\zeta : \rho(G_{C_2}(\zeta)) = 1\}$, since $d(n-1, n) = \epsilon_2$ and $d(j, j+1) \leq \epsilon_1$ for all $j = j^*, \dots, n-1$.

Finally, we prove that any algorithm needs $\Omega(\log n)$ queries to learn ϵ_1. Clearly, if the algorithm learns ϵ_1 then it can also output the index j^*, which is a function of ϵ_1. Therefore, we show that finding j^* requires $\Omega(\log n)$ queries.

First, we show that SEED is as powerful as SCQ. Consider any set of points $U \subseteq X$. Recall that, when $U \cap C_i \neq \emptyset$, SEED$(U, i)$ is allowed to return *any* node in $U \cap C_i$. Therefore, we let SEED(U, i) return $\min(U \cap C_i)$ if $i = 1$, and $\max(U \cap C_i)$ if $i = 2$. Now, observe that $\min(U \cap C_1) = \min U$ and $\max(U \cap C_1) = \max U$. Therefore, the output of SEED(U, i) can be emulated using SCQ. If $i = 1$, then we run SCQ$(\min(U), 1)$; if the response is $+1$, then we return $\min(U)$, else we return NIL. For $i = 2$, we do the same, but using $\max(U)$.

Therefore, SEED and SCQ are equivalent in this case. Since each call to SCQ reveals at most one bit of information, and j^* is chosen uniformly at random in a set of cardinality $\Omega(n)$, we need $\Omega(\log n)$ in order to learn j^* with constant probability.

In order to extend the construction to any $k \geq 2$, simply take $K = \frac{k}{2}$ disjoint weighted paths on $\frac{n}{K}$ nodes each (without loss of generality we can assume k is even). Each such path is weighted as in the construction above, with the weights of the h-th path all smaller than the weights of the $(h+1)$-th path. For each path, we draw j^* uniformly at random like above, and form two clusters. The same proof used above shows that, to learn the radii of all clusters with constant probability, any algorithm uses at least $\Omega(k \log \frac{n}{k})$ queries. □

Chapter 5

Correlation Clustering with Adaptive Similarity Queries

The scope of this chapter is to study the Correlation Clustering problem from the point of view of query-based algorithms. We begin with a simple and efficient algorithm that obtain a 3-factor approximation, plus an additive error term that depends on the query budget. We show that this algorithm is essentially optimal in terms of the additive (query-dependent) error. Second, we show that a slight modification of the algorithm obtains cluster recovery guarantees. Third and finally, we prove information-theoretical bounds on the number of queries necessary to guarantee a prescribed additive error.

5.1 Introduction

Clustering is a central problem in unsupervised learning. A clustering problem is typically represented by a set of elements together with a notion of similarity (or dissimilarity) between them. When the elements are points in a metric space, dissimilarity can be measured via a distance function. In more general settings, when the elements to be clustered are members of an abstract set V, similarity is defined by an arbitrary symmetric function σ defined on pairs of distinct elements in V. Correlation Clustering (CC) Bansal et al. [2004] is a well-known special case where σ is a $\{-1, +1\}$-valued function establishing whether any two distinct elements of V are similar or not. The objective of CC is to cluster the points in V so to maximize the correlation with σ. More precisely, CC seeks a clustering minimizing the number of errors, where an error is given by any pair of elements having similarity -1 and belonging to the same cluster, or having similarity $+1$ and belonging to different clusters. Importantly, there are no a priori limitations on the number of clusters or their sizes: all partitions of V, including the trivial ones, are valid. Given V and σ, the error achieved by an optimal clustering is known as the *Correlation Clustering index*, denoted by OPT. A convenient way of representing σ is through a graph $G = (V, E)$ where $\{u, v\} \in E$ iff $\sigma(u, v) = +1$. Note that $\mathrm{OPT} = 0$ is equivalent to a perfectly clusterable graph (i.e., G is the union of disjoint cliques). Since its introduction, CC has attracted a lot of interest in the machine learning community, and has found numerous applications in entity resolution Getoor and Machanavajjhala [2012], image analysis Kim et al. [2011], and social media analysis Tang et al. [2016]. Known problems in data integration Cohen and Richman [2002] and biology Ben-Dor et al. [1999] can be cast into the framework of CC Wirth [2010].

From a machine learning viewpoint, we are interested in settings when the similarity function σ is not available beforehand, and the algorithm must learn σ by querying for its value on pairs of objects. This setting is motivated by scenarios in which the similarity information is costly to obtain. For example, in entity resolution, disambiguating between two entities may require invoking the user's help. Similarly, deciding if two documents are similar may require a complex computation, and possibly the interaction with human experts. In these active learning settings, the learner's goal is to trade the clustering error against the number of queries. Hence, the fundamental question is: how many queries are needed to achieve a specified clustering error? Or, in other terms, how close can we get to OPT, under a prescribed query budget Q?

5.1.1 Contributions

In this cahpter we characterize the trade-off between the number Q of queries and the clustering error on n points. The table below here summarizes our bounds in the context of previous work. Running time and upper/lower bounds on the expected clustering error are expressed in terms of the number of queries Q, and all our upper bounds assume $Q = \Omega(n)$ while our lower bounds assume $Q = \mathcal{O}(n^2)$.

Expected clustering error	Reference
$3(\ln n + 1)\mathrm{OPT} + \mathcal{O}\big(n^{5/2}/\sqrt{Q}\big)$	Cesa-Bianchi et al. [2012]
$3\mathrm{OPT} + \mathcal{O}(n^3/Q)$	Theorem 5.2 (see also Bonchi et al. [2013])
$\mathrm{OPT} + \mathcal{O}\big(n^{5/2}/\sqrt{Q}\big)$	Theorem 5.10
$\widetilde{\mathcal{O}}\big(n^3/Q\big)$	Theorem 5.10
$\Omega\big(n^2/\sqrt{Q}\big)$	Theorem 5.11
$\mathrm{OPT} + \Omega\big(n^3/Q\big)$	Theorem 5.12

Our first set of contributions is algorithmic. We take inspiration from an existing greedy algorithm, KwikCluster Ailon et al. [2008], that has expected error 3OPT but a vacuous $\mathcal{O}(n^2)$ worst-case bound on the number of queries. We propose a variant of KwikCluster, called ACC, for which we prove several desirable properties. First, ACC achieves expected clustering error $3\mathrm{OPT} + \mathcal{O}(n^3/Q)$, where $Q = \Omega(n)$ is a deterministic bound on the number of queries. In particular, if ACC is run with $Q = \binom{n}{2}$, then it becomes exactly equivalent to KwikCluster. Second, ACC recovers adversarially perturbed latent clusters. More precisely, if the input contains a cluster C obtained from a clique by adversarially perturbing a fraction ϵ of its edges (internal to the clique or leaving the clique), then ACC returns a cluster $\hat{C}$ such that $\mathbb{E}\big[|C \oplus \hat{C}|\big] = \mathcal{O}\big(\varepsilon|C| + n^2/Q\big)$, where $\oplus$ denotes symmetric difference. This means that ACC recovers almost completely all perturbed clusters that are large enough to be "seen" with Q queries. We also show, under stronger assumptions, that via independent executions of ACC one can recover exactly all large clusters with high probability. Third, we show a variant of ACC, called ACCESS (for Early Stopping Strategy), that makes significantly less queries on some graphs. For example, when $\mathrm{OPT} = 0$ and there are $\Omega\big(n^3/Q\big)$ similar pairs, the expected number of queries made by ACCESS is only the square root of the queries made by ACC. In exchange, ACCESS makes at most Q queries in expectation rather than deterministically.

Our second set of contributions is a nearly complete information-theoretic characterization of the query vs. clustering error trade-off (thus, ignoring computational efficiency). Using VC theory, we prove that for all $Q = \Omega(n)$ the strategy of minimizing disagreements on a random subset of pairs achieves, with high probability, clustering error bounded by $\mathrm{OPT} + \mathcal{O}\big(n^{5/2}/\sqrt{Q}\big)$, which reduces to $\widetilde{\mathcal{O}}\big(n^3/Q\big)$ when $\mathrm{OPT} = 0$. The VC theory approach can be applied to any efficient approximation algorithm, too. The catch is that the approximation algorithm cannot ask the similarity of arbitrary pairs, but only of pairs included in the random sample of edges. The best known approximation factor in this case is $3(\ln n + 1)$ Demaine et al. [2006], which gives a clustering error bound of $3(\ln n + 1)\mathrm{OPT} + \mathcal{O}\big(n^{5/2}/\sqrt{Q}\big)$ with high probability. This was already observed in Cesa-Bianchi et al. [2012] albeit in a slightly different context.

We complement our upper bounds by developing two information-theoretic lower bounds; these lower bounds apply to any algorithm issuing $Q = \mathcal{O}(n^2)$ queries, possibly chosen in an adaptive way. For the general case, we show that any algorithm must suffer an expected clustering error of at least $\mathrm{OPT} + \Omega\big(n^3/Q\big)$. In particular, for $Q = \Theta(n^2)$ any algorithm still suffers an additive error of order n, and for $Q = \Omega(n)$ our algorithm ACC is essentially optimal in its additive error term. For the special case $\mathrm{OPT} = 0$, we show a lower bound $\Omega\big(n^2/\sqrt{Q}\big)$.

Finally, we evaluate our algorithms empirically on real-world and synthetic datasets.

5.2 Related work

Minimizing the correlation clustering error is APX-hard Charikar et al. [2005], and the best efficient algorithm found so far achieves 2.06 OPT Chawla et al. [2015]. This almost matches the best possible approximation factor 2 achievable via the natural LP relaxation of the problem Charikar et al. [2005]. A very simple and elegant algorithm for approximating CC is KwikCluster Ailon et al. [2008]. At each round, KwikCluster draws a random pivot π_r from V, queries the similarities between π_r and every other node in V, and creates a cluster C containing π_r and all points u such that $\sigma(\pi_r, u) = +1$. The algorithm then recursively invokes itself on $V \setminus C$. On any instance of CC, KwikCluster achieves an expected error bounded by 3OPT. However, it is easy to see that KwikCluster makes $\Theta(n^2)$ queries in the worst case (e.g., if σ is the constant function -1). Our algorithms can be seen as a parsimonious version of KwikCluster whose goal is reducing the number of queries.

The work closest to ours is Bonchi et al. [2013]. Their algorithm runs KwikCluster on a random subset of $1/(2\varepsilon)$ nodes and stores the set Π of resulting pivots. Then, each node $v \in V \setminus \Pi$ is assigned to the cluster identified by the pivot $\pi \in \Pi$ with smallest index and such that $\sigma(v, \pi) = +1$. If no such pivot is found, then v becomes a singleton cluster. According to [Bonchi et al., 2013, Lemma 4.1], the expected clustering error for this variant is $3\mathrm{OPT} + \mathcal{O}(\epsilon n^2)$, which can be compared to our bound for ACC by setting $Q = n/\epsilon$. On the other hand our algorithms are much simpler and significantly easier to analyze. This allows us to prove a set of additional properties, such as cluster recovery and instance-dependent query bounds. It is unclear whether these results are obtainable with the techniques of Bonchi et al. [2013].

Another line of work attempts to circumvent computational hardness by using the more powerful same-cluster queries (SCQ). A same-cluster query tells whether any two given nodes are clustered together according to an optimal clustering or not. In Ailon et al. [2018a] SCQs are used to design a FPTAS for a variant of CC with bounded number of clusters. In Saha and Subramanian [2019a] SCQs are used to design algorithms for solving CC optimally by giving bounds on Q which depend on OPT. Unlike our setting, both works assume *all* $\binom{n}{2}$ similarities are known in advance. The work Mazumdar and Saha [2017b] considers the case in which there is a latent clustering with OPT $= 0$. The algorithm can issue SCQs, however the oracle is noisy: each query is answered incorrectly with some probability, and the noise is persistent (repeated queries give the same noisy answer). The above setting is closely related to the stochastic block model (SBM), which is a well-studied model for cluster recovery Abbe and Sandon [2015]; Massoulié [2014]; Mossel et al. [2018]. However, few works investigate SBMs with pairwise queries Chen et al. [2016]. Our setting is strictly harder because our oracle has a budget of OPT adversarially incorrect answers.

A different model is edge classification. Here the algorithm is given a graph $\mathcal{G}$ with hidden binary labels on the edges. The task is to predict the sign of all edges by querying as few labels as possible Cesa-Bianchi et al. [2012]; Chen et al. [2014]; Chiang et al. [2014]. As before, the oracle can have a budget OPT of incorrect answers, or a latent clustering with OPT $= 0$ is assumed and the oracle's answers are affected by persistent noise. Unlike correlation clustering, in edge classification the algorithm is not constrained to predict in agreement with a partition of the nodes. On the other hand, the algorithm cannot query arbitrary pairs of nodes in V, but only those that form an edge in $\mathcal{G}$.

Preliminaries and notation. We denote by $V \equiv \{1, \dots, n\}$ the set of input nodes, by $\mathcal{E} \equiv \binom{V}{2}$ the set of all pairs $\{u, v\}$ of distincts nodes in V, and by $\sigma : \mathcal{E} \to \{-1, +1\}$ the binary similarity function. A clustering $\mathcal{C}$ is a partition of V in disjoint clusters $C_i : i = 1, \dots, k$. Given $\mathcal{C}$ and σ, the set $\Gamma_{\mathcal{C}}$ of mistaken edges contains all pairs $\{u, v\}$ such that $\sigma(u, v) = -1$ and u, v belong to same cluster of $\mathcal{C}$ and all pairs $\{u, v\}$ such that $\sigma(u, v) = +1$ and u, v belong to different clusters of $\mathcal{C}$. The cost $\Delta_{\mathcal{C}}$ of $\mathcal{C}$ is $\left|\Gamma_{\mathcal{C}}\right|$. The correlation clustering index is OPT $= \min_{\mathcal{C}} \Delta_{\mathcal{C}}$, where the minimum is over all clusterings $\mathcal{C}$. We often view V, σ as a graph $G = (V, E)$ where $\{u, v\} \in E$ is an edge if and only if $\sigma(u, v) = +1$. In this case, for any subset $U \subseteq V$ we let $G[U]$ be the subgraph of G induced by U, and for any $v \in V$ we let $\mathcal{N}_v$ be the neighbor set of v.

A triangle is any unordered triple $T = \{u, v, w\} \subseteq V$. We denote by $e = \{u, w\}$ a generic triangle edge; we write $e \subset T$ and $v \in T \setminus e$. We say T is a *bad triangle* if the labels $\sigma(u, v), \sigma(u, w), \sigma(v, w)$ are $\{+, +, -\}$ (the order is irrelevant). We denote by $\mathcal{T}$ the set of all bad triangles in V. It is easy to see that the number of edge-disjoint bad triangles is a lower bound on OPT.

5.3 The ACC algorithm

We introduce our active learning algorithm ACC (Active Correlation Clustering).

Algorithm 15 ACC with query rate f

```
Parameters: residual node set $V_r$, round index $r$
 1: if $|V_r| = 0$ then RETURN
 2: if $|V_r| = 1$ then output singleton cluster $V_r$ and RETURN
 3: if $r > \lceil f(|V_1| - 1) \rceil$ then RETURN
 4: Draw pivot $\pi_r$ u.a.r. from $V_r$
 5: $C_r \leftarrow \{\pi_r\}$                                  ▷ Create new cluster and add the pivot to it
 6: Draw a random subset $S_r$ of $\lceil f(|V_r| - 1) \rceil$ nodes from $V_r \setminus \{\pi_r\}$
 7: for each $u \in S_r$ do query $\sigma(\pi_r, u)$
 8: if $\exists\, u \in S_r$ such that $\sigma(\pi_r, u) = +1$ then     ▷ Check if there is at least a positive edge
 9:     Query all remaining pairs $(\pi_r, u)$ for $u \in V_r \setminus \big(\{\pi_r\} \cup S_r\big)$
10:     $C_r \leftarrow C_r \cup \{u : \sigma(\pi_r, u) = +1\}$            ▷ Populate cluster based on queries
11: Output cluster $C_r$
12: ACC$(V_r \setminus C_r, r + 1)$                          ▷ Recursive call on the remaining nodes
```

ACC has the same recursive structure as KwikCluster. First, it starts with the full instance $V_1 = V$. Then, for each round $r = 1, 2, \dots$ it selects a random pivot $\pi_r \in V_r$, queries the similarities between π_r and a subset of V_r, removes π_r and possibly other points from V_r, and proceeds on the remaining residual subset V_{r+1}. However, while KwikCluster queries $\sigma(\pi_r, u)$ for *all* $u \in V_r \setminus \{\pi_r\}$, ACC queries only $\lceil f(n_r) \rceil \le n_r$ other nodes u (lines 6–7), where $n_r = |V_r| - 1$. Thus, while KwikCluster always finds all positive labels involving the pivot π_r, ACC can find them or not, with a probability that depends on f. The function f is called *query rate function* and dictates the tradeoff between the clustering cost Δ and the number of queries Q, as we prove below. Now, if any of the aforementioned $\lceil f(n_r) \rceil$ queries returns a positive label (line 8), then all the labels between π_r and the remaining $u \in V_r$ are queried and the algorithm operates as KwikCluster until the end of the recursive call; otherwise, the pivot becomes a singleton cluster which is removed from the set of nodes. Another important difference is that ACC deterministically stops after at most $\lceil f(n) \rceil$ recursive calls (line 1), declaring all remaining points as singleton clusters. The intuition is that with good probability the clusters not found within $\lceil f(n) \rceil$ rounds are small enough to be safely disregarded. Since the choice of f is delicate, we avoid trivialities by assuming f is positive and smooth enough. Formally:

Definition 5.1. *$f : \mathbb{N} \to \mathbb{R}$ is a* query rate function *if $f(1) = 1$, and $f(n) \le f(n+1) \le \big(1 + \frac{1}{n}\big) f(n)$ for all $n \in \mathbb{N}$. This implies $\frac{f(n+k)}{n+k} \le \frac{f(n)}{n}$ for all $k \ge 1$.*

We can now state formally our bounds for ACC.

Theorem 5.2. *For any query rate function f and any labeling σ on n nodes, the expected cost $\mathbb{E}[\Delta_A]$ of the clustering output by* ACC *satisfies*

$$\mathbb{E}[\Delta_A] \le 3\mathrm{OPT} + \frac{2e-1}{2(e-1)} \frac{n^2}{f(n)} + \frac{n}{e} .$$

The number of queries made by ACC *is deterministically bounded as $Q \le n \lceil f(n) \rceil$. In the special case $f(n) = n$ for all $n \in \mathbb{N}$,* ACC *reduces to KwikCluster and achieves $\mathbb{E}[\Delta_A] \le 3\mathrm{OPT}$ with $Q \le n^2$.*

Note that Theorem 5.2 gives an upper bound on the error achievable when using Q queries: since $Q = nf(n)$, the expected error is at most $3\mathrm{OPT} + \mathcal{O}(n^3/Q)$. Furthermore, as one expects, if the learner is allowed to ask for all edge signs, then the *exact* bound of KwikCluster is recovered (note that the first formula in Theorem 5.2 clearly does not take into account the special case when $f(n) = n$, which is considered in the last part of the statement).

Proof sketch. Look at a generic round r, and consider a pair of points $\{u, w\} \in V_r$. The essence is that ACC can misclassify $\{u, w\}$ in one of two ways. First, if $\sigma(u, w) = -1$, ACC can choose as pivot π_r a node v such that $\sigma(v, u) = \sigma(v, w) = +1$. In this case, if the condition on line 8 holds, then ACC will cluster v together with u and w, thus mistaking $\{u, w\}$. If instead $\sigma(u, w) = +1$, then ACC could mistake $\{u, w\}$ by pivoting on a node v such that $\sigma(v, u) = +1$ and $\sigma(v, w) = -1$, and clustering together only v and u. Crucially, both cases imply the existence of a bad triangle $T = \{u, w, v\}$. We charge each such mistake to exactly one bad triangle T, so that no triangle is charged twice. The expected number of mistakes can then be bound by 3OPT using the packing argument of Ailon et al. [2008] for KwikCluster. Second, if $\sigma(u, w) = +1$ then ACC could choose one of them, say u, as pivot π_r, and assign it to a singleton cluster. This means the condition on line 8 fails. We can then bound the number of such mistakes as follows. Suppose π_r has $cn/f(n)$ positive labels towards V_r for some $c \geq 0$. Loosely speaking, we show that the check of line 8 fails with probability e^{-c}, in which case $cn/f(n)$ mistakes are added. In expectation, this gives $cne^{-c}/f(n) = \mathcal{O}\big(n/f(n)\big)$ mistakes. Over all $f(n) \leq n$ rounds, this gives an overall $\mathcal{O}\big(n^2/f(n)\big)$. (The actual proof has to take into account that all the quantities involved here are not constants, but random variables).

5.3.1 ACC with Early Stopping Strategy

We can refine our algorithm ACC so that, in some cases, it takes advantage of the structure of the input to reduce significantly the expected number of queries. To this end we see the input as a graph G with edges corresponding to positive labels (see above). Suppose then G contains a sufficiently small number $\mathcal{O}(n^2/f(n))$ of edges. Since ACC performs up to $\lceil f(n) \rceil$ rounds, it could make $Q = \Theta(f(n)^2)$ queries. However, with just $\lceil f(n) \rceil$ queries one could *detect* that G contains $\mathcal{O}(n^2/f(n))$ edges, and immediately return the trivial clustering formed by all singletons. The expected error would obviously be at most $\mathrm{OPT} + \mathcal{O}(n^2/f(n))$, i.e. the same of Theorem 5.2. More generally, at each round r with $\lceil f(n_r) \rceil$ queries one can check if the residual graph contains at least $n^2/f(n)$ edges; if the test fails, declaring all nodes in V_r as singletons gives expected additional error $\mathcal{O}(n^2/f(n))$. The resulting algorithm is a variant of ACC that we call ACCESS (ACC with Early Stopping Strategy). The pseudocode can be found in the supplementary material.

First, we show ACCESS gives guarantees virtually identical to ACC (only, with Q in expectation). Formally:

Theorem 5.3. *For any query rate function f and any labeling σ on n nodes, the expected cost $\mathbb{E}[\Delta_A]$ of the clustering output by* ACCESS *satisfies*

$$\mathbb{E}[\Delta_A] \leq 3\mathrm{OPT} + 2\frac{n^2}{f(n)} + \frac{n}{e} \ .$$

Moreover, the expected number of queries performed by ACCESS *is $\mathbb{E}[Q] \leq n(\lceil f(n) \rceil + 4)$.*

Theorem 5.3 reassures us that ACCESS is no worse than ACC. In fact, if most edges of G belong to relatively large clusters (namely, all but $O(n^2/f(n))$ edges), then we can show ACCESS uses much fewer queries than ACC (in a nutshell, ACCESS quickly finds all large clusters and then quits). The following theorem captures the essence. For simplicity we assume $\mathrm{OPT} = 0$, i.e. G is a disjoint union of cliques.

Theorem 5.4. *Suppose* $\mathrm{OPT} = 0$ *so G is a union of disjoint cliques $C_1, \ldots, C_\ell$ listed in nondecreasing order of size. Let i' be the smallest i such that $\sum_{j=1}^{i} |E_{C_j}| = \Omega(n^2/f(n))$, and let $h(n) = |C_{i'}|$. Then* ACCESS *makes in expectation $\mathbb{E}[Q] = \mathcal{O}\big(n^2 \lg(n)/h(n)\big)$ queries.*

As an example, say $f(n) = \sqrt{n}$ and G contains $n^{1/3}$ cliques of $n^{2/3}$ nodes each. Then for ACC Theorem 5.2 gives $Q \leq nf(n) = \mathcal{O}(n^{3/2})$, while for ACCESS Theorem 5.4 gives $\mathbb{E}[Q] = \mathcal{O}(n^{4/3} \lg(n))$.

5.4 Cluster recovery

In the previous section we gave bounds on $\mathbb{E}[\Delta]$, the expected *total* cost of the clustering. However, in applications such as community detection and alike, the primary objective is recovering accurately the latent clusters of the graph, the sets of nodes that are "close" to cliques. This is usually referred to as *cluster recovery.* For this problem, an algorithm that outputs a good approximation $\hat{C}$ of every latent cluster C is preferable to an algorithm that minimizes $\mathbb{E}[\Delta]$ globally. In this section we show that ACC natively outputs clusters that are close to the latent clusters in the graph, thus acting as a cluster recovery tool. We also show that, for a certain type of latent clusters, one can amplify the accuracy of ACC via independent executions and recover all clusters exactly with high probability.

To capture the notion of "latent cluster", we introduce the concept of $(1-\epsilon)$-*knit* set. As usual, we view V, σ as a graph $G = (V, E)$ with $e \in E$ iff $\sigma(e) = +1$. Let E_C be the edges in the subgraph induced by $C \subseteq V$ and $\mathrm{cut}(C, \bar{C})$ be the edges between C and $\bar{C} = V \setminus C$.

Definition 5.5. *A subset* $C \subseteq V$ *is* $(1-\epsilon)$-knit *if* $\left|E_C\right| \geq (1-\epsilon)\binom{|C|}{2}$ *and* $\left|\mathrm{cut}(C, \bar{C})\right| \leq \epsilon\binom{|C|}{2}$.

Suppose now we have a cluster $\hat{C}$ as "estimate" of C. We quantify the distance between C and $\hat{C}$ as the cardinality of their symmetric difference, $\left|\hat{C} \oplus C\right| = \left|\hat{C} \setminus C\right| + \left|C \setminus \hat{C}\right|$. The goal is to obtain, for each $(1-\epsilon)$-knit set C in the graph, a cluster $\hat{C}$ with $|\hat{C} \oplus C| = \mathcal{O}(\epsilon|C|)$ for some small ϵ. We prove ACC does exactly this. Clearly, we must accept that if C is too small, i.e. $|C| = o(n/f(n))$, then ACC will miss C entirely. But, for $|C| = \Omega(n/f(n))$, we can prove $\mathbb{E}[|\hat{C} \oplus C|] = \mathcal{O}(\epsilon|C|)$. We point out that the property of being $(1-\epsilon)$-knit is rather weak for an algorithm, like ACC, that is completely oblivious to the global topology of the cluster — all what ACC tries to do is to blindly cluster together all the neighbors of the current pivot. In fact, consider a set C formed by two disjoint cliques of equal size. This set would be close to $1/2$-knit, and yet ACC would never produce a single cluster $\hat{C}$ corresponding to C. Things can only worsen if we consider also the edges in $\mathrm{cut}(C, \bar{C})$, which can lead ACC to assign the nodes of C to several different clusters when pivoting on $\bar{C}$. Hence it is not obvious that a $(1-\epsilon)$-knit set C can be efficiently recovered by ACC.

Note that this task can be seen as an *adversarial* cluster recovery problem. Initially, we start with a disjoint union of cliques, so that $\mathrm{OPT} = 0$. Then, an adversary flips the signs of some of the edges of the graph. The goal is to retrieve every original clique that has not been perturbed excessively. Note that we put no restriction on how the adversary can flip edges; therefore, this adversarial setting subsumes constrained adversaries. For example, it subsumes the high-probability regime of the stochastic block model Holland et al. [1983] where edges are flipped according to some distribution.

We can now state our main cluster recovery bound for ACC.

Theorem 5.6. *For every* $C \subseteq V$ *that is* $(1-\epsilon)$*-knit,* ACC *outputs a cluster* $\hat{C}$ *such that* $\mathbb{E}\left[|C \oplus \hat{C}|\right] \leq 3\epsilon|C| + \min\left\{\frac{2n}{f(n)}, \left(1 - \frac{f(n)}{n}\right)|C|\right\} + |C|e^{-|C|f(n)/5n}$.

The min in the bound captures two different regimes: when $f(n)$ is very close to n, then $\mathbb{E}\left[|C \oplus \hat{C}|\right] = \mathcal{O}(\epsilon|C|)$ independently of the size of C, but when $f(n) \ll n$ we need $|C| = \Omega(n/f(n))$, i.e., $|C|$ must be large enough to be found by ACC.

5.4.1 Exact cluster recovery via amplification

For certain latent clusters, one can get recovery guarantees significantly stronger than the ones given natively by ACC (see Theorem 5.6). We start by introducing *strongly* $(1-\epsilon)$*-knit* sets (also known as quasi-cliques). Recall that $\mathcal{N}_v$ is the neighbor set of v in the graph G induced by the positive labels.

Definition 5.7. *A subset* $C \subseteq V$ *is* strongly $(1-\epsilon)$-knit *if, for every* $v \in C$*, we have* $\mathcal{N}_v \subseteq C$ *and* $|\mathcal{N}_v| \geq (1-\epsilon)(|C|-1)$*.*

We remark that ACC alone does not give better guarantees on strongly $(1-\epsilon)$-knit subsets than on $(1-\epsilon)$-knit subsets. Suppose for example that $|\mathcal{N}_v| = (1-\epsilon)(|C|-1)$ for all $v \in C$. Then C is strongly $(1-\epsilon)$-knit, and yet when pivoting on any $v \in C$ ACC will inevitably produce a cluster $\hat{C}$ with $|\hat{C} \oplus C| \geq \epsilon|C|$, since the pivot has edges to less than $(1-\epsilon)|C|$ other nodes of C.

To bypass this limitation, we run ACC several times to amplify the probability that every node in C is found. Recall that $V = [n]$. Then, we define the id of a cluster $\hat{C}$ as the smallest node of $\hat{C}$. The min-tagging rule is the following: when forming $\hat{C}$, use its id to tag all of its nodes. Therefore, if $u_{\hat{C}} = \min\{u \in \hat{C}\}$ is the id of $\hat{C}$, we will set $\mathrm{id}(v) = u_{\hat{C}}$ for every $v \in \hat{C}$. Consider now the following algorithm, called ACR (Amplified Cluster Recovery). First, ACR performs K independent runs of ACC on input V, using the min-tagging rule on each run. In this way, for each $v \in V$ we obtain K tags $\mathrm{id}_1(v), \ldots, \mathrm{id}_K(v)$, one for each run. Thereafter, for each $v \in V$ we select the tag that v has received most often, breaking ties arbitrarily. Finally, nodes with the same tag are clustered together. One can prove that, with high probability, this clustering contains all strongly $(1-\epsilon)$-knit sets. In other words, ACR with high probability recovers all such latent clusters *exactly.* Formally, we prove:

Theorem 5.8. *Let $\epsilon \leq \frac{1}{10}$ and fix $p > 0$. If* ACR *is run with $K = 48 \ln \frac{n}{p}$, then the following holds with probability at least $1-p$: for every strongly $(1-\epsilon)$-knit C with $|C| > 10\frac{n}{f(n)}$, the algorithm outputs a cluster $\hat{C}$ such that $\hat{C} = C$.*

It is not immediately clear that one can extend this result by relaxing the notion of strongly $(1-\epsilon)$-knit set so to allow for edges between C and the rest of the graph. We just notice that, in that case, every node $v \in C$ could have a neighbor $x_v \in V \setminus C$ that is smaller than every node of C. In this case, when pivoting on v ACC would tag v with x rather than with u_C, disrupting ACR.

5.5 A fully additive scheme

In this section, we introduce a(n inefficient) fully additive approximation algorithm achieving cost $\mathrm{OPT} + n^2\varepsilon$ in high probability using order of $\frac{n}{\varepsilon^2}$ queries. When $\mathrm{OPT} = 0$, $Q = \frac{n}{\varepsilon} \ln \frac{1}{\varepsilon}$ suffices. Our algorithm combines uniform sampling with empirical risk minimization and is analyzed using VC theory.

First, note that CC can be formulated as an agnostic binary classification problem with binary classifiers $h_{\mathcal{C}} : \mathcal{E} \to \{-1,+1\}$ associated with each clustering $\mathcal{C}$ of V (recall that $\mathcal{E}$ denotes the set of all pairs $\{u,v\}$ of distinct elements $u, v \in V$), and we assume $h_{\mathcal{C}}(u,v) = +1$ iff u and v belong to the same cluster of $\mathcal{C}$. Let $\mathcal{H}_n$ be the set of all such $h_{\mathcal{C}}$. The risk of a classifier $h_{\mathcal{C}}$ with respect to the uniform distribution over $\mathcal{E}$ is $\mathbb{P}(h_{\mathcal{C}}(e) \neq \sigma(e))$ where e is drawn u.a.r. from $\mathcal{E}$. It is easy to see that the risk of any classifier $h_{\mathcal{C}}$ is directly related to $\Delta_{\mathcal{C}}$, $\mathbb{P}\big(h_{\mathcal{C}}(e) \neq \sigma(e)\big) = \Delta_{\mathcal{C}} / \binom{n}{2}$. Hence, in particular, $\mathrm{OPT} = \binom{n}{2} \min_{h \in \mathcal{H}_n} \mathbb{P}\big(h(e) \neq \sigma(e)\big)$. Now, it is well known —see, e.g., [Shalev-Shwartz and Ben-David, 2014b, Theorem 6.8]— that we can minimize the risk to within an additive term of ε using the following procedure: query $\mathcal{O}\big(d/\varepsilon^2\big)$ edges drawn u.a.r. from $\mathcal{E}$, where d is the VC dimension of $\mathcal{H}_n$, and find the clustering $\mathcal{C}$ such that $h_{\mathcal{C}}$ makes the fewest mistakes on the sample. If there is $h^* \in \mathcal{H}_n$ with zero risk, then $\mathcal{O}\big((d/\varepsilon)\ln(1/\varepsilon)\big)$ random queries suffice. A trivial upper bound on the VC dimension of $\mathcal{H}_n$ is $\log_2 |\mathcal{H}_n| = \mathcal{O}\big(n \ln n\big)$. The next result gives the exact value.

Theorem 5.9. *The VC dimension of the class $\mathcal{H}_n$ of all partitions of n elements is $n-1$.*

Proof. Let d be the VC dimension of $\mathcal{H}_n$. We view an instance of CC as the complete graph K_n with edges labelled by σ. Let T be any spanning tree of K_n. For any labeling σ, we can find a clustering $\mathcal{C}$ of V such that $h_{\mathcal{C}}$ perfectly classifies the edges of T: simply remove the edges with label -1 in T and consider the clusters formed by the resulting connected components. Hence $d \geq n-1$ because any spanning tree has exactly $n-1$ edges. On the other hand, any set of n edges must contain at least a cycle. It is easy to see that no clustering $\mathcal{C}$ makes $h_{\mathcal{C}}$ consistent with the labeling σ that gives positive labels to all edges in the cycle but one. Hence $d < n$. □

An immediate consequence of the above is the following.

Theorem 5.10. *There exists a randomized algorithm A that, for all $0 < \varepsilon < 1$, finds a clustering $\mathcal{C}$ satisfying $\Delta_{\mathcal{C}} \leq \mathrm{OPT} + \mathcal{O}\big(n^2\epsilon\big)$ with high probability while using $Q = \mathcal{O}\big(\frac{n}{\varepsilon^2}\big)$*

queries. Moreover, if OPT $= 0$*, then* $Q = \mathcal{O}\big(\frac{n}{\varepsilon} \ln \frac{1}{\varepsilon}\big)$ *queries are enough to find a clustering* $\mathcal{C}$ *satisfying* $\Delta_{\mathcal{C}} = \mathcal{O}\big(n^2\epsilon\big)$*.*

5.6 Lower bounds

In this section we give two lower bounds on the expected clustering error of any (possibly randomized) algorithm. The first bound holds for OPT $= 0$, and applies to algorithms using a deterministically bounded number of queries. This bound is based on a construction from [Cesa-Bianchi et al., 2015, Lemma 11] and related to kernel-based learning.

Theorem 5.11. *For any* $\varepsilon > 0$ *such that* $\frac{1}{\varepsilon}$ *is an even integer, and for every (possibly randomized) learning algorithm asking fewer than* $\frac{1}{50\varepsilon^2}$ *queries with probability* 1*, there exists a labeling* σ *on* $n \geq \frac{16}{\varepsilon} \ln \frac{1}{\varepsilon}$ *nodes such that* OPT $= 0$ *and the expected cost of the algorithm is at least* $\frac{n^2\varepsilon}{8}$*.*

Our second bound relaxed the assumption on OPT. It uses essentially the same construction of [Bonchi et al., 2013, Lemma 6.1], giving asymptotically the same guarantees. However, the bound of Bonchi et al. [2013] applies only to a very restricted class of algorithms: namely, those where the number q_v of queries involving any specific node $v \in V$ is deterministically bounded. This rules out a vast class of algorithms, including KwikCluster, ACC, and ACCESS, where the number of queries involving a node is a function of the random choices of the algorithm. Our lower bound is instead fully general: it holds unconditionally for *any* randomized algorithm, with no restriction on what or how many pairs of points are queried.

Theorem 5.12. *Choose any function* $\epsilon = \epsilon(n)$ *such that* $\Omega\big(\frac{1}{n}\big) \leq \epsilon \leq \frac{1}{2}$ *and* $\frac{1}{\epsilon} \in \mathbb{N}$*. For every (possibly randomized) learning algorithm and any* $n_0 > 0$ *there exists a labeling* σ *on* $n \geq n_0$ *nodes such that the algorithm has expected error* $\mathbb{E}[\Delta] \geq$ OPT $+ \frac{n^2\epsilon}{80}$ *whenever its expected number of queries satisfies* $\mathbb{E}[Q] < \frac{n}{80\,\epsilon}$*.*

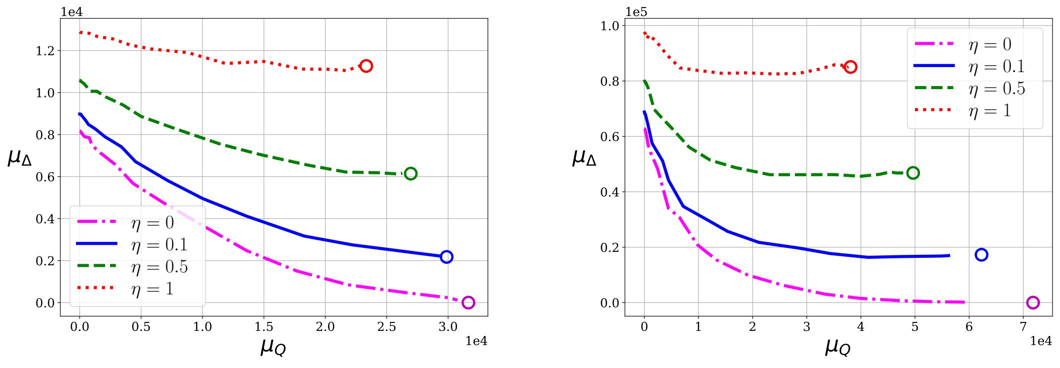
In fact, the bound of Theorem 5.12 can be put in a more general form: for any constant $c \geq 1$, the expected error is at least $c \cdot$ OPT $+ A(c)$ where $A(c) = \Omega(n^2\epsilon)$ is an additive term with constant factors depending on c (see the proof). Thus, our algorithms ACC and ACCESS are essentially optimal in the sense that, for $c = 3$, they guarantee an optimal additive error up to constant factors.

5.7 Experiments

We verify experimentally the tradeoff between clustering cost and number of queries of ACC, using six datasets from Mazumdar and Saha [2017b,a]. Four datasets come from real-world data, and two are synthetic; all of them provide a ground-truth partitioning of some set V of nodes. Here we show results for one real-world dataset (`cora`, with $|V|$=1879 and 191 clusters) and one synthetic dataset (`skew`, with $|V|$=900 and 30 clusters). Results for the remaining datasets are similar and can be found in the supplementary material. Since the original datasets have OPT $= 0$, we derived perturbed versions where OPT > 0 as follows. First, for each $\eta \in \{0, 0.1, 0.5, 1\}$ we let $p = \eta|E|/\binom{n}{2}$ where $|E|$ is the number of edges (positive labels) in the dataset (so η is the expected number of flipped edges measured as a multiple of $|E|$). Then, we flipped the label of each pair of nodes independently with probability p. Obviously for $p = 0$ we have the original dataset.

For every dataset and its perturbed versions we then proceeded as follows. For $\alpha = 0, 0.05, ..., 0.95, 1$, we set the query rate function to $f(x) = x^\alpha$. Then we ran 20 independent executions of ACC, and computed the average number of queries μ_Q and average clustering cost μ_Δ. The variance was often negligible, but is reported in the full plots in the supplementary material. The tradeoff between μ_Δ and μ_Q is depicted in Figure 5.7.1, where the circular marker highlights the case $f(x) = x$, i.e. KwikCluster.

The clustering cost clearly drops as the number of queries increases. This drop is particularly marked on `cora`, where ACC achieves a clustering cost close to that of KwikCluster using an order of magnitude fewer queries. It is also worth noting that, for the case OPT $= 0$,



(A) skew. (B) cora.

Figure 5.7.1: Performance of ACC.

the measured clustering cost achieved by ACC is 2 to 3 times lower than the theoretical bound of $\approx 3.8n^3/Q$ given by Theorem 5.2.

Appendix

5.A Probability bounds

We give Chernoff-type probability bounds that can be found in e.g. Dubhashi and Panconesi [2009b] and that we repeatedly use in our proofs. Let $X_1, \dots, X_n$ be binary random variables. We say that $X_1, \dots, X_n$ are non-positively correlated if for all $I \subseteq \{1, \dots, n\}$ we have:

$$\mathbb{P}[\forall i \in I : X_i = 0] \le \prod_{i \in I} \mathbb{P}[X_i = 0] \quad \text{and} \quad \mathbb{P}[\forall i \in I : X_i = 1] \le \prod_{i \in I} \mathbb{P}[X_i = 1] \tag{5.1}$$

The following holds:

Lemma 5.13. *Let $X_1, \dots, X_n$ be independent or, more generally, non-positively correlated binary random variables. Let $a_1, \dots, a_n \in [0,1]$ and $X = \sum_{i=1}^n a_i X_i$. Then, for any $\delta > 0$, we have:*

$$\mathbb{P}[X < (1-\delta)\mathbb{E}[X]] < e^{-\frac{\delta^2}{2}\mathbb{E}[X]} \tag{5.2}$$

$$\mathbb{P}[X > (1+\delta)\mathbb{E}[X]] < e^{-\frac{\delta^2}{2+\delta}\mathbb{E}[X]} \tag{5.3}$$

5.B Supplementary Material for Section 3

5.B.1 Pseudocode of ACC

For ease of reference we report the pseudocode of ACC below.

Algorithm 16 ACC with query rate f

```
Parameters: residual node set $V_r$, round index $r$
 1: if $|V_r| = 0$ then RETURN
 2: if $|V_r| = 1$ then output singleton cluster $V_r$ and RETURN
 3: if $r > \lceil f(|V_1| - 1) \rceil$ then RETURN
 4: Draw pivot $\pi_r$ u.a.r. from $V_r$
 5: $C_r \leftarrow \{\pi_r\}$                          ▷ Create new cluster and add the pivot to it
 6: Draw a random subset $S_r$ of $\lceil f(|V_r| - 1) \rceil$ nodes from $V_r \setminus \{\pi_r\}$
 7: for each $u \in S_r$ do query $\sigma(\pi_r, u)$
 8: if $\exists\, u \in S_r$ such that $\sigma(\pi_r, u) = +1$ then    ▷ Check if there is at least a positive edge
 9:     Query all remaining pairs $(\pi_r, u)$ for $u \in V_r \setminus (\{\pi_r\} \cup S_r)$
10:     $C_r \leftarrow C_r \cup \{u : \sigma(\pi_r, u) = +1\}$       ▷ Populate cluster based on queries
11: Output cluster $C_r$
12: ACC($V_r \setminus C_r, r + 1$)                    ▷ Recursive call on the remaining nodes
```

5.B.2 Proof of Theorem 1

We refer to the pseudocode above (Algorithm 15). We use V_r to denote the set of remaining nodes at the beginning of the r-th recursive call, and we let $n_r = |V_r| - 1$. Hence $V_1 = V$ and $n_1 = n - 1$. If the condition in the **if** statement on line 8 is not true, then C_r is a singleton cluster. We denote by V_{sing} the set nodes that are output as singleton clusters.

Let Γ_A be the set of mistaken edges for the clustering output by ACC and let $\Delta_A = |\Gamma_A|$ be the cost of this clustering. Note that, in any recursive call, ACC misclassifies an edge $e = \{u, w\}$ if and only if e is part of a bad triangle whose third node v is chosen as pivot and does not become a singleton cluster, or if $\sigma(e) = +1$ and at least one of u, w becomes a singleton cluster. More formally, ACC misclassifies an edge $e = \{u, w\}$ if and only if one of the following three disjoint events holds:

$B_1(e)$: There exists $r \le \lceil f(n-1) \rceil$ and a bad triangle $T \equiv \{u, v, w\} \subseteq V_r$ such that $\pi_r = v$ and $v \notin V_{\text{sing}}$.

$B_2(e)$: There exists $r \le \lceil f(n-1) \rceil$ such that $u, w \in V_r$ with $\sigma(u, w) = +1$ and $\pi_r \in \{u, w\} \cap V_{\text{sing}}$.

$B_3(e)$: ACC stops after $\lceil f(n-1) \rceil$ rounds without removing neither u nor w, and $\sigma(u, w) = +1$.

Therefore the indicator variable for the event "e is mistaken" is:

$$\{e \in \Gamma_A\} = \{B_1(e)\} + \{B_2(e)\} + \{B_3(e)\}$$

The expected cost of the clustering is therefore:

$$\mathbb{E}[\Delta_A] = \sum_{e \in \mathcal{E}} \mathbb{P}(B_1(e)) + \sum_{e \in \mathcal{E}} \mathbb{P}(B_2(e)) + \sum_{e \in \mathcal{E}} \mathbb{P}(B_3(e)) \tag{5.4}$$

We proceed to bound the three terms separately.

Bounding $\sum_{e \in \mathcal{E}} \mathbb{P}(B_1(e))$. Fix an arbitrary edge $e = \{u, w\}$. Note that, if $B_1(e)$ occurs, then T is unique, i.e. exactly one bad triangle T in V satisfies the definition of $B_1(e)$. Each occurrence of $B_1(e)$ can thus be charged to a single bad triangle T. We may thus write

$$\begin{aligned}
\sum_{e \in \mathcal{E}} \{B_1(e)\} &= \sum_{e \in \mathcal{E}} \{(\exists r)(\exists T \in \mathcal{T}) \,:\, T \subseteq V_r \,\wedge\, e \subset T \,\wedge\, \pi_r \in T \setminus e \,\wedge\, \pi_r \notin V_{\text{sing}}\} \\
&= \sum_{T \in \mathcal{T}} \{(\exists r) \,:\, T \subseteq V_r \,\wedge\, \pi_r \in T \,\wedge\, \pi_r \notin V_{\text{sing}}\} \\
&\le \sum_{T \in \mathcal{T}} \{A_T\}
\end{aligned}$$

where $A_T \equiv \big\{(\exists r) \,:\, T \subseteq V_r \,\wedge\, \pi_r \in T\big\}$. Let us then bound $\sum_{T \in \mathcal{T}} \mathbb{P}(A_T)$. Let $\mathcal{T}(e) \equiv \{T' \in \mathcal{T} \,:\, e \in T'\}$. We use the following fact extracted from the proof of [Ailon et al., 2008, Theorem 6.1]. If $\{\beta_T \ge 0 \,:\, T \in \mathcal{T}\}$ is a set of weights on the bad triangles such that $\sum_{T \in \mathcal{T}(e)} \beta_T \le 1$ for all $e \in \mathcal{E}$, then $\sum_{T \in \mathcal{T}} \beta_T \le \text{OPT}$. Given $e \in \mathcal{E}$ and $T \in \mathcal{T}$, let $F_T(e)$ be the event corresponding to T being the first triangle in the set $\mathcal{T}(e)$ such that $T \in V_r$ and $\pi_r \in T \setminus e$ for some r. Now if $F_T(e)$ holds then A_T holds and no other $A_{T'}$ for $T' \in \mathcal{T}(e) \setminus \{T\}$ holds. Therefore

$$\sum_{T \in \mathcal{T}(e)} \{A_T \,\wedge\, F_T(e)\} = 1 \,.$$

If A_T holds for some r_0, then it cannot hold for any other $r > r_0$ because $\pi_{r_0} \in T$ implies that for all $r > r_0$ we have $\pi_{r_0} \notin V_r$ implying $T \not\subseteq V_r$. Hence, given that A_T holds for r_0, if $F_T(e)$ holds too, then it holds for the same r_0 by construction. This implies that $\mathbb{P}\big(F_T(e) \mid A_T\big) = \frac{1}{3}$ because ACC chooses the pivot u.a.r. from the nodes in V_{r_0}. Thus, for each $e \in E$ we can write

$$1 = \sum_{T \in \mathcal{T}(e)} \mathbb{P}\big(A_T \,\wedge\, F_T(e)\big) = \sum_{T \in \mathcal{T}(e)} \mathbb{P}\big(F_T(e) \mid A_T\big)\mathbb{P}(A_T) = \sum_{T \in \mathcal{T}(e)} \frac{1}{3}\mathbb{P}(A_T) \,. \tag{5.5}$$

Choosing $\beta_T = \frac{1}{3}\mathbb{P}(A_T)$ we get $\sum_{T \in \mathcal{T}} \mathbb{P}(A_T) \le 3\text{OPT}$.

In the proof of KwikCluster, the condition $\sum_{T \in \mathcal{T}(e)} \beta_T \le 1$ was ensured by considering events $G_T(e) = A_T \,\wedge\, e \in \Gamma_A$. Indeed, in KwikCluster the events $\{G_T(e) \,:\, T \in \mathcal{T}(e)\}$ are disjoint, because $G_T(e)$ holds iff T is the first and only triangle in $\mathcal{T}(e)$ whose node opposite to e is chosen as pivot. For ACC this is not true because a pivot can become a singleton cluster, which does not cause $e \in \Gamma_A$ necessarily to hold.

Bounding $\sum_{e\in\mathcal{E}}\mathbb{P}(B_2(e))$. For any $u \in V_r$, let $d_r^+(u) = \big|\{v \in V_r : \sigma(u,v) = +1\}\big|$. We have:

$$\sum_{e\in\mathcal{E}}\{B_2(e)\} = \frac{1}{2}\sum_{u\in V}\sum_{r=1}^{\lceil f(n-1)\rceil}\{\pi_r = u \wedge \pi_r \in V_{\text{sing}}\}\, d_r^+(u)\ .$$

Taking expectations with respect to the randomization of ACC,

$$\begin{aligned}\sum_{e\in\mathcal{E}}\mathbb{P}\big(B_2(e)\big) &= \frac{1}{2}\sum_{u\in V}\sum_{r=1}^{\lceil f(n-1)\rceil}\mathbb{E}\Big[\{\pi_r = u \wedge \pi_r \in V_{\text{sing}}\}\, d_r^+(u)\Big]\\ &= \frac{1}{2}\sum_{u\in V}\sum_{r=1}^{\lceil f(n-1)\rceil}\mathbb{E}\Big[\{\pi_r \in V_{\text{sing}}\}\, d_r^+(u)\,\Big|\,\pi_r = u\Big]\mathbb{P}(\pi_r = u)\end{aligned}$$

For any round r, let H_{r-1} be the sequence of random draws made by the algorithm before round r. Then $\mathbb{P}\big(\pi_r \in V_{\text{sing}}\,\big|\,\pi_r = u,\, H_{r-1}\big)d_r^+(u) = 0$ if either $d_r^+(u) = 0$, or $d_r^+(u) \geq 1$ and $d_r^-(u) < \lceil f(n_r)\rceil$. Otherwise,

$$\mathbb{P}\big(\pi_r \in V_{\text{sing}}\,\big|\,\pi_r = u,\, H_{r-1}\big) = \prod_{j=0}^{\lceil f(n_r)\rceil-1}\frac{d_r^-(u)-j}{n_r-j} \leq \left(\frac{d_r^-(u)}{n_r}\right)^{\lceil f(n_r)\rceil} = \left(1-\frac{d_r^+(u)}{n_r}\right)^{\lceil f(n_r)\rceil} \tag{5.6}$$

where the inequality holds because $d_r^-(u) \leq n_r$. Therefore, when $d_r^+(u) \geq 1$ and $d_r^-(u) \geq \lceil f(n_r)\rceil$,

$$\begin{aligned}\mathbb{E}\Big[\{\pi_r \in V_{\text{sing}}\}\, d_r^+(u)\,\Big|\,\pi_r = u,\, H_{r-1}\Big] &= \mathbb{P}\big(\pi_r \in V_{\text{sing}}\,\big|\,\pi_r = u,\, H_{r-1}\big)d_r^+(u)\\ &= \left(1-\frac{d_r^+(u)}{n_r}\right)^{\lceil f(n_r)\rceil} d_r^+(u)\\ &= \left(1-\frac{d_r^+(u)}{n_r}\right)^{\lceil f(n_r)\rceil} d_r^+(u)\\ &\leq \exp\left(-\frac{d_r^+(u)\lceil f(n_r)\rceil}{n_r}\right) d_r^+(u)\\ &\leq \max_{z>0}\exp\left(-\frac{z\,\lceil f(n_r)\rceil}{n_r}\right) z\\ &\leq \frac{n_r}{e\lceil f(n_r)\rceil}\\ &\leq \frac{n_r}{ef(n_r)}\ .\end{aligned}$$

Combining with the above, this implies

$$\sum_{e\in\mathcal{E}}\mathbb{P}\big(B_2(e)\big) \leq \frac{1}{2e}\sum_{r=1}^{\lceil f(n-1)\rceil}\mathbb{E}\left[\frac{n_r}{f(n_r)}\right] \leq \frac{1}{2e}\sum_{r=1}^{\lceil f(n-1)\rceil}\frac{n}{f(n)} \leq \frac{n}{e}$$

where we used the facts that $n_r \leq n$ and the properties of f.

Bounding $\sum_{e\in\mathcal{E}}\mathbb{P}(B_3(e))$. Let V_{fin} be the remaining vertices in V_r after the algorithm stops and assume $|V_{\text{fin}}| > 1$ (so that there is at least a query left). Let $n_{\text{fin}} = |V_{\text{fin}}| - 1$ and, for any $u \in V_{\text{fin}}$, let $d_{\text{fin}}^+(u) = \big|\{v \in V_{\text{fin}} : \sigma(u,v) = +1\}\big|$. In what follows, we conventionally assume $V_r \equiv V_{\text{fin}}$ for any $r > \lceil f(n-1)\rceil$, and similarly for n_{fin} and d_{fin}^+. We have

$$\sum_{e\in\mathcal{E}}\{B_3(e)\} = \frac{1}{2}\sum_{u\in V_{\text{fin}}} d_{\text{fin}}^+(u) \leq \frac{1}{2}\left(\sum_{u\in V_{\text{fin}}}\frac{n_{\text{fin}}}{\lceil f(n_{\text{fin}})\rceil} + \sum_{u\in V_{\text{fin}}}\left\{d_{\text{fin}}^+(u) > \frac{n_{\text{fin}}}{\lceil f(n_{\text{fin}})\rceil}\right\} d_{\text{fin}}^+(u)\right).$$

Fix some $r \leq \lceil f(n-1)\rceil$. Given any vertex $v \in V_r$ with $d_r^+(v) \geq \frac{n_r}{\lceil f(n_r)\rceil}$, let $E_r(v)$ be the event that, at round r, ACC queries $\sigma(v,u)$ for all $u \in V_r \setminus \{v\}$. Introduce the notation

$S_r = \sum_{u \in V_r} \left\{ d_r^+(u) > \frac{n_r}{\lceil f(n_r) \rceil} \right\} d_r^+(u)$ with $S_r = S_{\text{fin}}$ for all $r > \lceil f(n) \rceil$, and let $\delta_r = n_r - n_{r+1}$ be the number of nodes that are removed from V_r at the end of the r-th recursive call. Then

$$\delta_r \geq \{E_r(\pi_r)\}\, d_r^+(\pi_r) \geq \left\{ d_r^+(\pi_r) > \frac{n_r}{\lceil f(n_r) \rceil} \right\} \{E_r(\pi_r)\}\, d_r^+(\pi_r)$$

and

$$\mathbb{E}[\delta_r \mid H_{r-1}] \geq \sum_{v \in V_r} \left\{ d_r^+(v) > \frac{n_r}{\lceil f(n_r) \rceil} \right\} \mathbb{P}\big(E_r(v) \mid \pi_r = v,\, H_{r-1}\big) \mathbb{P}(\pi_r = v \mid H_{r-1}) d_r^+(v) \ .$$

Using the same argument as the one we used to bound (5.6),

$$\mathbb{P}\big(E_r(v) \mid \pi_r = v,\, H_{r-1}\big) \geq 1 - \left(1 - \frac{d_r^+(v)}{n_r}\right)^{\lceil f(n_r) \rceil} \geq 1 - \left(1 - \frac{1}{\lceil f(n_r) \rceil}\right)^{\lceil f(n_r) \rceil} \geq 1 - \frac{1}{e}$$

and $\mathbb{P}(\pi_r = v \mid H_{r-1}) = \frac{1}{n_r + 1}$ for any $v \in V_r$, we may write

$$\mathbb{E}[\delta_r \mid H_{r-1}] \geq \left(1 - \frac{1}{e}\right) \frac{\mathbb{E}[S_r \mid H_{r-1}]}{n_r + 1} \geq \left(1 - \frac{1}{e}\right) \frac{\mathbb{E}[S_r \mid H_{r-1}]}{n} \ .$$

Observe now that $\sum_{r=1}^{\lceil f(n-1) \rceil} \delta_r \leq n_1 - n_{\text{fin}} \leq n - 1$ and S_r is monotonically nonincreasing in r. Thus

$$n - 1 \geq \sum_{r=1}^{\lceil f(n-1) \rceil} \mathbb{E}[\delta_r] \geq \frac{1}{n}\left(1 - \frac{1}{e}\right) \sum_{r=1}^{\lceil f(n) \rceil} \mathbb{E}[S_r] \geq \frac{\lceil f(n-1) \rceil}{n}\left(1 - \frac{1}{e}\right) \mathbb{E}[S_{\text{fin}}]$$

which implies $\mathbb{E}[S_{\text{fin}}] \leq \left(\frac{e}{e-1}\right) \frac{n(n-1)}{\lceil f(n-1) \rceil} \leq \left(\frac{e}{e-1}\right) \frac{n(n-1)}{f(n-1)}$. By the properties of f, however, $\left(\frac{e}{e-1}\right) \frac{n(n-1)}{f(n-1)} \leq \left(\frac{e}{e-1}\right) \frac{n^2}{f(n)}$. So we have

$$\sum_{e \in \mathcal{E}} \mathbb{P}\big(B_3(e)\big) \leq \frac{1}{2} \left(\sum_{u \in V_{\text{fin}}} \mathbb{E}\left[\frac{n_{\text{fin}}}{f(n_{\text{fin}})} \right] + \mathbb{E}[S_{\text{fin}}] \right) \leq \frac{1}{2} \left(\frac{n^2}{f(n)} + \frac{e}{e-1} \frac{n^2}{f(n)} \right)$$

as claimed.

Bounding the number of queries. In any given round, ACC asks less than n queries. Since the number of rounds is at most $\lceil f(n) \rceil$, the overall number of queries is less than $n\lceil f(n) \rceil$.

KwikCluster as special case. One can immediately see that, if $f(n) = n$ for all n, then ACC coincides with KwikCluster and therefore the bound $\mathbb{E}[\Delta] \leq 3\text{OPT}$ applies Ailon et al. [2008].

5.B.3 Pseudocode of ACCESS

5.B.4 Proof of Theorem 2

We refer to the pseudocode of ACCESS (Algorithm 2).

Bounding $\mathbb{E}[\Delta_A]$. Let G_r be the residual graph at round r. The total clustering cost Δ_A of ACCESS can be bounded by the sum of two terms: the clustering cost Δ_1 of ACC without round restriction (i.e. ACC terminating only when the residual graph is empty), and the number of edges Δ_2 in the residual graph G_r if r is the round at which ACCESS stops. Concerning Δ_1, the proof of Theorem 1 shows that $\mathbb{E}[\Delta_1] \leq 3\text{OPT} + n/e$. Concerning Δ_2, we have two cases. If ACCESS stops at line 1, then obviously $\Delta_2 \leq 2n^2/f(n)$. If instead ACCESS stops at line 4, then note that for any $k \geq 0$ the probability that such an event

Algorithm 2 ACCESS with query rate f

```
Parameters: residual node set $V_r$, round index $r$
 1: if $\binom{|V_r|}{2} \le 2n^2/f(n)$ then STOP and declare every $v \in V_r$ as singleton
 2: Sample the labels of $\lceil \binom{|V_r|}{2} f(n)/n^2 \rceil$ pairs chosen u.a.r. from $\binom{V_r}{2}$
 3: if no label is positive then
 4:     STOP and declare every $v \in V_r$ as singleton
 5: Draw pivot $\pi_r$ u.a.r. from $V_r$
 6: $C_r \leftarrow \{\pi_r\}$                    ▷ Create new cluster and add the pivot to it
 7: Draw a random subset $S_r$ of $\lceil f(|V_r|-1) \rceil$ nodes from $V_r \setminus \{\pi_r\}$
 8: for each $u \in S_r$ do query $\sigma(\pi_r, u)$
 9: if $\exists\, u \in S_r$ such that $\sigma(\pi_r, u) = +1$ then      ▷ Check if there is at least an edge
10:     Query all remaining pairs $(\pi_r, u)$ for $u \in V_r \setminus (\{\pi_r\} \cup S_r)$
11:     $C_r \leftarrow C_r \cup \{u : \sigma(\pi_r, u) = +1\}$          ▷ Populate cluster based on queries
12: Output cluster $C_r$
13: ACCESS$(V_r \setminus C_r, r+1)$                ▷ Recursive call on the remaining nodes
```

happens given that $\Delta_2 = k$ is at most:

$$\left(1 - \frac{k}{\binom{|V_r|}{2}}\right)^{\lceil \binom{|V_r|}{2} f(n)/n^2 \rceil} \le e^{-kf(n)/n^2}$$

Thus $\mathbb{E}[\Delta_2] \le \max_{k \ge 1}(ke^{-kf(n)/n^2}) \le \frac{n^2}{ef(n)} < 2n^2/f(n)$.

Bounding $\mathbb{E}[Q]$. The queries performed at line 1 are deterministically at most $n\lceil f(n) \rceil$. Concerning the other queries (line 8 and line 10), we divide the algorithm in two phases: the "heavy" rounds r where G_r still contains at least $n^2/(2f(n))$ edges, and the remaining "light" rounds where G_r contains less than $n^2/(2f(n))$ edges.

Consider first a "heavy" round r. We see G_r as an arbitrary fixed graph: for all random variables mentioned below, the distribution is thought solely as a function of the choices of the algorithm in the current round (i.e., the pivot node π_r and the queried edges). Now, let Q_r be the number of queries performed at lines 8 and 10), and $R_r = |V_r| - |V_{r+1}|$ be the number of nodes removed. Let π_r be the pivot, and let D_r be its degree in G_r. Let X_r be the indicator random variable of the event that $\sigma(\pi_r, u) = +1$ for some $u \in S_r$. Observe that:

$$Q_r \le \lceil f(|V_r|-1) \rceil + X_r(|V_r|-1) \qquad \text{and} \qquad R_r = 1 + X_r\, D_r$$

Thus $\mathbb{E}[Q_r] \le \lceil f(|V_r|-1) \rceil + \mathbb{E}[X_r]|V_r|$, while $\mathbb{E}[R_r] = 1 + \mathbb{E}[X_r D_r]$. However, X_r is monotonically increasing in D_r, so $\mathbb{E}[X_r D_r] = \mathbb{E}[X_r]\mathbb{E}[D_r] + \mathrm{Cov}(X_r, D_r) \ge \mathbb{E}[X_r]\mathbb{E}[D_r]$. Moreover, by hypothesis $\mathbb{E}[D_r] \ge 2\big(n^2/(2f(n))\big)/|V_r| \ge n/f(n)$. Thus:

$$\begin{aligned}
\mathbb{E}[R_r] &\ge 1 + \mathbb{E}[X_r]\mathbb{E}[D_r] \\
&\ge 1 + \mathbb{E}[X_r]\frac{n}{f(n)} \\
&\ge 1 + \mathbb{E}[X_r]\frac{|V_r|}{f(|V_r|)} \\
&\ge 1 + \mathbb{E}[X_r]\frac{|V_r|}{\lceil f(|V_r|) \rceil} \\
&\ge \frac{\mathbb{E}[Q_r]}{\lceil f(|V_r|) \rceil} \\
&\ge \frac{\mathbb{E}[Q_r]}{\lceil f(n) \rceil}
\end{aligned}$$

But then, since obviously $\sum_r R_r \le n$:

$$\mathbb{E}\left[\sum_{r \text{ heavy}} Q_r\right] \le \lceil f(n) \rceil \mathbb{E}\left[\sum_{r \text{ heavy}} R_r\right] \le n\lceil f(n) \rceil$$

Consider now the "light" rounds, where G_r contains less than $n^2/(2f(n))$ edges. In any such round the expected number of edges found at line 2 is less than:

$$\frac{n^2/f(n)}{2\binom{|V_r|}{2}} \left\lceil \binom{|V_r|}{2} f(n)/n^2 \right\rceil \tag{5.7}$$

However, $\binom{|V_r|}{2} > 2n^2/f(n)$ otherwise ACCESS would have stopped at line 1, hence:

$$\left\lceil \binom{|V_r|}{2} f(n)/n^2 \right\rceil \le \frac{3}{2}\binom{|V_r|}{2} f(n)/n^2 \tag{5.8}$$

which implies that the expression in (5.7) is bounded by $\frac{3}{4}$. By Markov's inequality this is also an upper bound on the probability that ACCESS finds some edge at line 2, so in every light round ACCESS stops at line 4 with probability at least $\frac{1}{4}$. Hence ACCESS completes at most 4 light rounds in expectation; the corresponding expected number of queries is then at most $4n$.

5.B.5 Proof of Theorem 3

First of all, note that if the residual graph G_r contains $\mathcal{O}(n^2/f(n))$ edges, from r onward ACCESS stops at each round independently with constant probability. The expected number of queries performed before stopping is therefore $\mathcal{O}(n)$, and the expected error incurred is obviously at most $\mathcal{O}(n^2/f(n))$.

We shall then bound the expected number of queries required before the residual graph contains $\mathcal{O}(n^2/f(n))$ edges. In fact, by definition of i', if ACCESS removes $C_{i'}, \dots, C_\ell$, then the residual graph contains $\mathcal{O}(n^2/f(n))$ edges. We therefore bound the expected number of queries before $C_{i'}, \dots, C_\ell$ are removed.

First of all recall that, when pivoting on a cluster of size c, the probability that the cluster is *not* removed is at most $e^{-cf(n)/n}$. Thus the probability that the cluster is not removed after $\Omega(c)$ of its nodes have been used as pivot is $e^{-\Omega(c^2)f(n)/n}$. Hence the probability that *any* of $C_{i'}, \dots, C_\ell$ is not removed after $\Omega(c)$ of its nodes are used as pivot is, setting $c = \Omega\big(h(n)\big)$ and using a union bound, at most $p = ne^{-\Omega(h(n)^2)f(n)/n}$. Observe that $h(n) = \Omega\big(n/f(n)\big)$, for otherwise $\sum_{j=1}^{i'} \binom{C_j}{2} = o\big(n^2/f(n)\big)$, a contradiction. Therefore $p \le ne^{-\Omega(h(n))}$. Note also that we can assume $h(n) = \omega(\ln n)$, else the theorem bound is trivially $O(n^2)$. This gives $p = \mathcal{O}\big(ne^{-\omega(\ln n)}\big) = o\big(1/\operatorname{poly}(n)\big)$. We can thus condition on the events that, at any point along the algorithm, every cluster among $C_{i'}, \dots, C_\ell$ that is still in the residual graph has size $\Omega\big(h(n)\big)$; the probability of any other event changes by an additive $\mathcal{O}(p)$, which can be ignored.

Let now $k = \ell - i' + 1$, and suppose at a generic point $k' \le k$ of the clusters $C_{i'}, \dots, C_\ell$ are in the residual graph. Their total size is therefore $\Omega\big(k'h(n)\big)$. Therefore $\mathcal{O}\big(n/k'h(n)\big)$ rounds in expectation are needed for the pivot to fall among those clusters. Each time this happens, with probability $1 - e^{-\Omega(h(n))f(n)/n} = \Omega(1)$ the cluster containing the pivot is removed. Hence, in expectation a new cluster among $C_{i'}, \dots, C_\ell$ is removed after $\mathcal{O}\big(n/k'h(n)\big)$ rounds. By summing over all values of k', the number of expected rounds to remove all of $C_{i'}, \dots, C_\ell$ is

$$\mathcal{O}\left(\sum_{k'=1}^{k} \frac{n}{k'h(n)}\right) = \mathcal{O}\big(n(\ln n)/h(n)\big)$$

Since each round involves $\mathcal{O}(n)$ queries, the bound follows.

5.C Supplementary Material for Section 4

5.C.1 Proof of Theorem 4

Fix any C that is $(1-\epsilon)$-knit. We show that ACC outputs a $\hat{C}$ such that

$$\mathbb{E}\big[|\hat{C}\cap C|\big] \geq \max\left\{\left(1-\frac{5}{2}\epsilon\right)|C| - 2\frac{n}{f(n)}, \left(\frac{f(n)}{n}-\frac{5}{2}\epsilon\right)|C|\right\} \text{ and } \mathbb{E}\big[|\hat{C}\cap\bar{C}|\big] \leq \frac{\epsilon}{2}|C| \quad (5.9)$$

One can check that these two conditions together imply the first two terms in the bound. We start by deriving a lower bound on $\mathbb{E}\big[|\hat{C}\cap C|\big]$ for KwikCluster assuming $|E_C| = \binom{|C|}{2}$. Along the way we introduce most of the technical machinery. We then port the bound to ACC, relax the assumption to $|E_C| \geq (1-\epsilon)\binom{|C|}{2}$, and bound $\mathbb{E}\big[|\hat{C}\cap\bar{C}|\big]$ from above. Finally, we add the $|C|e^{-|C|f(n)/5n}$ part of the bound. To lighten the notation, from now on C denotes both the cluster and its cardinality $|C|$.

For the sake of analysis, we see KwikCluster as the following equivalent process. First, we draw a random permutation π of V. This is the ordered sequence of *candidate pivots*. Then, we set $G_1 = G$, and for each $i = 1,\ldots,n$ we proceed as follows. If $\pi_i \in G_i$, then π_i is used as an actual pivot; in this case we let $G_{i+1} = G_i \setminus (\pi_i \cup \mathcal{N}_{\pi_i})$ where $\mathcal{N}_v$ is the set of neighbors of v. If instead $\pi_i \notin G_i$, then we let $G_{i+1} = G_i$. Hence, G_i is the residual graph just before the i-th candidate pivot π_i is processed. We indicate the event $\pi_i \in G_i$ by the random variable P_i:

$$P_i = \{\pi_i \in G_i\} = \{\pi_i \text{ is used as pivot}\} \quad (5.10)$$

More in general, we define a random variable indicating whether node v is "alive" in G_i:

$$X(v,i) = \{v \in G_i\} = \big\{v \notin \cup_{j<i\,:\,P_j=1}(\pi_j \cup \mathcal{N}_{\pi_j})\big\} \quad (5.11)$$

Let $i_C = \min\{i : \pi_i \in C\}$ be the index of the first candidate pivot of C. Define the random variable:

$$S_C = |C \cap G_{i_C}| = \sum_{v\in C} X(v,i_C) \quad (5.12)$$

In words, S_C counts the nodes of C still alive in G_{i_C}. Now consider the following random variable:

$$S = P_{i_C} \cdot S_C \quad (5.13)$$

Let $\hat{C}$ be the cluster that contains π_{i_C} in the output of KwikCluster. It is easy to see that $|C\cap\hat{C}| \geq S$. Indeed, if $P_{i_C} = 1$ then $\hat{C}$ includes $C \cap G_{i_C}$, so $|C\cap\hat{C}| \geq P_{i_C}S_C = S$. If instead $P_{i_C} = 0$, then $S = 0$ and obviously $|C\cap\hat{C}| \geq 0$. Hence in any case $|C\cap\hat{C}| \geq S$, and $\mathbb{E}\big[|C\cap\hat{C}|\big] \geq \mathbb{E}[S]$. Therefore we can bound $\mathbb{E}\big[|C\cap\hat{C}|\big]$ from below by bounding $\mathbb{E}[S]$ from below.

Before continuing, we simplify the analysis by assuming KwikCluster runs on the graph G after all edges not incident on C have been deleted. We can easily show that this does not increase S. First, by (5.11) each $X(v,i_C)$ is a nonincreasing function of $\{P_i : i < i_C\}$. Second, by (5.12) and (5.13), S is a nondecreasing function of $\{X(v,i_C) : v \in C\}$. Hence, S is a nonincreasing function of $\{P_i : i < i_C\}$. Now, the edge deletion forces $P_i = 1$ for all $i < i_C$, since any $\pi_i : i < i_C$ has no neighbor $\pi_j : j < i$. Thus the edge deletion does not increase S (and, obviously, $\mathbb{E}[S]$). We can then assume $G[V\setminus C]$ is an independent set. At this point, any node not adjacent to C is isolated and can be ignored. We can thus restrict the analysis to C and its neighborhood in G. Therefore we let $\bar{C} = \{v : \{u,v\} \in E,\, u \in C, v \notin C\}$ denote both the neighborhood and the complement of C.

We turn to bounding $\mathbb{E}[S]$. For now we assume $G[C]$ is a clique; we will then relax the assumption to $|E_C| \geq (1-\epsilon)\binom{C}{2}$. Since by hypothesis $\text{cut}(C,\bar{C}) < \epsilon C^2$, the average degree of the nodes in $\bar{C}$ is less than $\epsilon C^2/\bar{C}$. This is also a bound on the expected number of edges between C and a node drawn u.a.r. from $\bar{C}$. But, for any given i, conditioned on $i_C - 1 = i$ the nodes $\pi_1,\ldots,\pi_{i_C-1}$ are indeed drawn u.a.r. from $\bar{C}$, and so have a total of at most $i\epsilon C^2/\bar{C}$

edges towards C in expectation. Thus, over the distribution of π, the expected number of edges between C and $\pi_1, \ldots, \pi_{i_C - 1}$ is at most:

$$\sum_{i=0}^{n} \frac{i\epsilon C^2}{\bar{C}} \mathbb{P}(i_C - 1 = i) = \frac{\epsilon C^2}{\bar{C}} \mathbb{E}[i_C - 1] = \frac{\epsilon C^2}{\bar{C}} \frac{\bar{C}}{C+1} < \epsilon C \tag{5.14}$$

where we used the fact that $\mathbb{E}[i_C - 1] = \bar{C}/(C+1)$. Now note that (5.14) is a bound on $C - \mathbb{E}[S_C]$, the expected number of nodes of C that are adjacent to $\pi_1, \ldots, \pi_{i_C - 1}$. Therefore, $\mathbb{E}[S_C] \geq (1-\epsilon)C$.

Recall that P_{i_C} indicates whether π_{i_C} is not adjacent to any of $\pi_1, \ldots, \pi_{i_C - 1}$. Since the distribution of π_{i_C} is uniform over C, $\mathbb{P}(P_{i_C} \mid S_C) = S_C/C$. But $S = P_{i_C} S_C$, hence $\mathbb{E}[S \mid S_C] = (S_C)^2/C$, and thus $\mathbb{E}[S] = \mathbb{E}\big[(S_C)^2\big]/C$. Using $\mathbb{E}[S_C] \geq (1-\epsilon)C$ and invoking Jensen's inequality we obtain

$$\mathbb{E}[S] \geq \frac{\mathbb{E}[S_C]^2}{C} \geq (1-\epsilon)^2 C \geq (1-2\epsilon)C \tag{5.15}$$

which is our bound on $\mathbb{E}\big[|C \cap \hat{C}|\big]$ for KwikCluster.

Let us now move to ACC. We have to take into account the facts that ACC performs $f(|G_r|-1)$ queries on the pivot before deciding whether to perform $|G_r|-1$ queries, and that ACC stops after $f(n-1)$ rounds. We start by addressing the first issue, assuming for the moment ACC has no restriction on the number of rounds.

Recall that $\mathbb{P}(P_{i_C} \mid S_C) = S_C/C$. Now, if $P_{i_C} = 1$, then we have $S_C - 1$ edges incident on π_{i_C}. It is easy to check that, if $n_r + 1$ is the number of nodes at the round when π_{i_C} is used, then the probability that ACC finds some edge incident on π_{i_C} is at least:

$$1 - \left(1 - \frac{S_C - 1}{n_r}\right)^{\lceil f(n_r) \rceil} \geq 1 - e^{-f(n_r)\frac{S_C - 1}{n_r}} \geq 1 - e^{-f(n)\frac{S_C - 1}{n}} \tag{5.16}$$

and, if this event occurs, then $S = S_C$. Thus

$$\mathbb{E}[S \mid S_C] = \mathbb{P}(P_{i_C} \mid S_C) S_C \geq \left(1 - e^{-f(n)\frac{S_C - 1}{n}}\right) \frac{S_C^2}{C} \geq \frac{S_C^2}{C} - S_C \frac{2n}{f(n)C} \tag{5.17}$$

where we used the facts that for $S_C \leq 1$ the middle expression in (5.17) vanishes, that $e^{-x} < 1/x$ for $x > 0$, and that $1/x < 2/(x+1)$ for all $x \geq 2$. Simple manipulations, followed by Jensen's inequality and an application of $\mathbb{E}[S_C] \geq (1-\epsilon)C$, give

$$\mathbb{E}[S] \geq (1-\epsilon)^2 C - (1-\epsilon)C\frac{2n}{f(n)C} \geq (1-2\epsilon)C - 2\frac{n}{f(n)} \tag{5.18}$$

We next generalize the bound to the case $E_C \geq (1-\epsilon)\binom{C}{2}$. To this end note that, since at most $\epsilon\binom{C}{2}$ edges are missing from any subset of C, then any subset of S_C nodes of C has average degree at least

$$\max\left\{0, S_C - 1 - \binom{C}{2}\frac{2\epsilon}{S_C}\right\} \geq S_C - \frac{\epsilon C(C-1)}{2S_C} - 1 \tag{5.19}$$

We can thus re-write (5.17) as

$$\mathbb{E}[S \mid S_C] \geq \frac{S_C}{C}\left(1 - e^{-f(n)\frac{S_C - 1}{n}}\right)\left(S_C - \frac{\epsilon C(C-1)}{2S_C}\right) \tag{5.20}$$

Standard calculations show that this expression is bounded from below by $\frac{S_C^2}{C} - S_C\frac{2n}{f(n)C} - \frac{\epsilon C}{2}$, which by calculations akin to the ones above leads to $\mathbb{E}[S] \geq (1 - \frac{5}{2}\epsilon)C - 2\frac{n}{f(n)}$.

Similarly, we can show that $\mathbb{E}[S] \geq \left(\frac{f(n)}{n} - \frac{5}{2}\epsilon\right)C$. To this end note that when ACC pivots on π_{i_C} all the remaining cluster nodes are found with probability at least $\frac{f(n)}{n}$ (this includes the cases $S_C \leq 1$, when such a probability is indeed 1). In (5.17), we can then replace

$1 - e^{-f(n)\frac{S_C - 1}{n}}$ with $\frac{f(n)}{n}$, which leads to $\mathbb{E}[S] \geq \big(\frac{f(n)}{n} - \frac{5}{2}\epsilon\big)C$. This proves the first inequality in (5.9).

For the second inequality in (5.9), note that any subset of S_C nodes has $\operatorname{cut}(C, \bar{C}) \leq \epsilon\binom{C}{2}$. Thus, π_{i_C} is be incident to at most $\frac{\epsilon}{S_C}\binom{C}{2}$ such edges in expectation. The expected number of nodes of $\bar{C}$ that ACC assigns to $\hat{C}$, as a function of S_C, can thus be bounded by $\frac{S_C}{C}\frac{\epsilon}{S_C}\binom{C}{2} < \frac{\epsilon}{2}C$.

As far as the $\mathcal{O}(Ce^{-Cf(n)/n})$ part of the bound is concerned, simply note that the bounds obtained so far hold unless $i_C > \lceil f(n-1) \rceil$, in which case ACC stops before ever reaching the first node of C. If this happens, $\hat{C} = \{\pi_{i_C}\}$ and $|\hat{C} \oplus C| < |C|$. The event $i_C > \lceil f(n-1) \rceil$ is the event that no node of C is drawn when sampling $\lceil f(n-1) \rceil$ nodes from V without replacement. We can therefore apply Chernoff-type bounds to the random variable X counting the number of draws of nodes of C and get $\mathbb{P}\big(X < (1-\beta)\mathbb{E}[X]\big) \leq \exp(-\beta^2\mathbb{E}[X]/2\big)$ for all $\beta > 0$. In our case $\mathbb{E}[X] = \lceil f(n-1) \rceil |C|/n$, and we have to bound the probability that X equals $0 < (1-\beta)\mathbb{E}[X]$. Thus

$$\mathbb{P}(X = 0) \leq \exp\left(-\frac{\beta^2\mathbb{E}[X]}{2}\right) = \exp\left(-\frac{\beta^2\lceil f(n-1) \rceil |C|}{2n}\right)$$

Note however that $\lceil f(n-1) \rceil \geq f(n)/2$ unless $n = 1$ (in which case V is trivial). Then, choosing e.g. $\beta > \sqrt{4/5}$ yields $\mathbb{P}(X = 0) < \exp\big(-|C|f(n)/5n\big)$. This case therefore adds at most $|C|\exp(-|C|f(n)/5n)$ to $\mathbb{E}[|\hat{C} \oplus C|]$.

5.C.2 Proof of Theorem 5

Before moving to the actual proof, we need some ancillary results. The next lemma bounds the probability that ACC does not pivot on a node of C in the first k rounds.

Lemma 5.14. *Fix a subset $C \subseteq V$ and an integer $k \geq 1$, and let $\pi_1, \dots, \pi_n$ be a random permutation of V. For any $v \in C$ let $X_v = \{v \in \{\pi_1, \dots, \pi_k\}\}$, and let $X_C = \sum_{v \in C} X_v$. Then $\mathbb{E}[X_C] = \frac{k|C|}{n}$, and $\mathbb{P}(X_C = 0) < e^{-\frac{k|C|}{3n}}$.*

Proof. Since π is a random permutation, then for each $v \in C$ and each each $i = 1, \dots, k$ we have $\mathbb{P}(\pi_i = v) = \frac{1}{n}$. Therefore $\mathbb{E}[X_v] = \frac{k}{n}$ and $\mathbb{E}[X_C] = \frac{k|C|}{n}$. Now, the process is exactly equivalent to sampling without replacement from a set of n items of which $|C|$ are marked. Therefore, the X_v's are non-positively correlated and we can apply standard concentration bounds for the sum of independent binary random variables. In particular, for any $\eta \in (0,1)$ we have:

$$\mathbb{P}(X_C = 0) \leq \mathbb{P}(X_C < (1-\eta)\mathbb{E}[X_C]) < \exp\Big(-\frac{\eta^2\mathbb{E}[X_C]}{2}\Big)$$

which drops below $e^{-\frac{k|C|}{3n}}$ by replacing $\mathbb{E}[X_C]$ and choosing $\eta \geq \sqrt{2/3}$. $\square$

The next lemma is the crucial one.

Lemma 5.15. *Let $\epsilon \leq \frac{1}{10}$. Consider a strongly $(1-\epsilon)$-knit set C with $|C| > \frac{10n}{f(n)}$. Let $u_C = \min\{v \in C\}$ be the id of C. Then, for any $v \in C$, in any single run of* ACC *we have $\mathbb{P}(\operatorname{id}(v) = u_C) \geq \frac{2}{3}$.*

Proof. We bound from above the probability that any of three "bad" events occurs. As in the proof of Theorem 4, we equivalently see ACC as going through a sequence of candidate pivots $\pi_1, \dots, \pi_n$ that is a uniform random permutation of V. Let $i_C = \min\{i : \pi_i \in C\}$ be the index of the first node of C in the random permutation of candidate pivots. The first event, B_1, is $\{i_C > \lceil f(n-1) \rceil\}$. Note that, if B_1 does not occur, then ACC will pivot on π_{i_C}. The second event, B_2, is the event that $\pi_{i_C} \in V_{sing}$ if ACC pivots on π_{i_C} (we measure the probability of B_2 conditioned on $\bar{B}_1$). The third event, B_3, is $\{\pi_{i_C} \notin P\}$ where $P = \mathcal{N}_{u_C} \cap \mathcal{N}_v$. If none among B_1, B_2, B_3 occurs, then ACC forms a cluster $\hat{C}$ containing both u_C and v, and by the min-tagging rule sets $\operatorname{id}(v) = \min_{u \in \hat{C}} = u_C$. We shall then show that $\mathbb{P}(B_1 \cup B_2 \cup B_3) \leq 1/3$.

For B_1, we apply Lemma 5.14 by observing that $i_C > \lceil f(n-1) \rceil$ corresponds to the event $X_C = 0$ with $k = \lceil f(n-1) \rceil$. Thus

$$\mathbb{P}(i_C > \lceil f(n-1) \rceil) < e^{-\frac{\lceil f(n-1) \rceil |C|}{3n}} \leq e^{-\frac{f(n-1)}{3n}\frac{10\,n}{f(n)}} = e^{-\frac{f(n-1)}{f(n)}\frac{10}{3}} < e^{-3}$$

where we used the fact that $n \geq |C| \geq 11$ and therefore $f(n-1) \geq \frac{10}{11} f(n)$.

For B_2, recall that by definition every $v \in C$ has at least $(1-\epsilon)c$ edges. By the same calculations as the ones above, if ACC pivots on π_{i_C}, then:

$$\mathbb{P}(\pi_{i_C} \in V_{sing}) \leq \exp\Big(-\frac{f(n-1)}{n-1}(1-\epsilon)c\Big) \leq \exp\Big(-\frac{f(n-1)}{n-1}\big(1-\frac{1}{10}\big)\frac{10\,n}{f(n)}\Big) \leq e^{-9}$$

For B_3, note that the distribution of π_{i_C} is uniform over C. Now, let $\mathcal{N}_{u_C}$ and $\mathcal{N}_v$ be the neighbor sets of u_C and v in C, and let $P = \mathcal{N}_{u_C} \cap \mathcal{N}_v$. We call P the set of good pivots. Since C is strongly $(1-\epsilon)$-knit, both u_C and v have at least $(1-\epsilon)c$ neighbors in C. But then $|C \setminus P| \leq 2\epsilon c$ and

$$\mathbb{P}(\pi_{i_C} \notin P) = \frac{|C \setminus P|}{|C|} \leq 2\epsilon \leq 1/5$$

By a union bound, then, $\mathbb{P}(B_1 \cup B_2 \cup B_3) \leq e^{-3} + e^{-9} + 1/5 < 1/3$. □

We are now ready to conclude the proof. Suppose we execute ACC independently $K = 48\lceil \ln(n/p) \rceil$ times with the min-tagging rule. For a fixed $v \in G$ let X_v be the number of executions giving $\mathrm{id}(v) = u_C$. On the one hand, by Lemma 5.15, $\mathbb{E}[X_v] \geq \frac{2}{3}K$. On the other hand, v will not be assigned to the cluster with id u_C by the majority voting rule only if $X_v \leq \frac{1}{2}K \leq \mathbb{E}[X_v](1-\delta)$ where $\delta = \frac{1}{4}$. By standard concentration bounds, then, $\mathbb{P}(X_v \leq \frac{1}{2}K) \leq \exp(-\frac{\delta^2 \mathbb{E}[X_v]}{2}) = \exp(-\frac{K}{48})$. By setting $K = 48\ln(n/p)$, the probability that v is not assigned id u_C is thus at most p/n. A union bound over all nodes concludes the proof.

5.D Supplementary Material for Section 6

5.D.1 Proof of Theorem 8

We prove that there exists a distribution over labelings σ with OPT $= 0$ on which any deterministic algorithm has expected cost at least $\frac{n\varepsilon^2}{8}$. Yao's minimax principle then implies the claimed result.

Given $V = \{1, \ldots, n\}$, we define σ by a random partition of the vertices in $d \geq 2$ isolated cliques $T_1, \ldots, T_d$ such that $\sigma(v, v') = +1$ if and only if v and v' belong to the same clique. The cliques are formed by assigning each node $v \in V$ to a clique I_v drawn uniformly at random with replacement from $\{1, \ldots, d\}$, so that $T_i = \{v \in V : I_v = i\}$. Consider a deterministic algorithm making queries $\{s_t, r_t\} \in \mathcal{E}$. Let E_i be the event that the algorithm never queries a pair of nodes in T_i with $|T_i| \geq \frac{n}{2d} > 5$. Apply Lemma 5.16 below with $d = \frac{1}{\varepsilon}$. This implies that the expected number of non-queried clusters of size at least $\frac{n}{2d}$ is at least $\frac{d}{2} = \frac{1}{2\varepsilon}$. The overall expected cost of ignoring these clusters is therefore at least

$$\frac{d}{2}\left(\frac{n}{2d}\right)^2 = \frac{n^2}{8d} = \frac{\varepsilon n^2}{8}$$

and this concludes the proof.

Lemma 5.16. *Suppose $d > 0$ is even, $n \geq 16d\ln d$, and $B < \frac{d^2}{50}$. Then for any deterministic learning algorithm making at most B queries,*

$$\sum_{i=1}^{d} \mathbb{P}(E_i) > \frac{d}{2} .$$

Proof. For each query $\{s_t, r_t\}$ we define the set L_t of all cliques T_i such that $s_t \notin T_i$ and some edge containing both s_t and a node of T_i was previously queried. The set R_t is defined similarly using r_t. Formally,

$$\begin{aligned} L_t &= \{i \,:\, (\exists \tau < t)\; s_\tau = s_t \wedge r_\tau \in T_i \wedge \sigma(s_\tau, r_\tau) = -1\} \\ R_t &= \{i \,:\, (\exists \tau < t)\; r_\tau = r_t \wedge s_\tau \in T_i \wedge \sigma(s_\tau, r_\tau) = -1\} \ . \end{aligned}$$

Let D_t be the event that the t-th query discovers a new clique of size at least $\frac{n}{2d}$, and let $P_t = \max\bigl\{|L_t|, |R_t|\bigr\}$. Using this notation,

$$\sum_{t=1}^{B} \{D_t\} = \sum_{t=1}^{B} \{D_t \,\wedge\, P_t < d/2\} + \underbrace{\sum_{t=1}^{B} \{D_t \,\wedge\, P_t \geq d/2\}}_{N} \; . \tag{5.21}$$

We will now show that unless $B \geq \frac{d^2}{50}$, we can upper bound N deterministically by $\sqrt{2B}$.

Suppose $N > \frac{d}{2}$, and let $t_1, \dots, t_N$ be the times t_k such that $\{D_{t_k} \,\wedge\, P_{t_k} \geq d/2\} = 1$. Now fix some k and note that, because the clique to which s_{t_k} and r_{t_k} both belong is discovered, neither s_{t_k} nor r_{t_k} can occur in a future query $\{s_t, r_t\}$) that discovers a new clique. Therefore, in order to have $\{D_t \,\wedge\, P_t \geq d/2\} = 1$ for $N > \frac{d}{2}$ times, at least

$$\binom{N}{2} \geq \frac{d^2}{8}$$

queries must be made, since each one of the other $N - 1 \geq \frac{d}{2}$ discovered cliques can contribute with at most a query to making $P_t \geq \frac{d}{2}$. So, it takes at least $B \geq \frac{d^2}{8}$ queries to discover the first $\frac{d}{2}$ cliques of size at least two, which contradicts the lemma's assumption that $B \leq \frac{d^2}{16}$. Therefore, $N \leq \frac{d}{2}$.

Using the same logic as before, in order to have $\{D_t \,\wedge\, P_t \geq d/2\} = 1$ for $N \leq \frac{d}{2}$ times, at least

$$\frac{d}{2} + \left(\frac{d}{2} - 1\right) + \cdots + \left(\frac{d}{2} - N + 1\right)$$

queries must be made. So, it must be

$$B \geq \sum_{k=1}^{N} \left(\frac{d}{2} - (k-1)\right) = (d+1)\frac{N}{2} - \frac{N^2}{2}$$

or, equivalently, $N^2 - (d+1)N + 2B \geq 0$. Solving this quadratic inequality for N, and using the hypothesis $N \leq \frac{d}{2}$, we have that $N \leq \frac{(d+1)-\sqrt{(d+1)^2-8B}}{2}$. Using the assumption that $B \leq \frac{d^2}{50}$ we get that $N \leq \sqrt{2B}$.

We now bound the first term of (5.21) in expectation. The event D_t is equivalent to $s_t, r_t \in T_i$ for some $i \in \neg L_t \cap \neg R_t$, where for any $S \subseteq \{1, \dots, d\}$ we use $\neg S$ to denote $\{1, \dots, d\} \setminus S$.

Let $\mathbb{P}_t = \mathbb{P}\bigl(\cdot \mid P_t < d/2\bigr)$. For L', R' ranging over all subsets of $\{1, \dots, d\}$ of size strictly less than $\frac{d}{2}$,

$$\begin{aligned}
\mathbb{P}_t(D_t) &= \sum_{L',R'} \sum_{i \in \neg L' \cap \neg R'} \mathbb{P}_t\bigl(s_t \in T_i \,\wedge\, r_t \in T_i \bigm| L_t = L',\, R_t = R'\bigr)\, \mathbb{P}_t(L_t = L' \,\wedge\, R_t = R') \\
&= \sum_{L',R'} \sum_{i \in \neg L' \cap \neg R'} \mathbb{P}_t\bigl(s_t \in T_i \bigm| L_t = L'\bigr)\, \mathbb{P}_t\bigl(r_t \in T_i \bigm| R_t = R'\bigr)\, \mathbb{P}_t(L_t = L' \,\wedge\, R_t = R') \qquad (5.22) \\
&= \sum_{L',R'} \sum_{i \in \neg L' \cap \neg R'} \frac{1}{|\neg L'|}\, \frac{1}{|\neg R'|}\, \mathbb{P}_t(L_t = L' \,\wedge\, R_t = R') \qquad (5.23) \\
&= \sum_{L',R'} \frac{|\neg L' \cap \neg R'|}{|\neg L'|\,|\neg R'|}\, \mathbb{P}_t(L_t = L' \,\wedge\, R_t = R') \\
&\leq \frac{2}{d} \; . \qquad (5.24)
\end{aligned}$$

Equality (5.22) holds because $P_t = \max\{L_t, R_t\} < \frac{d}{2}$ implies that there are at least two remaining cliques to which s_t and r_t could belong, and each node is independently assigned to one of these cliques. Equality (5.23) holds because, by definition of L_t, the clique of s_t is not in L_t, and there were no previous queries involving s_t and a node belonging to a clique in $\neg L_t$ (similarly for r_t). Finally, (5.24) holds because $|\neg L'| \geq \frac{d}{2}$, $|\neg R'| \geq \frac{d}{2}$, and $|\neg L' \cap \neg R'| \leq \min\{|\neg L'|, |\neg R'|\}$. Therefore,

$$\sum_{t=1}^{B} \mathbb{P}\big(D_t \wedge P_t < d/2\big) \leq \sum_{t=1}^{B} \mathbb{P}\big(D_t \mid P_t < d/2\big) \leq \frac{2B}{d} \ .$$

Putting everything together,

$$\mathbb{E}\left[\sum_{t=1}^{B} \{D_t\}\right] \leq \frac{2B}{d} + \sqrt{2B} \ . \tag{5.25}$$

On the other hand, we have

$$\sum_{t=1}^{B} \{D_t\} = \sum_{i=1}^{d} \Big(\{|T_i| \geq \tfrac{n}{2d}\} - \{E_i\} \Big) = d - \sum_{i=1}^{d} \Big(\{|T_i| < \tfrac{n}{2d}\} + \{E_i\} \Big) \tag{5.26}$$

Combining (5.25) and (5.26), we get that

$$\sum_{i=1}^{d} \mathbb{P}(E_i) \geq d - \sum_{i=1}^{d} \mathbb{P}\big(|T_i| < \tfrac{n}{2d}\big) - \frac{2B}{d} - \sqrt{2B} \ .$$

By Chernoff-Hoeffding bound, $\mathbb{P}\big(|T_i| < \frac{n}{2d}\big) \leq \frac{1}{d^2}$ for each $i = 1, \dots, d$ when $n \geq 16 d \ln d$. Therefore,

$$\sum_{i=1}^{d} \mathbb{P}(E_i) \geq d - \frac{2B+1}{d} - \sqrt{2B} \ .$$

To finish the proof, suppose on the contrary that $\sum_{i=1}^{d} \mathbb{P}(E_i) \leq \frac{d}{2}$. Then from the inequality above, we would get that

$$\frac{d}{2} \geq d - \frac{2B+1}{d} - \sqrt{2B}$$

which implies $B \geq \left(\frac{2-\sqrt{2}}{4}\right)^2 d^2 > \frac{d^2}{50}$, contradicting the assumptions. Therefore, we must have $\sum_{i=1}^{d} \mathbb{P}(E_i) > \frac{d}{2}$ as required. □

5.D.2 Proof of Theorem 9

Choose a suitably large n and let $V = [n]$. We partition V in two sets A and B, where $|A| = \alpha n$ and $|B| = (1-\alpha)n$; we will eventually set $\alpha = 0.9$, but for now we leave it free to have a clearer proof. The set A is itself partitioned into $k = 1/\epsilon$ subsets $A_1, \dots, A_k$, each one of equal size $\alpha n / k$ (the subsets are not empty because of the assumption on ε). The labeling σ is the distribution defined as follows. For each $i = 1, \dots, k$, for each pair $u, v \in A_i$, $\sigma(u,v) = +1$; for each $u, v \in B$, $\sigma(u,v) = -1$. Finally, for each $v \in B$ we have a random variable i_v distributed uniformly over $[k]$. Then, $\sigma(u,v) = +1$ for all $u \in A_{i_v}$ and $\sigma(u,v) = -1$ for all $u \in A \setminus A_{i_v}$. Note that the distribution of i_v is independent of the (joint) distributions of the i_w's for all $w \in B \setminus \{v\}$.

Let us start by giving an upper bound on $\mathbb{E}[\mathrm{OPT}]$. To this end consider the (possibly suboptimal) clustering $\mathcal{C} = \{C_i : i \in [k]\}$ where $C_i = A_i \cup \{v \in B : i_v = i\}$. One can check that $\mathcal{C}$ is a partition of V. The expected cost $\mathbb{E}[\Delta_{\mathcal{C}}]$ of $\mathcal{C}$ can be bound as follows. First, note the only mistakes are due to pairs $u, v \in B$. However, for any such fixed pair u, v, the probability of a mistake (taken over σ) is $\mathbb{P}(i_u \neq i_v) = 1/k$. Thus,

$$\mathbb{E}[\mathrm{OPT}] \leq \mathbb{E}[\Delta_0] < \frac{|B|^2}{k} = \frac{(1-\alpha)^2 n^2}{k} \tag{5.27}$$

Let us now turn to the lower bound on the expected cost of the clustering produced by an algorithm. For each $v \in B$ let Q_v be the total number of distinct queries the algorithm makes to pairs $\{u, v\}$ with $u \in A$ and $v \in B$. Let Q be the total number of queries made by the algorithm; obviously, $Q \geq \sum_{v \in B} Q_v$. Now let S_v be the indicator variable of the event that one of the queries involving v returned $+1$. Both Q_v and S_v as random variables are a function of the input distribution and of the choices of the algorithm. The following is key:

$$\mathbb{P}(S_v \wedge Q_v < {}^{k}\!/_{2}) < \frac{1}{2} \tag{5.28}$$

The validity of (5.28) is seen by considering the distribution of the input limited to the pairs $\{u, v\}$. Indeed, $S_v \wedge Q_v < {}^{k}\!/_{2}$ implies the algorithm discovered the sole positive pair involving v in less than $k/2$ queries. Since there are k pairs involving v, and for any fixed j the probability (taken over the input) that the algorithm finds that particular pair on the j-th query is exactly $1/k$. Now,

$$\mathbb{P}(S_v \wedge Q_v < {}^{k}\!/_{2}) + \mathbb{P}(\bar{S}_v \wedge Q_v < {}^{k}\!/_{2}) + \mathbb{P}(Q_v \geq {}^{k}\!/_{2}) = 1 \tag{5.29}$$

and therefore

$$\mathbb{P}(\bar{S}_v \wedge Q_v < {}^{k}\!/_{2}) + \mathbb{P}(Q_v \geq {}^{k}\!/_{2}) > \frac{1}{2} \tag{5.30}$$

Let us now consider R_v, the number of mistakes involving v made by the algorithm. We analyse $\mathbb{E}[R_v \,|\, \bar{S}_v \wedge Q_v < {}^{k}\!/_{2}]$. For all $i \in [k]$ let Q_v^i indicate the event that, for some $u \in A_i$, the algorithm queried the pair $\{u, v\}$. Let $I = \{i \in [k] : Q_v^i = 0\}$; thus I contains all i such that the algorithm did not query any pair u, v with $u \in A_i$. Suppose now the event $\bar{S}_v \wedge Q_v < {}^{k}\!/_{2}$ occurs. On the one hand, $\bar{S}_v$ implies that:

$$\mathbb{P}(\sigma(u, v) = +1 \mid I) = \begin{cases} {}^{1}\!/_{|I|} & u \in A_i, i \in I \\ 0 & u \in A_i, i \in [k] \setminus I \end{cases} \tag{5.31}$$

Informally speaking, this means that the random variable i_v is distributed uniformly over the (random) set I. Now observe that, again conditioning on the joint event $\bar{S}_v \wedge Q_v < {}^{k}\!/_{2}$, whatever label s the algorithm assigns to a pair u, v with $u \in A_i$ where $i \in I$, the distribution of $\sigma(u, v)$ is independent of s. This holds since s can obviously be a function only of I and of the queries made so far, all of which returned -1, and possibly of the algorithm's random bits. In particular, it follows that:

$$\mathbb{P}(\sigma(u, v) \neq s \mid I) \geq \min\left\{{}^{1}\!/_{|I|}, 1 - {}^{1}\!/_{|I|}\right\} \tag{5.32}$$

However, $Q_v < {}^{k}\!/_{2}$ implies that $|I| \geq k - Q_v > {}^{k}\!/_{2} = {}^{2}\!/_{\epsilon} > 2$, which implies $\min\{{}^{1}\!/_{|I|}, 1 - {}^{1}\!/_{|I|}\} \geq {}^{1}\!/_{|I|}$. Therefore, $\mathbb{P}(\sigma(u, v) \neq s \mid I) \geq {}^{1}\!/_{|I|}$ for all $u \in A_i$ with $i \in I$.

We can now turn to back to R_v, the number of total mistakes involving v. Clearly, $R_v \geq \sum_{i=1}^{k} \sum_{u \in A_i} \{\sigma(u, v) \neq s\}$. Then:

$$\begin{aligned}
\mathbb{E}[R_v \,|\, E] &= \mathbb{E}\Big[\sum_{i=1}^{k} \sum_{u \in A_i} \{\sigma(u, v) \neq s\} \;\Big|\; \bar{S}_v \wedge Q_v < {}^{k}\!/_{2}\Big] && (5.33)\\
&= \mathbb{E}\Big[\mathbb{E}\Big[\sum_{i=1}^{k} \sum_{u \in A_i} \{\sigma(u, v) \neq s\} \;\Big|\; I\Big] \;\Big|\; \bar{S}_v \wedge Q_v < {}^{k}\!/_{2}\Big] && (5.34)\\
&\geq \mathbb{E}\Big[\mathbb{E}\Big[\sum_{i \in I} \sum_{u \in A_i} \{\sigma(u, v) \neq s\} \;\Big|\; I\Big] \;\Big|\; \bar{S}_v \wedge Q_v < {}^{k}\!/_{2}\Big] && (5.35)\\
&\geq \mathbb{E}\Big[\mathbb{E}\Big[\sum_{i \in I} \sum_{u \in A_i} \frac{1}{|I|} \;\Big|\; I\Big] \;\Big|\; \bar{S}_v \wedge Q_v < {}^{k}\!/_{2}\Big] && (5.36)\\
&= \mathbb{E}\Big[\mathbb{E}\Big[\frac{\alpha n}{k}\Big] \;\Big|\; \bar{S}_v \wedge Q_v < {}^{k}\!/_{2}\Big] && (5.37)\\
&= \frac{\alpha n}{k} && (5.38)
\end{aligned}$$

And therefore:

$$\begin{aligned}\mathbb{E}[R_v] &\geq \mathbb{E}[R_v \,|\, \bar{S}_v \,\wedge\, Q_v < {}^{k}\!/_{2}] \cdot \mathbb{P}(\bar{S}_v \,\wedge\, Q_v < {}^{k}\!/_{2}) \\ &> \frac{\alpha n}{k} \cdot \mathbb{P}(\bar{S}_v \,\wedge\, Q_v < {}^{k}\!/_{2})\end{aligned}$$

This concludes the bound on $\mathbb{E}[R_v]$. Let us turn to $\mathbb{E}[Q_v]$. Just note that:

$$\mathbb{E}[Q_v] \geq \frac{k}{2} \cdot \mathbb{P}(Q_v \geq {}^{k}\!/_{2}) \tag{5.39}$$

By summing over all nodes, we obtain:

$$\mathbb{E}[Q] \geq \sum_{v \in B} \mathbb{E}[Q_v] \geq \frac{k}{2}\Big(\sum_{v \in B} \mathbb{P}(Q_v \geq {}^{k}\!/_{2})\Big) \tag{5.40}$$

$$\mathbb{E}[\Delta] \geq \sum_{v \in B} \mathbb{E}[R_v] > \frac{\alpha n}{k}\Big(\sum_{v \in B} \mathbb{P}(\bar{S}_v \,\wedge\, Q_v < {}^{k}\!/_{2})\Big) \tag{5.41}$$

to which, by virtue of (5.30), applies the constraint:

$$\Big(\sum_{v \in B} \mathbb{P}(Q_v \geq {}^{k}\!/_{2})\Big) + \Big(\sum_{v \in B} \mathbb{P}(\bar{S}_v \,\wedge\, Q_v < {}^{k}\!/_{2})\Big) > |B|\frac{1}{2} = \frac{(1-\alpha)n}{2} \tag{5.42}$$

This constrained system gives the bound. Indeed, by (5.40), (5.41) and (5.42), it follows that if $\mathbb{E}[Q] < \frac{k}{2}\frac{(1-\alpha)n}{4} = \frac{(1-\alpha)nk}{8}$ then $\mathbb{E}[\Delta] > \frac{\alpha n}{k}\frac{(1-\alpha)n}{2} = \frac{\alpha(1-\alpha)n^2}{4k}$. It just remains to set α and k properly so to get the statement of the theorem.

Let $\alpha = {}^{9}\!/_{10}$ and recall that $k = 1/\epsilon$. Then, first, $\frac{(1-\alpha)nk}{8} = \frac{nk}{80} = \frac{n}{80\,\epsilon}$. Second, (5.27) gives $\mathbb{E}[\mathrm{OPT}] < \frac{(1-\alpha)^2 n^2}{k} = \frac{n^2}{100k} = \frac{\epsilon n^2}{100}$. Third, $\frac{\alpha(1-\alpha)n^2}{4k} = \frac{9n^2}{400k} = \frac{9\epsilon n^2}{400} > \mathbb{E}[\mathrm{OPT}] + \frac{\epsilon n^2}{80}$. The above statement hence becomes: if $\mathbb{E}[Q] < \frac{n}{80\epsilon}$, then $\mathbb{E}[\Delta] > \mathbb{E}[\mathrm{OPT}] + \frac{\epsilon n^2}{80}$. An application of Yao's minimax principle completes the proof.

As a final note, we observe that for every $c \geq 1$ the bound can be put in the form $\mathbb{E}[\Delta] \geq c \cdot \mathbb{E}[\mathrm{OPT}] + \Omega(n^2\epsilon)$ by choosing $\alpha \geq c/(c + {}^{1}\!/_{4})$.

Chapter 6

Robust Unsupervised Learning via L-statistic Minimization

The scope of this chapter is to study (a notion) of robustness in the context of unsupervised learning. We start defining a notion of outliers and show how to leverage this notion by using a L-statistic minimization algorithm. This notion assumes that the perturbing distribution is characterized by larger losses relative to a given class of admissible models. We then analyze the proposed method and bound its reconstruction error relative to the underlying unperturbed distribution. As a byproduct, we prove uniform convergence bounds for several popular model classes in unsupervised learning, a result which may be of independent interest.

6.1 Introduction

Making learning methods robust is a fundamental problem in machine learning and statistics. In this chapter we proposes an approach to unsupervised learning which is resistant to unstructured contaminations of the underlying data distribution. As noted by Hampel [Hampel, 2001], "outliers" are an ill-defined concept, and an approach to robust learning, which relies on rules for the rejection of outliers [see Ord, 1996, and references therein] prior to processing may be problematic, since the hypothesis class of the learning process itself may determine which data is to be regarded as structured or unstructured. Instead of the elimination of outliers – quoting Hampel "data that don't fit the pattern set by the majority of the data" – in this chapter we suggest to restrict attention to "a *sufficient portion of the data in good agreement with one of the hypothesized models*".

To implement the above idea, we propose using L-estimators [Serfling, 1980], which are formed by a weighted average of the order statistics. That is, given a candidate model, we first rank its losses on the empirical data and than take a weighted average which emphasizes small losses more. An important example of this construction is the average of a fraction of the smallest losses. However, our observations apply to general classes of weight functions, which are only restricted to be non-increasing and in some cases Lipschitz continuous.

We highlight that although L-statistics have a long tradition, a key novelty is to use them as objective functions based on which to search for a robust model. This approach is general in nature and can be applied to robustify any learning method, supervised or unsupervised, based on empirical risk minimization. We focus on unsupervised learning, and our analysis includes KMEANS clustering, principal subspace analysis and sparse coding, among others.

6.1.1 Contributions

We make the following contributions:

- A theoretical analysis of the robustness of the proposed method (Theorem 1). Under the assumption that the data-distribution is a mixture of an unperturbed distribution adapted to our model class and a perturbing distribution, we identify conditions under which we can bound the reconstruction error, when the minimizer of the proposed objective trained from the perturbed distribution is tested on the unperturbed distribution.

- An analysis of generalization (Theorems 4–6). We give dimension-free uniform bounds in terms of Rademacher averages as well as a dimension- and variance-dependent uniform bounds in terms of covering numbers which can outperform the dimension-free bounds under favorable conditions.

- A meta-algorithm operating on the empirical objective which can be used whenever there is a descent algorithm for the underlying loss function (Theorem 9).

The chapter is organized as follows. In Section 6.2 we give a brief overview of unsupervised (representation) learning. In Sections 6.3 to 6.5 we present and analyze our method. In Section 6.6 we discuss an algorithm optimizing the proposed objective and in Section 6.7 we present numerical experiments with this algorithm for KMEANS clustering and principal subspace analysis, which indicate that the proposed method is promising. Proofs can be found in the supplementary material.

6.1.2 Previous Work

Some elements of our approach have a long tradition. For fixed models the proposed empirical objectives are called L-statistics or L-estimators. They have been used in robust statistics since the middle of the last century [Lloyd, 1952] and their asymptotic properties have been studied by many authors [see Serfling, 1980, and references therein]. Although influence functions play a certain role, our approach is somewhat different from the traditions of robust statistics. Similar techniques to ours have been experimentally explored in the context of classification [Han et al., 2018] or latent variable selection [Kumar et al., 2010]. [Cuesta-Albertos et al., 1997] give a special case of our method applied to k-means with hard threshold. The method is analyzed with the lenses of different robustness properties in [Garcia-Escudero and Gordaliza, 1999]. Finite sample bounds, uniform bounds, the minimization of L-statistics over model classes and the so called risk based-objectives however are more recent developments [Maurer and Pontil, 2018, 2019; Lee et al., 2020], and we are not aware of any other general bounds on the reconstruction error of models trained from perturbed data. A very different line of work for robust statistics are model-independent methods available in high dimensions [Elmore et al., 2006; Fraiman et al., 2019]. Although elegant and very general, these depth-related pre-processing methods may perform sub-optimally in practice, as our numerical experiments indicate. Similar data-generating assumption are adopted in Robust estimation, a related line of work where the goal is to identify the parameters of a target distribution up to a small error. In this context, strong parametric assumptions are made on the target distributions and the learning problems involves typically simpler model classes such as, mean and covariance estmations. In constrast, we focus allows for non-parametric distributions and more complex model classes as singletons, subspaces and linear operators. Please refer to [Diakonikolas and Kane, 2020] for an up-to-date survey on the results and the techniques employed to derive efficient algorithms for robust estimation. Finally, we note that previous work on PAC learning [e.g. Angluin and Laird, 1987] has addressed the problem of learning a good classifier with respect to a target, when the data comes from a perturbed distribution affected by unstructured noise. Similarly to us, they consider that the target distribution is well adapted to the model class.

6.2 Unsupervised Learning

Let $\mathcal{S}$ be a class of subsets of $\mathbb{R}^d$, which we call the model class. For $S \in \mathcal{S}$ define the distortion function $d_S : \mathbb{R}^d \to [0, \infty)$ by[1]

$$d_S(x) = \min_{y \in S} \|x - y\|^2 \text{ for } x \in \mathbb{R}^d. \tag{6.1}$$

We assume that the members of $\mathcal{S}$ are either compact sets or subspaces, so the minimum in (6.1) is always attained. For instance $\mathcal{S}$ could be the class of singletons, a class of subsets of cardinality k, the class of subspaces of dimension k, or a class of compact convex polytopes with k vertices[2].

[1] In most parts our analysis applies also to other distortion measures, for example omitting the square in (6.1). The chosen form is important for generalization bounds, when we want to bound the complexity of the class $\{x \mapsto d_S(x) : S \in \mathcal{S}\}$ for specific cases.

[2] In these cases the set S is the image of a linear operator on a prescribed set of code vectors, see [Maurer and Pontil, 2010]. Our setting is more general, e.g. it includes non-linear manifolds.

We write $\mathcal{P}(\mathcal{X})$ for the set of Borel probability measures on a locally compact Hausdorff space $\mathcal{X}$. If $\mu \in \mathcal{P}\left(\mathbb{R}^d\right)$, define the probability measure $\mu_S \in \mathcal{P}([0,\infty))$ as the push-forward of μ under d_S, that is, $\mu_S(A) = \mu(\{x : d_S(x) \in A\})$ for $A \subseteq [0,\infty)$. Now consider the functional $\Phi : \mathcal{P}([0,\infty)) \to [0,\infty)$ defined by

$$\Phi(\rho) = \int_0^\infty r d\rho(r), \quad \rho \in \mathcal{P}. \tag{6.2}$$

Then $\Phi(\mu_S) = \mathbb{E}_{X\sim\mu}[d_S(X)]$ is the expected *reconstruction error*, incurred when coding points by the nearest neighbors in S. The measures $\mu_S \in \mathcal{P}([0,\infty))$ and the functional Φ allow the compact and general description of several problems of unsupervised learning as

$$\min_{S\in\mathcal{S}} \Phi(\mu_S) = \min_{S\in\mathcal{S}} \mathbb{E}_{X\sim\mu}[d_S(X)]. \tag{6.3}$$

Denote with $S^* = S^*(\mu)$ a global minimizer of (6.3). Returning to the above examples, if $\mathcal{S}$ is the class of singleton sets, then $S^*(\mu)$ is the mean of μ. If it is the class of subsets of cardinality k, then $S^*(\mu)$ is the optimal set of centers for KMEANS clustering. If $\mathcal{S}$ is the class of k-dimensional subspaces, then $S^*(\mu)$ is the principal k-dimensional subspace.

An important drawback of the above formulation is that the functional Φ is very sensitive to perturbing masses at large distortions R. In the tradition of robust statistics [see e.g. Hampel, 1974; Serfling, 1980] this can be expressed in terms of the *influence function*, measuring the effect of an infinitesimal point mass perturbation of the data. Let δ_R be the unit mass at $R > 0$, then the influence function

$$\begin{aligned}\mathrm{IF}(R;\rho,\Phi) &:= \frac{d}{dt}\Phi\big((1-t)\rho + t\delta_R\big)\Big|_{t=0} \\ &= R - \Phi(\rho),\end{aligned} \tag{6.4}$$

can be arbitrarily large, indicating that even a single datapoint could already corrupt $S^*(\mu)$. To overcome this problem, in the next section we introduce a class of robust functional based on L-statistics.

6.3 Proposed Method

Our goal is to minimize the reconstruction error on unperturbed test data, from perturbed training data. Specifically, we assume that the data we observe comes from a perturbed distribution μ that is the mixture of an unperturbed distribution μ^*, which is locally concentrated on the minimizer $S^* = S^*(\mu^*)$, and a perturbing distribution ν which is unstructured in the sense that it does not concentrate on any of our models[3]. Figure 6.3.1 depicts such a situation, when $\mathcal{S}$ is the set of singletons and $d = 1$.

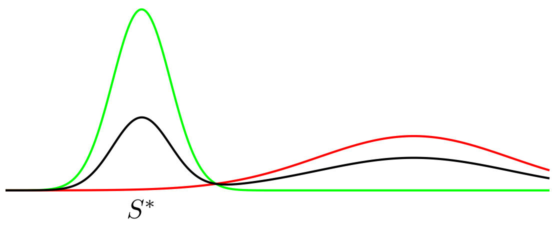


FIGURE 6.3.1: Densities of the unperturbed distribution μ^* (light green) with high local concentration on the optimal model S^*, the perturbing distribution ν (light red) without significant concentration, and the observable mixture $\mu = (1-\lambda)\mu^* + \lambda\nu^*$ (black) at $\lambda = 0.6$.

We wish to train from the available, perturbed data a model $\hat{S} \in \mathcal{S}$, which nearly minimizes the reconstruction error on the unperturbed distribution μ^*. To this end we exploit the

[3]This is in contrast with the assumptions made in adversarial learning, where the goal is to increase robustness against adversarial worst-case perturbations [see e.g. Lee and Raginsky, 2018].

assumption that the unperturbed distribution μ^* is much more strongly concentrated at S^* than the mixture $\mu = (1-\lambda)\,\mu^* + \lambda\nu$ is at models S *away* from S^* in terms of reconstruction error.

The key observation is that if the mixture parameter λ is not too large, the concentration of μ^* causes the cumulative distribution function of the losses for the optimal model $F_{\mu_{S^*}} : r \mapsto \mu_{S^*}[0,r]$ to increase rapidly for small values of r, until it reaches the value $\zeta = F_{\mu_{S^*}}(r^*)$, where r^* is a critical distortion radius depending on S^*. Thus, when searching for a model, we can consider as irrelevant the remaining mass $1-\zeta = \mu_{S^*}(r^*,\infty)$, which can be attributed to ν and may arise from outliers or other contaminating effects. To achieve this, we modify the functional (6.2) so as to consider only the relevant portion of data, replacing $\Phi(\mu_S)$ by

$$\zeta^{-1}\int_0^{F_{\mu_S}^{-1}(\zeta)} r d\mu_S(r)\,. \tag{6.5}$$

Intuitively, the minimization of (6.5) forces the search towards models with the smallest *truncated* expected loss. Among such models there is also S^*, whose losses have the strongest concentration around a *small* value and then leading to a very small value r^* for $F_{\mu_{S^*}}^{-1}(\zeta)$.

More generally, since the choice of the hard quantile-thresholding at ζ is in many ways an ad hoc decision, we might want a more gentle transition of the boundary between relevant and irrelevant data. Let $W : [0,1] \to [0,\infty)$ be a bounded weight function and define, for every $\rho \in \mathcal{P}[0,\infty)$,

$$\Phi_W(\rho) = \int_0^\infty rW(F_\rho(r))\,d\rho(r)\,.$$

We require W to be non-increasing and zero on $[\zeta,1]$ for some critical mass $\zeta < 1$. The parameter ζ must be chosen on the basis of an estimate of the amount λ of perturbing data. Note that if W is identically 1 then $\Phi_1 = \Phi$ in (6.2), while if $W = \zeta^{-1}1_{[0,\zeta]}$ then Φ_W is the hard thresholding functional in (6.5).

We now propose to “robustify" unsupervised learning by replacing the original problem (6.3) by

$$\min_{S\in\mathcal{S}} \Phi_W(\mu_S)\,, \tag{6.6}$$

and denote a global minimizer by $S^\dagger \equiv S^\dagger(\mu)$.

In practice, μ is unknown and the search for the model $S^\dagger$ has to rely on finite data. If $\hat{\mu}(\mathbf{X}) = \frac{1}{n}\sum_{i=1}^n \delta_{X_i}$ is the empirical measure induced by an i.i.d. sample $\mathbf{X} = (X_1,...,X_n) \sim \mu^n$, then the empirical objective is the plug-in estimate

$$\begin{aligned}&\Phi_W(\hat{\mu}(\mathbf{X})_S)\\ &= \frac{1}{n}\sum_{i=1}^n d_S(X_i)\,W\left(\frac{1}{n}\left|\{X_j : d_S(X_j) \le d_S(X_i)\}\right|\right)\\ &= \frac{1}{n}\sum_{i=1}^n d_S(X)_{(i)}\,W\left(\frac{i}{n}\right),\end{aligned} \tag{6.7}$$

where $d_S(X)_{(i)}$ is the i-th smallest member of $\{d_S(X_1),...,d_S(X_n)\}$.

The empirical estimate $\Phi_W(\hat{\mu}(\mathbf{X})_S)$ is an L-statistic [Serfling, 1980]. We denote a minimizer of this objective by

$$\hat{S}(\mathbf{X}) = \arg\min_{S\in\mathcal{S}} \Phi_W(\hat{\mu}(\mathbf{X})_S)\,. \tag{6.8}$$

In the sequel we study three questions:

1 If the underlying probability measure is a mixture $\mu = (1-\lambda)\,\mu^* + \lambda\nu$ of an unperturbed measure μ^* and a perturbing measure ν, and $S^\dagger = S^\dagger(\mu)$ is the minimizer of (6.6), under which assumptions will the reconstruction error $\Phi\left(\mu^*_{S^\dagger}\right)$ incurred by $S^\dagger$ on the unperturbed distribution approximate the minimal reconstruction error $\Phi(\mu^*_{S^*})$?

2 When solving (6.6) for a finite amount of data $\mathbf{X}$, under which conditions can we reproduce the behavior of $S^\dagger$ by the empirical minimizer $\hat{S}(\mathbf{X})$ in (6.8)?

3 How can the method be implemented and how does it perform in practice?

6.4 Resilience to Perturbations

Before we address the first question we make a preliminary observation in the tradition of robust statistics and compare the influence functions of the functional Φ_1 to that one of the proposed Φ_W with bounded W, and $W(t) = 0$ for $\zeta \leq t < 1$. While we saw in (6.4) that for any $\rho \in \mathcal{P}([0,\infty))$ the influence function $\mathrm{IF}(R;\rho,\Phi) = R - \Phi(\rho)$ is unbounded in R, in the case of Φ_W we have, for any $R \in \mathbb{R}^d$, that

$$\begin{aligned}\mathrm{IF}(R;\rho,\Phi_W) &\leq \mathrm{IF}_{\max}(\rho,W)\\ &:= \int_0^{F_\rho^{-1}(\zeta)} W(F_\rho(r))\,F_\rho(r)\,dr.\end{aligned}$$

Notice that the right hand side is always bounded, which already indicates the improved robustness of Φ_W [Hampel, 1974]. The upper bound $\mathrm{IF}_{\max}$ on the influence function plays also an important role in the subsequent analysis.

Returning now to the data generating mixture $\mu = (1-\lambda)\mu^* + \lambda\nu$, where $\mu^* \in \mathcal{P}(\mathbb{R}^d)$ is the the ideal, unperturbed distribution and $\nu \in \mathcal{P}(\mathbb{R}^d)$ the perturbation, we make the following assumption.

Assumption A. There exists $S_0 \in \mathcal{S}$, $\delta > 0$, $\beta \in (0, 1-\lambda)$ and a scale parameter $r^* > 0$ (in units of squared euclidean distance), such that for every model $S \in \mathcal{S}$ satisfying $\Phi(\mu_S^*) > \Phi(\mu_{S_0}^*) + \delta$ we have $F_{\mu_S}(r) < \beta F_{\mu_{S_0}^*}(r)$ for all $r \leq r^*$.

Assumption A does not depend so much on the richness of $\mathcal{S}$ (which will be relevant to generalization) but on the concentration properties of μ^* and ν (see the extreme example in the supplement). Loosely speaking the assumption prescribes that, under the perturbed distribution μ, any model S with a large reconstruction error on μ^*, should have its losses far less concentrated than the losses of S_0 for small values of r (any $r \leq r^*$). It generally helps if the perturbing distribution ν has a bounded density with a small bound, so that its contributions to $F_{\mu_S}(r)$ remain small for small values of r. Illustrating examples, which apply to the cases of K-MEANS clustering and principal subspace analysis are given in Figures 6.3.1 and 6.4.1.

We now state the main result of this section.

Theorem 6.1. *Let $\mu^*, \nu \in \mathcal{P}(\mathbb{R}^d)$, $\mu = (1-\lambda)\mu^* + \lambda\nu$, and $\lambda \in (0,1)$ and suppose there are S_0, r^*, $\delta > 0$ and $\beta \in (0, 1-\lambda)$, satisfying Assumption A. Let W be nonzero on a set of positive Lebesgue measure, nonincreasing and $W(t) = 0$ for $t \geq \zeta = F_{\mu_{S_0}}(r^*)$. Then $\mathrm{IF}_{\max}(\mu_{S_0}, W) > 0$, and if any $S \in \mathcal{S}$ satisfies*

$$\Phi_W(\mu_S) - \Phi_W(\mu_{S_0}) \leq \left(1 - \frac{\beta}{1-\lambda}\right) \mathrm{IF}_{\max}(\mu_{S_0}, W) \tag{6.9}$$

then we have that $\Phi(\mu_S^*) \leq \Phi(\mu_{S_0}^*) + \delta$. In particular we always have that $\Phi(\mu_{S^\dagger}^*) \leq \Phi(\mu_{S_0}^*) + \delta$.

We close this section by stating some important conclusions of the above theorem.

1. A simplifying illustration of Theorem 6.1 for principal subspace analysis is provided by Figure 6.4.1. The distributions μ^* and ν are assumed to have uniform densities $\rho(\mu^*)$ and $\rho(\nu)$ supported on dark red and light red areas of the unit disk respectively. Suppose $\beta = \rho(\nu)/\rho(\mu^*) < 1-\lambda$, let $r^* = \sin^2(\pi/\rho(\mu^*))$ and $\delta = 4r^*$. If $\Phi(\mu_S^*) > \Phi(\mu_{S^*}^*) + \delta$ then the direction of the subspace S does not intersect the black part of the unit circle and therefore $F_{\mu_S}(r) \leq \beta F_{\mu_{S^*}^*}(r)$ for all $r \leq r^*$. Thus Assumption A is satisfied and consequently, if $W(t) = 0$ for $t \geq F_{\mu_{S^*}}(r^*)$, then $S^\dagger$ must intersect the black part of the unit circle and $\Phi(\mu_{S^\dagger}^*) \leq \Phi(\mu_{S^*}^*) + \delta$. Refer also to Figure 6.4.1 for an illustration.

2. The generic application of this result assumes that $S_0 = S^*(\mu^*)$, but this is not required. Suppose $\mathcal{S}$ is the set of singletons and μ^* is bimodal, say the mixture of distant standard normal distributions, and $\lambda = 0$ for simplicity. Clearly there is no local concentration

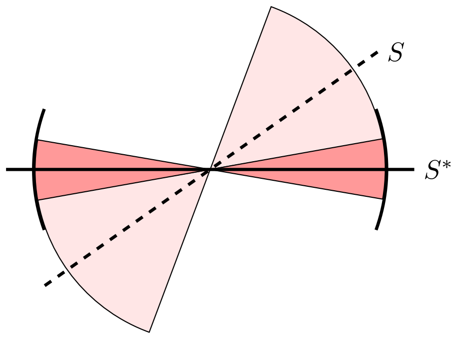


Figure 6.4.1: Illustration of Theorem 6.1 for $d = 2$ and $k = 1$ in the case of PSA. The target distribution (dark gray) is concentrated on the subspace S^*, while the perturbing distribution (light gray) does not concentrate well on any individual subspace.

on the midpoint $S^* (\mu^*)$, but there is on each of the modes. If S_0 is the mean of the first mode and ζ is sufficiently small, then $S^\dagger$ can be near the mean of the other mode, because it has comparable reconstruction error. In this way the result also explains the astonishing behavior of our algorithm in clustering experiments with mis-specified number of clusters. Refer also to Figure 6.7.1 (top right) for an illustration.

3. The conditions on W prescribe an upper bound on the cutoff parameter ζ. If the cutoff parameter ζ is chosen smaller (so that $W(t) = 0$ for $t \geq \zeta \ll F_{\mu_{S^*}}(r^*)$), the required upper bound in (6.9) decreases and it becomes more difficult to find S satisfying the upper bound. This problem becomes even worse in practice, because the bounds on the estimation error also increase with ζ, as we will see in the next section.

6.5 Generalization Analysis

Up to this point we were working with distributions and essentially infinite data. In practice we only have samples $\mathbf{X} \sim \mu^n$ and then it is important to understand to which extend we can obtain the conclusion of Theorem 6.1, when S is the minimizer of the empirical robust functional $\Phi_W(\hat{\mu}(\mathbf{X})_S)$. This can be settled by a uniform bound on the estimation error for Φ_W.

Proposition 6.2. *Under the conditions of Theorem 6.1 with $\mathbf{X} \sim \mu^n$ we have that*

$$\mathbb{P}\left\{\Phi\left(\mu^*_{\hat{S}(\mathbf{X})}\right) \leq \Phi(\mu^*_{S^*}) + \delta\right\}$$
$$\geq \mathbb{P}\left\{2 \sup_{S \in \mathcal{S}} |\Phi_W(\mu_S) - \Phi_W(\hat{\mu}_S(\mathbf{X}))| \right.$$
$$\left. \leq \left(1 - \frac{\beta}{1-\lambda}\right) \mathrm{IF}_{\max}(\mu_{S^*}, W)\right\}.$$

The left hand side is the probability that the minimization of our robust L-statistic objective returns a δ-optimal model for the target distribution μ^*. The right hand side goes to 1 as n grows. As we show next, this is due to the fact that the class $\{\mu_S\}$ enjoys a uniform convergence property with respect to the functional Φ_W. Particularly, we present three uniform bounds that control the rate of decay of the same estimation error $|\Phi_W(\mu_S) - \Phi_W(\hat{\mu}_S(\mathbf{X}))|$.

The first two bounds are dimension-free and rely on Rademacher and Gaussian averages of the function class $\{x \mapsto d(x, S) : S \in \mathcal{S}\}$. Bounds for these complexity measures in the practical cases considered can be found in [Maurer and Pontil, 2010; Lee and Raginsky, 2019; Vainsencher et al., 2011]. Our last bound is dimension dependent but may outperform the other two if the variance of the robust objective is small under its minimizer. All three bounds require special properties of the weight function W.

For this section we assume $\mu \in \mathcal{P}\left(\mathbb{R}^d\right)$ to have compact support, write $\mathcal{X}$ =support(μ) and let $\mathcal{F}$ be the function class

$$\mathcal{F} = \left\{x \in \mathcal{X} \mapsto d\left(x, S\right) : S \in \mathcal{S}\right\}.$$

We also set $R_{\max} = \sup_{f \in \mathcal{F}} \|f\|_\infty$.

The first bound is tailored to the hard-threshold $\zeta^{-1} 1_{[0,\zeta]}$. It follows directly from the elegant recent results of [Lee et al., 2020]. For the benefit of the reader we give a proof in the appendix, without any claim of originality and only slightly improved constants.

Theorem 6.3. *Let $W = \zeta^{-1} 1_{[0,\zeta]}$ and $\eta > 0$. With probability at least $1-\eta$ in $\mathbf{X} \sim \mu^n$ we have that*

$$\sup_{S \in \mathcal{S}} \left|\Phi_W\left(\mu_S\right) - \Phi_W\left(\hat{\mu}_S\left(\mathbf{X}\right)\right)\right| \leq \frac{2}{\zeta n} \mathbb{E}_{\mathbf{X}} \mathcal{R}\left(\mathcal{F}, \mathbf{X}\right) + \frac{R_{\max}}{\zeta \sqrt{n}} \left(2 + \sqrt{\frac{\ln\left(2/\eta\right)}{2}}\right),$$

where $\mathcal{R}\left(\mathcal{F}, \mathbf{X}\right)$ is the Rademacher average

$$\mathcal{R}\left(\mathcal{F}, \mathbf{X}\right) = \mathbb{E}_\epsilon \left[\sup_{S \in \mathcal{S}} \sum_{i=1}^n \epsilon_i d\left(X_i, S\right)\right]$$

with independent Rademacher variables $\epsilon = \left(\epsilon_1, ..., \epsilon_n\right)$.

The next bound requires boundedness and a Lipschitz property for the weight function W which can otherwise be arbitrary. We define the norm $\|W\|_\infty = \sup_{t \in [0,1]} |W(t)|$ and seminorm $\|W\|_{\text{Lip}} = \inf\left\{L : \forall t, s \in [0,1],\ W(t) - W(s) \leq L|t-s|\right\}.$

Theorem 6.4. *For any $\eta > 0$*

$$\sup_{S \in \mathcal{S}} \left|\Phi_W\left(\mu_S\right) - \Phi_W\left(\hat{\mu}_S\left(\mathbf{X}\right)\right)\right| \leq \frac{2\sqrt{\pi}\left(R_{\max}\|W\|_\infty + \|W\|_{\text{Lip}}\right)}{n} \mathbb{E}_{\mathbf{X}} \mathcal{G}\left(\mathcal{F}, \mathbf{X}\right) + R_{\max}\|W\|_\infty \sqrt{\frac{2\ln\left(2/\eta\right)}{n}}$$

where $\mathcal{G}\left(\mathcal{F}, \mathbf{X}\right)$ is the Gaussian average

$$\mathcal{G}\left(\mathcal{F}, \mathbf{X}\right) = \mathbb{E}_\gamma \left[\sup_{S \in \mathcal{S}} \sum_{i=1}^n \gamma_i d\left(X_i, S\right)\right],$$

with independent standard normal variables $\gamma_1, ..., \gamma_n$.

Our last result also requires a Lipschitz property for W and uses a classical counting argument with covering numbers for a variance-dependent bound.

Theorem 6.5. *Under the conditions of the previous theorem, with probability at least $1-\eta$ in $\mathbf{X} \sim \mu^n$ we have that for all $S \in \mathcal{S}$*

$$\left|\Phi_W\left(\mu_S\right) - \Phi_W\left(\hat{\mu}_S\left(\mathbf{X}\right)\right)\right| \leq \sqrt{2 V_S C} + \frac{6 R_{\max}\left(\|W\|_\infty + \|W\|_{\text{Lip}}\right) C}{n} + \frac{\|W\|_\infty R_{\max}}{\sqrt{n}},$$

where V_S is the variance of the random variable $\Phi_W(\hat{\mu}_S(\mathbf{X}))$, and C is the complexity term

$$C = kd \ln\left(16n \|\mathcal{S}\|^2 / \eta\right)$$

if $\mathcal{S}$ is the set of sets with k elements, or convex polytopes with k vertices and $\|\mathcal{S}\| = \sup_{x \in S \in \mathcal{S}} \|x\|$, or

$$C = kd \ln\left(16nR^2_{\max}/\eta\right)$$

if $\mathcal{S}$ is the set of set of k-dimensional subspaces.

We state two important conclusion from the above theorems.

1. Our bounds decrease at least as quickly as $n^{-1/2}\ln n$. However, the bound in the last theorem may be considerably smaller than the previous two if n is large and the unperturbed distribution is very concentrated. The last term, which is of order $n^{-1/2}$ does not carry the burden of the complexity measure and decays quickly. The second term contains the complexity, but it decreases as n^{-1}. It can be shown from the Efron-Stein inequality [see e.g. Boucheron et al., 2013, Theorem 3.1] that the variance V_S of our L-statistic estimator is at most of order n^{-1}, so the entire bound is at most of order $n^{-1/2}\ln n$. On the other hand V_s can be very small. For example, if the unperturbed distribution is completely concentrated at S^* and ζ is chosen appropriately $V_{S^*} = 0$ and, apart from the complexity-free last term the decay is as $n^{-1}\ln n$.

2. The above bounds implies our method recovers a δ-optimal (w.r.t. μ^*) model with probability at least equal to $1 - \exp(-n)$.

Finally, we highlight that the above uniform bounds may be of independent interest. For example, consider the case that the test data also come from the perturbed distribution. In such a situation one might be interested in evaluating the performance of the learned model only on data that fit the model class, i.e. $\Phi_W(\mu_S)$. These bounds guarantee that by minimizing the empirical robust functional, one also get good performances on future data from the same distribution.

6.6 Algorithms

In this section we present our algorithm for (approximately) minimizing the robust L-statistic $\Phi_W(\hat{\mu}(\mathbf{X})_S)$ w.r.t. model $S \in \mathcal{S}$. Throughout we assume W non-increasing and fixed, and to simplify the notation we use the shorthand $\hat{\Phi}_S(\mathbf{X}) \equiv \Phi_W(\hat{\mu}(\mathbf{X})_S)$.

6.6.1 General Algorithm

Let $\mathbf{x} = (x_1, \ldots, x_n)$ be a realization of $\mathbf{X} \sim \mu^n$, consider the following function of $S \in \mathcal{S}$

$$\hat{\Phi}_S(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n} W\left(\frac{\pi(i)}{n}\right) d_S(x_i) \tag{6.10}$$

where π is the ascending ordering of the $d_S(x)_{(i)}$ and notice that minimizing (6.10) is equivalent to minimize (6.7). Let p any fixed element in Sym_n[4] and let

$$\phi_S(\mathbf{x}, p) = \frac{1}{n}\sum_{i=1}^{n} W\left(\frac{p(i)}{n}\right) d_S(x_i).$$

In the following we will leverage the following property of ϕ_S.

Lemma 6.6. *For any $S \in \mathcal{S}$ and any $p \in \mathrm{Sym}_n$, if π is the ascending ordering of the $d_S(x_i)$s, then $\phi_S(\mathbf{x}, p) \geq \phi_S(\mathbf{x}, \pi) = \hat{\Phi}_S(\mathbf{x})$.*

We need also the following definition.

[4]Here Sym_n denotes the set of all $n!$ permutations over n objects.

Definition 6.7. *A mapping $\mathcal{D} : \mathcal{S} \times S_n \to \mathcal{S}$ is a Descent Oracle for ϕ_S iff for any $S \in \mathcal{S}$ and any $p \in \mathrm{Sym}_n$, $\phi_{\mathcal{D}(S,p)}(\mathbf{x}, p) \leq \phi_S(\mathbf{x}, p)$.*

The algorithm attempts to minimize (6.10) via alternating minimization of ϕ_S. At the beginning, it picks an initial model S_0 and sort the induced losses in ascending order, i.e. pick the optimal permutation π_0. Then it starts iterating this two steps by first calling the descent oracle $\mathcal{D}(S_t, \pi_t)$ and then sorting the induced losses. At each step either the permutation π_t or the model S_t are fixed. Pseudo-code is given in Algorithm 3.

Algorithm 3 RobustUnsupervisedLearning($\mathbf{x}$)

```
1: Pick any $S_0 \in \mathcal{S}$;
2: $\pi_0 \leftarrow \mathrm{argmin}_{p \in \mathrm{Sym}_n} \phi_{S_0}(\mathbf{x}, p)$;
3: for $t = 1, \dots, T$ do
4:     $S_t \leftarrow \mathcal{D}(S_{t-1}, \pi_{t-1})$;
5:     $\pi_t \leftarrow \mathrm{argmin}_{p \in S_n} \phi_{S_t}(\mathbf{x}, p)$;
6: return $S_T$;
```

Indeed, at each step the algorithm first finds a descending iteration S_{t+1} of $\phi_{S_t}(\mathbf{x}, \pi_t)$ and then sort the losses according to π_{t+1}, an operation that by Lemma 6.6 cannot increase the value of $\phi_{S_{t+1}}$. Thus the following holds.

Theorem 6.8. *Algorithm 3 is a descent algorithm for the problem of minimizing (6.10), i.e. for any $t, \hat{\Phi}_{S_{t+1}}(\mathbf{x}) \leq \hat{\Phi}_{S_t}(\mathbf{x})$.*

This algorithm is general and to apply it to a specific learning problem an implementation of the descent oracle is needed. The efficiency of Algorithm 3 depends upon such oracle. In the following we show two descent oracles for the cases of KMEANS and PSA. On the negative side notice that in the case of KMEANS when W is the identity, the problem reduces to finding the optimal KMEANS solution, a problem which is known to be hard (further hardness evidence are provided in the supplement). Thus, in the general case, it is not possible to solve our empirical problem optimally. Our algorithms, are a first step towards the design of methods with provable approximation guarantees.

k-Means Clustering (KMEANS). In this case $\mathcal{S}$ is the set of all possible k-tuples of centers in $\mathbb{R}^d$ and $d_S(x) = \min_{c \in S} \|x - c\|_2^2$. Keeping fixed the permutation p, we consider as descent oracle the following Lloyd-like update for the centers. Each center $c \in S$ induces a cluster formed by a subset of training points x_i, $i \in \mathcal{I}$ which are closer to c than every other center (breaking ties arbitrarily). The overall *loss* of representing point in $\mathcal{I}$ with c is

$$\sum_{i \in \mathcal{I}} W\left(\frac{p(i)}{n}\right) \|x_i - c\|_2^2.$$

This loss is minimized at

$$\hat{c} = \frac{1}{\sum_{i \in \mathcal{I}} W\left(\frac{p(i)}{n}\right)} \sum_{i \in \mathcal{I}} W\left(\frac{p(i)}{n}\right) x_i,$$

so the following holds.

Proposition 6.9. *Given S and p, the mapping that for every $c \in S$ returns the $\hat{c}$ defined above is a descent oracle for* KMEANS *and its runtime is $O(nkd)$.*

The resulting algorithm can is a generalization of the method proposed in [Chawla and Gionis, 2013].

Principal Subspace Analysis (PSA). In this case $\mathcal{S}$ is the set of all possible $d \times k$ matrices U such that $U^\top U = I_d$, $d_S(x) = \|x - UU^\top x\|_2^2$ and

$$\phi_U(\mathbf{x}, p) = \sum_{i=1}^{n} W\left(\frac{p(i)}{n}\right) \|x - UU^\top x_i\|_2^2.$$

Given p, it is easy to see that the above function is minimized at the matrix $\hat{U}$ formed by stacking as columns the k eigenvectors of $\sum_i^n W\left(\frac{p(i)}{n}\right) x_i x_i^\top$ associated to the top k eigenvalues, so the following holds.

Proposition 6.10. *Given U and p, the mapping that returns the $\hat{U}$ defined above is a descent oracle for* PSA *and its runtime is* $O(\min\{d^3 + nd^2, n^3 + n^2 d\})$.

6.7 Experiments

The purpose of the numerical experiments is to show that:

- Our algorithms for PSA and KMEANS outperform standard SVD, KMEANS++ and the *Spherical Depth* method (SD) in presence of outliers, while obtain similar performances on clean data.
- Our algorithms on real data are not too sensitive to the parameters of the weight function. In particular, we show that there exist a wide-range of ζ values such that using the hard-threshold function leads to good results.
- In the case of KMEANS our method is able to accurately reconstruct some of the true centers even when the value of k is miss-specified. This matches the second remark after Theorem 1.

Implemented Algorithms. For KMEANS++ we used the *sklearn* implementation fed with the same parameters for the maximum number of iterations T and the initializations r we used for our method. Notice that T is only an upper bound to the number of iterations, the algorithms stop when the difference between the current objective value and the previous one is smaller than 10^{-7}. To set r we used the largest value before diminishing returns were observed. For standard PSA we compute the SVD of $\sum_i x_i x_i^\top$. The SD method is a general purpose pre-processing technique that is applied on the data before performing KMEANS and PSA [see e.g. Elmore et al., 2006; Fraiman et al., 2019]. This method computes a score for each point in the dataset by counting in how many balls, whose antipodes are pairs of points in the data, it is contained. The $1 - \zeta n$ points with the smallest scores are discarded. If the data contain n points, the methods needs to check $O(n^2)$ balls for each of the n point resulting in a runtime of $O(n^3)$. For scalability on real data, we implemented a randomized version of this method that for each point only check M balls picked uniformly at random from the set of all possible balls and used $M = O(n)$; the resulting runtime is $O(n^2)$. In the following we refers to our methods as RKM and RPSA respectively. All experiments have been run on an standard laptop equipped with an Intel i9 with 8 cores each working at 2,4 GHz and 16 GB of RAM DDR4 working at 2,6 GHz.

6.7.1 KMEANS Clustering

Synthetic Data. We run two experiments with artificial data in $\mathbb{R}^2$. In the first experiment, we generated 300 inliers from 3 isotropic truncated Gaussians (100 points each) with variance 0.1 along both axis and mean $(-3, 0)$, $(0, 1)$ and $(3, 0)$ respectively. We then corrupt the data adding 100 points from a fourth isotropic truncated Gaussian centered at $(-1, -5)$ with variance 5 along both axis. For both RKM and KMEANS++ we $T = 10$ and $r = 30$. We initialized RKM with uniform centers and set $\zeta = 0.75$, the same ζ is used for SD. Results are shown in Figure 6.7.1 top left, where it is possible to see that while RKM recovers the true centers, SD and KMEANS++ both fail badly placing one centers in the middle of the two clusters and the other close to the mean of the perturbing distribution. In the second experiment, we generated 300 points from the same 3 inliers Gaussians and set the algorithms with $k = 2$ and $\zeta = 0.6$, while T and r are as above. Results are shown in the top right of Figure 6.7.1, where it is possible to see that KMEANS++ and SD – although to a lesser extend – wasted a center to merge 2 clusters, while RKM correctly recovers 2 out of the 3 centers.

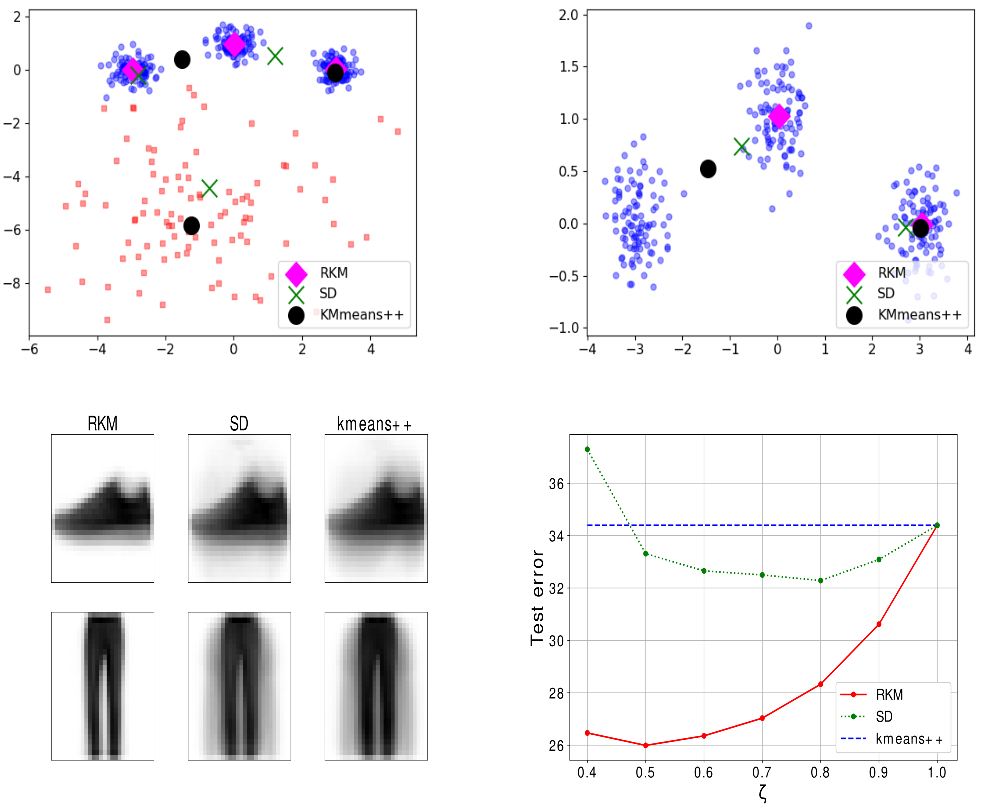


FIGURE 6.7.1: Experiments for KMEANS on synthetic data (top row) and real data (bottom row).

Real Data. In the synthetic experiments we choose ζ according to the exact fraction of outliers, a quantity which is usually unknown in practice. Here we show that there is a wide range of values for ζ such that RKM performs better than KMEANS++. We used the Fashion-MNIST dataset which consists of about 70000 28×28 images of various types of clothes splitted in a training set of 60000 images and a test set of 10000 images. Specifically, there are 10 classes in the dataset: t-shirts, trousers, pullover, dresses, coats, sandals, shirts, sneakers, bags and ankle boots. The training data were generated by sampling 1000 points, from the training set, each from the sneakers and the trousers classes as inliears, and 250 points from each other class as outliers. The resulting fraction of outliers is about 0.5. The test data consist of all the sneakers and the trousers in the test set and has size of about 2000. We run the algorithms with $T = 50, r = 30, M = 4000, k = 2$ and ζ in the range $[0.4, 1]$. Results are shown in the bottom row of Figure 6.7.1. In the lower left, it is possible to see that the centers learned by RKM at the optimal threshold value $\zeta = 0.5$ look good, while the centers found by SD and KMEANS++ are affected by the outliers. Specifically, the such centers arise from the overlap of multiple classes. One center suffers from the effect of the other two shoes classes (sandald and boots) as witnessed by the elongated background area, while the other is affected by the clothes classes (most noticeably, the coats) as suggested by background shadow. As for the reconstruction error, RKM outperforms SD uniformly over the range of considered values of ζ.

6.7.2 Principal Subspace Analysis

Synthetic Data. We run a synthetic experiment with artificial data in $\mathbb{R}^2$. We generate 50 points from the uniform distribution over $[-1, 1] \times [-0.1, 0.1]$ as inliers and 50 points for the uniform distribution over $\mathbb{R}_{++} \cup \mathbb{R}_{--} \cap B(0, 1)$[5] as outliers. We run RPSA with $T = 50$, $r = 30$, $\zeta = 0.5$ and initialize U as a normalized Gaussian matrix. We set $k = 1$ for all

[5]Here with $\mathbb{R}_{++}$ and $\mathbb{R}_{--}$ we denote the top right and the bottom left orthant of $\mathbb{R}^2$.

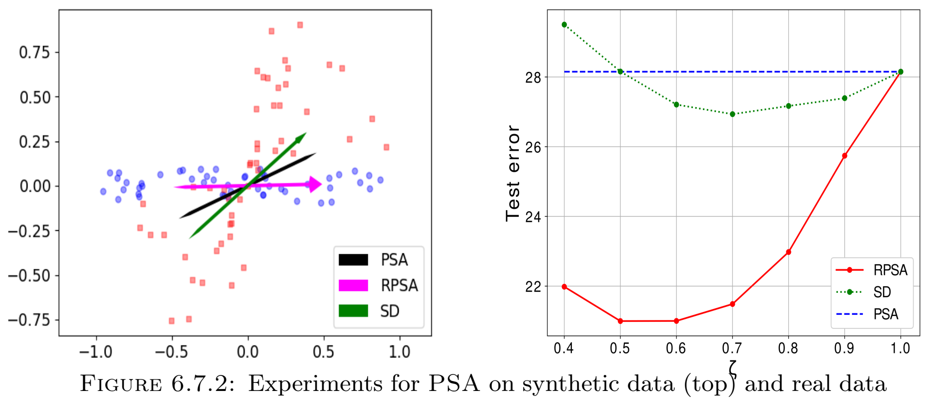


FIGURE 6.7.2: Experiments for PSA on synthetic data (top) and real data (bottom).

algorithms. Results are shown in the top plot of Figure 6.7.2 where it is possible to see that the principal subspace learned by RPSA is not affected by the outliers, as opposed to SD and PSA.

Real Data. Similarly to the case of KMEANS, we tested our method on real data for a range of values of ζ. We used again the same setting as before on the Fashion-MNIST dataset. We run the algorithms we $T = 50$, $r = 5$, $M = 4000$, $k = 2$ and ζ in the range $[0.4, 1]$. Results are shown in the bottom plot of Figure 6.7.2, where it is possible our algorithm outperforms both PSA and does better than SD.

Appendix

6.A Statistical Properties of the Proposed Method

We first analyze some basic properties of the functional Φ_W. The following is easily seen to be an alternative definition of Φ_W.

$$K_W(t) = \int_0^t W(u)\,du$$

and

$$\Phi_W(\rho) = \int_0^\infty r dK_W(F_\rho(r)) \text{ for } \rho \in \mathcal{P}([0,\infty)).$$

From this we find

Lemma 6.11. *For $\rho_1, \rho_2 \in \mathcal{P}$ and W bounded*

$$\Phi_W(\rho_1) - \Phi_W(\rho_2) = -\int_0^\infty \left(K_W(F_{\rho_1}(r)) - K_W(F_{\rho_2}(r))\right) dr, \tag{6.11}$$

and

$$\frac{d}{dt}\Phi_W((1-t)\rho_1 + t\rho_2) = \int_0^\infty W\left(F_{(1-t)\rho_1+t\rho_2}(r)\right)\left(F_{\rho_1}(r) - F_{\rho_2}(r)\right) dr.$$

Proof. Since members of $\mathcal{P}$ have finite first moments we have for any $\rho \in \mathcal{P}$ that $r\rho(r,\infty) \to 0$ as $r \to \infty$, so

$$\lim_{r\to\infty} r\left(K_W(F_{\rho_1}(r)) - K_W(F_{\rho_2}(r))\right) \leq \|W\|_\infty \lim_{r\to\infty} r\left|\rho_2(r,\infty) - \rho_1(r,\infty)\right| = 0,$$

and the formula (6.11) follows from integration by parts. Thus for arbitrary $\rho \in \mathcal{P}$

$$\Phi_W((1-t)\rho_1 + t\rho_2) - \Phi_W(\rho) = -\int_0^\infty \left(K_W((1-t)F_{\rho_1}(r) + tF_{\rho_2}(r)) - K_W(F_\rho(r))\right) dr.$$

Taking the derivative w.r.t. t and using the chain rule and $K_W' = W$ gives the second identity. $\square$

We now analyze the influence function of the functional Φ_W.

Lemma 6.12. *Let $R \in [0,\infty)$, $\rho \in \mathcal{P}([0,\infty))$*

(i) If W is nonnegative, bounded and $W(t) = 0$ for $t \geq \zeta$ and $\zeta < 1$ then

$$IF(R;\rho,\Phi_W) \leq IF_{\max}(\rho, W) := \int_0^{F_\rho^{-1}(\zeta)} W(F_\rho(r))\,F_\rho(r)\,dr.$$

(ii) If $\zeta > 0$, $W = \zeta^{-1}1_{[0,\zeta]}$, ρ is non-atomic and $F_\rho^{-1}(\zeta)(\rho) = F_\rho^{-1}(\zeta) = F_\rho^{-1}(\zeta)$. Then

$$\begin{aligned} IC(R;\rho,\Phi_W) &= \begin{cases} \zeta^{-1}\left(R + (\zeta - 1)F_\rho^{-1}(\zeta)\right) - \Phi_W(\rho) & \text{if} \quad 0 \leq R \leq F_\rho^{-1}(\zeta) \\ F_\rho^{-1}(\zeta) - \Phi_W(\rho) & \text{if} \quad F_\rho^{-1}(\zeta) < R \end{cases} \\ &\leq F_\rho^{-1}(\zeta) - \Phi_W(\rho). \end{aligned}$$

Proof. (i) In the second conclusion of Lemma 6.11, letting $\rho_2 = \delta_R$ and taking the limit $t \to 0$ we obtain the influence function

$$IF(R;\rho,\Phi_W) = \int_0^\infty W(F_\rho(r))\left(F_\rho(r) - F_{\delta_R}(r)\right) dr.$$

Part (i) follows.

(ii) From Lemma 6.11 we get

$$\begin{aligned}\frac{d}{dt}\Phi\left((1-t)\,\rho+t\delta_R\right)(t=0) &= \zeta^{-1}\int_0^{F_\rho^{-1}(\zeta)}\left(F_\rho\left(r\right)-1_{[R,\infty)}\left(r\right)\right)dr\\ &= \zeta^{-1}\left(\int_0^{F_\rho^{-1}(\zeta)}F_\rho\left(r\right)dr-\int_0^{F_\rho^{-1}(\zeta)}1_{[R,\infty)}\left(r\right)dr\right).\end{aligned}$$

From integration by parts the first term in parenthesis is $\zeta\left(F_\rho^{-1}\left(\zeta\right)-\Phi_W\left(\rho\right)\right)$. The second term is zero if $F_\rho^{-1}\left(\zeta\right)<R$, otherwise it is $F_\rho^{-1}\left(\zeta\right)-R$. This gives the identity. For the inequality observe that $R\leq F_\rho^{-1}\left(\zeta\right)$ implies $\zeta^{-1}\left(R+\left(\zeta-1\right)F_\rho^{-1}\left(\zeta\right)\right)\leq F_\rho^{-1}\left(\zeta\right)$. □

6.A.1 Resilience to Perturbations

We prove Theorem 1.

Lemma 6.13. *Let $S,S^*\in\mathcal{S}$, $\mu\in\mathcal{P}\left(\mathbb{R}^d\right)$, and suppose that there exists $r^*>0$ and $\alpha\in(0,1)$ such that*

$$\forall r\in\left(0,r^*\right),\ F_{\mu_S}\left(r\right)\leq\alpha F_{\mu_{S^*}}\left(r\right). \tag{6.12}$$

If W is nonzero on a set of positive Lebesgue measure, nonincreasing and $W\left(t\right)=0$ for all $t\geq F_{\mu_{S^*}}\left(r^*\right)$ then

$$\Phi_W\left(\mu_S\right)-\Phi_W\left(\mu_{S^*}\right)\geq\left(1-\alpha\right)\int_0^{r^*}W\left(F_{\mu_{S^*}}\left(r\right)\right)F_{\mu_{S^*}}\left(r\right)dr=\left(1-\alpha\right)IF_{\max}\left(\mu_{S^*},W\right)>0.$$

Proof. By Lemma 6.11 and the fundamental theorem of calculus

$$\Phi_W\left(\mu_S\right)-\Phi_W\left(\mu_{S^*}\right)=\int_0^\infty\left(\int_{[0,1]}W\left(sF_{\mu_S}\left(r\right)+\left(1-s\right)F_{\mu_{S^*}}\left(r\right)\right)ds\right)\left(F_{\mu_{S^*}}\left(r\right)-F_{\mu_S}\left(r\right)\right)dr.$$

Suppose first $r^*\leq r$. If $W>0$ then $sF_{\mu_S}\left(r\right)+\left(1-s\right)F_{\mu_{S^*}}\left(r\right)<F_{\mu_{S^*}}\left(r^*\right)\leq F_{\mu_{S^*}}\left(r\right)$ and therefore $F_{\mu_S}\left(r\right)<F_{\mu_{S^*}}\left(r^*\right)$, so the integrand is positive, or else $W=0$. For a lower bound we can therefore restrict the integration in r to the interval $[0,r^*)$.

If $r<r^*$ then by (6.12) $sF_{\mu_S}\left(r\right)+\left(1-s\right)F_{\mu_{S^*}}\left(r\right)<F_{\mu_{S^*}}\left(r\right)\leq F_{\mu_{S^*}}\left(r^*\right)$ so $W\left(sF_{\mu_S}\left(r\right)+\left(1-s\right)F_{\mu_{S^*}}\left(r\right)\right)\geq W\left(F_{\mu_{S^*}}\left(r\right)\right)$, since W is nonincreasing. The conclusion follows from (6.12). □

We restate Assumption A and Theorem 1.

Assumption A. There exists $S_0\in\mathcal{S}$, $\delta>0$, $\beta\in\left(0,1-\lambda\right)$ and a scale parameter $r^*\in(0,1)$ (in units of squared euclidean distance), such that for every model $S\in\mathcal{S}$ satisfying $\Phi\left(\mu_S^*\right)>\Phi\left(\mu_{S_0}^*\right)+\delta$ we have $F_{\mu_S}\left(r\right)<\beta F_{\mu_{S_0}^*}\left(r\right)$ for all $r\leq r^*$.

Theorem 6.14. *Let $\mu^*,\nu\in\mathcal{P}\left(\mathbb{R}^d\right)$, $\mu=\left(1-\lambda\right)\mu^*+\lambda\nu$, and $\lambda\in(0,1)$ and suppose there are S_0, r^*, $\delta>0$ and $0<\beta<1-\lambda$, satisfying Assumption A. Suppose that W is nonzero on a set of positive Lebesgue measure, nonincreasing and $W\left(t\right)=0$ for $t\geq\zeta=F_{\mu_{S_0}}\left(r^*\right)$.*

Proof. Let $S,S_0\in\mathcal{S}$ and assume that $\Phi\left(\mu_S^*\right)>\Phi\left(\mu_{S_0}^*\right)+\delta$. Then for $r\leq r^*$ Assumption A implies $F_{\mu_S}\left(r\right)\leq\beta F_{\mu_{S_0}^*}\left(r\right)\leq\frac{\beta}{1-\lambda}F_{\mu_{S_0}}\left(r\right)$, and the conditions on W also imply that $W=0$ on $\left[F_{\mu_{S^*}}\left(r^*\right),1\right]$. Thus Lemma 6.13 with $a=\beta/\left(1-\lambda\right)<1$ gives

$$\Phi_W\left(\mu_S\right)-\Phi_W\left(\mu_{S_0}\right)\geq\left(1-\frac{\beta}{1-\lambda}\right)IF_{\max}\left(\mu_{S_0},W\right)>0.$$

Thus, if $\Phi_W\left(\mu_S\right)-\Phi_W\left(\mu_{S_0}\right)<\left(1-\frac{\beta}{1-\lambda}\right)IF_{\max}\left(\mu_{S_0},W\right)$, we must have $\Phi\left(\mu_S^*\right)\leq\Phi\left(\mu_{S_0}^*\right)+\delta$. The condition (6.12) is clearly always satisfied by the minimizer $S^\dagger\left(\mu\right)$ of $\Phi_W\left(\mu_S\right)$. □

An additional example. We conclude this section with an example to demonstrate that Assumption A does not depend so much on the richness of $\mathcal{S}$ (which will be relevant to generalization) but on the concentration properties of μ^* and ν. Let $\mathcal{S} = \left\{S \subseteq \mathbb{R}^d\right\}$, μ^* be supported in a ball of diameter ϵ, ν have any bounded density and $\lambda \in (0,1)$ be arbitrary. Then for any $\delta > \epsilon^2$ we can find $r^* > 0$, such that Assumption A holds with arbitrary $\beta \in (0,1)$.

Proof. Let $\delta > \epsilon^2$ and $\beta \in (0,1)$ be arbitrary and S_0 be an arbitrary set containing the support of μ^*, so that $\Phi\left(\mu^*_{S_0}\right) = 0$. Let B be the ball containing the support of μ^*. Assume $\Phi\left(\mu^*_S\right) > \Phi\left(\mu^*_{S_0}\right) + \delta = \delta$ and suppose that $x \in S$. This implies $\|x-y\| > \sqrt{\delta} - \epsilon > 0$ for all points in B. The first term in $F_{\mu_S}(r) = (1-\lambda) F_{\mu^*_S}(r) + \lambda F_{\nu_S}(r)$ is zero for all $r < \left(\sqrt{\delta}-\epsilon\right)^2$ and the second increases continuously from $r = 0$ because of the bounded density. We can therefore find $r^* < \left(\sqrt{\delta}-\epsilon\right)^2$ such that $F_{\mu_S}(r) = \lambda F_{\nu_S}(r) < \beta = \beta F_{\mu^*_{S_0}}(r)$ for all $r \leq r^*$. Thus A holds for arbitrary λ and β. □

6.A.2 Generalization

A second application of Lemma 6.11 gives a Lipschitz property of Φ_W relative to the Wasserstein and Kolmogorov metrics for distributions with bounded support.

Lemma 6.15. *For $\rho_1, \rho_2 \in \mathcal{P}$ with support in $[0, R_{\max}]$ and $\|W\|_\infty < \infty$*

$$\Phi_W(\rho_2) - \Phi_W(\rho_1) \leq \|W\|_\infty d_{\mathcal{W}}(\rho_1, \rho_2)$$

and

$$\Phi_W(\rho_2) - \Phi_W(\rho_1) \leq R_{\max} \|W\|_\infty d_K(\rho_1, \rho_2).$$

Here $d_{\mathcal{W}}(\rho_1,\rho_2) = \|F_{\rho_1} - F_{\rho_2}\|_1$ is the 1-Wasserstein distance and $d_K(\rho_1,\rho_2) = \|F_{\rho_1} - F_{\rho_2}\|_\infty$ the Kolmogorov-Smirnov distance.

Proof. From (6.11) and Hoelder's inequality we get

$$\Phi_W(\rho_1) - \Phi_W(\rho_2) = -\int_0^\infty \left(\int_{F_{\rho_2}(r)}^{F_{\rho_1}(r)} W(u)\, du\right) dr \leq 2\|W\|_\infty \int_0^\infty |F_{\rho_1}(r) - F_{\rho_2}(r)|\, dr.$$

We can bound the integral either by $\|F_{\rho_1} - F_{\rho_2}\|_1 = d_{\mathcal{W}}(\rho_1,\rho_2)$, which gives the first inequality, or by

$$\int_0^{R_{\max}} |F_{\rho_1}(r) - F_{\rho_2}(r)|\, dr \leq \|F_{\rho_1} - F_{\rho_2}\|_\infty \int_0^{R_{\max}} dr = R_{\max} d_K(\rho_1,\rho_2),$$

which gives the second inequality. □

The Lipschitz properties imply estimation and bias bounds for the plug-in estimator.

Corollary 6.16. *Let $\rho \in \mathcal{P}$ with support in $[0, R_{\max}]$ and $\|W\|_\infty < \infty$ and suppose that $\hat{\rho}$ is the empirical measure generated from n iid observations $\mathbf{R} = (R_1, ..., R_n) \sim \rho^n$*

$$\hat{\rho}(\mathbf{R}) = \frac{1}{n}\sum_{i=1}^n \delta_{R_i}.$$

Then (i)

$$\mathbb{P}\left\{|\Phi_W(\rho) - \Phi_W(\hat{\rho}(\mathbf{R}))| > t\right\} \leq 2\exp\left(\frac{-2nt^2}{R_{\max}^4 \|W\|_\infty^2}\right).$$

and (ii)

$$\Phi_W(\rho) - \mathbb{E}\left[\Phi_W(\hat{\rho}(\mathbf{R}))\right] \leq \frac{R_{\max}\|W\|_\infty}{\sqrt{2n}}.$$

Proof. (i) By Lemma 6.15 and the Dvoretzky-Kiefer-Wolfowitz Theorem in the version of Massart [Massart, 1990]

$$\mathbb{P}\left\{\left|\Phi_W(\rho)-\Phi_W(\hat\rho(\mathbf{R}))\right|>t\right\}\leq\mathbb{P}\left\{d_K(\rho,\hat\rho(\mathbf{R}))>\frac{t}{R_{\max}\|W\|_\infty}\right\}\leq 2\exp\left(\frac{-2nt^2}{R_{\max}^2\|W\|_\infty^2}\right).$$

(ii) Let $\mathbf{R}'=(R_1,...,R_n)$ be iid to $\mathbf{R}$. Then

$$\begin{aligned}\Phi_W(\rho)-\mathbb{E}\left[\Phi_W(\hat\rho(\mathbf{R}))\right] &\leq \|W\|_\infty\mathbb{E}\left[d_\mathcal{W}(\rho_1,\hat\rho(\mathbf{R}))\right]\\ &= \|W\|_\infty\mathbb{E}_\mathbf{R}\int_0^{R_{\max}}\left|\mathbb{E}_{\mathbf{R}'}\left[\frac{1}{n}\sum_i 1_{[R_i',\infty)}(t)\right]-\left[\frac{1}{n}\sum_i 1_{[R_i,\infty)}(t)\right]\right|dt\\ &\leq \frac{\|W\|_\infty}{n}\int_0^{R_{\max}}\mathbb{E}_{\mathbf{R}\mathbf{R}'}\left|\sum_i\left(1_{[R_i',\infty)}(t)-1_{[R_i,\infty)}(t)\right)\right|dt\\ &\leq \frac{\|W\|_\infty}{n}\int_0^{R_{\max}}\left(\mathbb{E}_{\mathbf{R}\mathbf{R}'}\sum_i\left(1_{[R_i',\infty)}(t)-1_{[R_i,\infty)}(t)\right)^2\right)^{1/2}dt\\ &= \frac{\|W\|_\infty}{\sqrt{n}}\int_0^{R_{\max}}\left(\mathbb{E}_{R_1R_1'}\left(1_{[R_1',\infty)}(t)-1_{[R_1,\infty)}(t)\right)^2\right)^{1/2}dt\end{aligned}$$

by Jensens inequality and independence. But the expectation is just twice the variance of the Bernoulli variable $1_{[R_1,\infty)}(t)$, and therefore at most $1/2$. The result follows. □

Rephrasing part (i) of this corollary in terms of confidence windows we have, for any $\delta>0$ with probability at least $1-\delta$ that

$$\left|\Phi_W(\rho)-\Phi_W(\hat\rho(\mathbf{R}))\right|\leq R_{\max}\|W\|_\infty\sqrt{\frac{\ln(2/\delta)}{2n}}.$$

For the weight function $W=\zeta^{-1}1_{[0,\zeta]}$ the bound on the estimation error scales with ζ^{-1}, which is not surprising, since we only consider a fraction ζ of the data. So for decreasing ζ the functional becomes more robust (because the influence R_ζ decreases) but it becomes more difficult to estimate.

Restatement of Proposition 2.

Proposition 6.17. *Assume the conditions of Theorem 1. Then*

$$\mathbb{P}\left\{\Phi\left(\mu^*_{\hat S(\mathbf{X})}\right)\leq\Phi\left(\mu^*_{S^*}\right)+\delta\right\}\geq\mathbb{P}\left\{2\sup_{S\in\mathcal{S}}\left|\Phi_W(\mu_S)-\Phi_W(\hat\mu_S(\mathbf{X}))\right|\leq\left(1-\frac{\beta}{1-\lambda}\right)IC_{\max}(\mu_{S^*},W)\right\}.$$

Proof.

$$\begin{aligned}\Phi_W\left(\mu_{\hat S(\mathbf{X})}\right)-\Phi_W(\mu_{S^*}) \leq{}& \left(\Phi_W\left(\mu_{\hat S(\mathbf{X})}\right)-\Phi_W\left(\hat\mu_{\hat S(\mathbf{X})}(\mathbf{X})\right)\right)+\left(\Phi_W\left(\hat\mu_{\hat S(\mathbf{X})}(\mathbf{X})\right)-\Phi_W\left(\hat\mu_{S^\dagger}(\mathbf{X})\right)\right)\\ &+\left(\Phi_W\left(\hat\mu_{S^\dagger}(\mathbf{X})\right)-\Phi_W(\mu_{S^\dagger})\right)+\left(\Phi_W(\mu_{S^\dagger})-\Phi_W(\mu_{S^*})\right).\end{aligned}$$

The second term and the last term are negative by the minimality properties of $\hat S(\mathbf{X})$ and $S^\dagger$. The remaining terms are bounded by $2\sup_{S\in\mathcal{S}}\left|\Phi_W(\mu_S)-\Phi_W(\hat\mu_S(\mathbf{X}))\right|$. Thus

$$\begin{aligned}&\mathbb{P}\left\{2\sup_{S\in\mathcal{S}}\left|\Phi_W(\mu_S)-\Phi_W(\hat\mu_S(\mathbf{X}))\right|\leq\left(1-\frac{\beta}{1-\lambda}\right)IC_{\max}(\mu_{S^*},W)\right\}\\ &\leq\mathbb{P}\left\{\Phi_W\left(\mu_{\hat S(\mathbf{X})}\right)-\Phi_W(\mu_{S^*})\leq\left(1-\frac{\beta}{1-\lambda}\right)IC_{\max}(\mu_{S^*},W)\right\}\\ &\leq\mathbb{P}\left\{\Phi\left(\mu^*_{\hat S(\mathbf{X})}\right)\leq\Phi\left(\mu^*_{S^*}\right)+\delta\right\},\end{aligned}$$

where the last inequality follows from Theorem 1. □

Lemma 6.18. *If $W=\zeta^{-1}1_{[0,\zeta]}$ with $\zeta<1$, then for $\rho\in\mathcal{P}\left([0,R_{\max})\right)$*

$$\Phi_W\left(\rho\right)=\sup_{\lambda\in[0,R_{\max}]}\left\{\lambda-\zeta^{-1}\int_0^\infty\max\left\{\lambda-t,0\right\}d\rho\left(t\right)\right\}$$

Proof. Integration by parts gives

$$\int_0^\infty\max\left\{\lambda-t,0\right\}d\rho\left(t\right)=\int_0^\lambda F_\rho\left(t\right)dt=\lambda F_\rho\left(\lambda\right)-\int_0^\lambda td\rho\left(t\right).$$

The maximum of $\lambda-\zeta^{-1}\int_0^\lambda F_\rho\left(t\right)dt$ is attained at $\zeta=F_\rho\left(\lambda\right)$, which shows $\lambda\leq R_{\max}$, and substitution gives

$$\begin{aligned}\sup_{\lambda\in\mathbb{R}}\left\{\lambda-\zeta^{-1}\int_0^\infty\max\left\{\lambda-t,0\right\}d\rho\left(t\right)\right\}&=\lambda-\zeta^{-1}\left(\lambda F_\rho\left(\lambda\right)-\int_0^\lambda td\rho\left(t\right)\right)\\&=\zeta^{-1}\int_0^{F_\rho^{-1}(\zeta)}td\rho\left(t\right)=\int_0^\infty t\zeta^{-1}1_{[0,\zeta]}\left(F_\rho\left(t\right)\right)d\rho\left(t\right)\\&=\Phi_W\left(\rho\right).\end{aligned}$$

□

Restatement of Theorem 3.

Theorem 6.19. *Let $W=\zeta^{-1}1_{[0,\zeta]}$ and $\eta>0$. With probability at least $1-\eta$ in $\mathbf{X}\sim\mu^n$ we have that*

$$\sup_{S\in\mathcal{S}}\left|\Phi_W\left(\mu_S\right)-\Phi_W\left(\hat\mu_S\left(\mathbf{X}\right)\right)\right|\leq\frac{2}{\zeta n}\mathbb{E}_{\mathbf{X}}\mathcal{R}\left(\mathcal{F},\mathbf{X}\right)+\frac{R_{\max}}{\zeta\sqrt{n}}\left(2+\sqrt{\frac{\ln\left(2/\eta\right)}{2}}\right),$$

where $\mathcal{R}\left(\mathcal{F},\mathbf{X}\right)$ is the Rademacher average

$$\mathcal{R}\left(\mathcal{F},\mathbf{X}\right)=\mathbb{E}_\epsilon\left[\sup_{S\in\mathcal{S}}\sum_{i=1}^n\epsilon_i d\left(X_i,S\right)\right]$$

with independent Rademacher variables $\epsilon=\left(\epsilon_1,...,\epsilon_n\right)$.

Proof. Using Lemma 6.18 we get with independent Rademacher variables $\epsilon=\left(\epsilon_1,...,\epsilon_n\right)$

$$\begin{aligned}&\mathbb{E}\left[\sup_{S\in\mathcal{S}}\Phi_W\left(\mu_S\right)-\Phi_W\left(\hat\mu_S\left(\mathbf{X}\right)\right)\right]\\\leq\;&\zeta^{-1}\mathbb{E}_{\mathbf{X}}\left[\sup_{\lambda\in[0,R_{\max}],S\in\mathcal{S}}\int_0^\infty\max\left\{\lambda-t,0\right\}d\hat\mu_S\left(\mathbf{X}\right)\left(t\right)-\int_0^\infty\max\left\{\lambda-t,0\right\}d\mu_S\left(t\right)\right]\\=\;&\zeta^{-1}\mathbb{E}_{\mathbf{X}}\left[\sup_{\lambda\in[0,R_{\max}],S\in\mathcal{S}}\frac{1}{n}\sum_{i=1}^n\max\left\{\lambda-d\left(X_i,S\right),0\right\}-\mathbb{E}_{X\sim\mu}\left[\max\left\{\lambda-d\left(X,S\right),0\right\}\right]\right]\\=\;&\frac{1}{\zeta n}\mathbb{E}_{\mathbf{X}\mathbf{X}'}\left[\sup_{\lambda\in[0,R_{\max}],S\in\mathcal{S}}\sum_{i=1}^n\left(\max\left\{\lambda-d\left(X_i,S\right),0\right\}-\max\left\{\lambda-d\left(X_i',S\right),0\right\}\right)\right]\\=\;&\frac{1}{\zeta n}\mathbb{E}_{\mathbf{X}\mathbf{X}'\epsilon}\left[\sup_{\lambda\in[0,R_{\max}],S\in\mathcal{S}}\sum_{i=1}^n\epsilon_i\left(\max\left\{\lambda-d\left(X_i,S\right),0\right\}-\max\left\{\lambda-d\left(X_i',S\right),0\right\}\right)\right]\\\leq\;&\frac{2}{\zeta n}\mathbb{E}_{\mathbf{X}\epsilon}\left[\sup_{\lambda\in[0,R_{\max}],S\in\mathcal{S}}\sum_{i=1}^n\epsilon_i\max\left\{\lambda-d\left(X_i,S\right),0\right\}\right]\\\leq\;&\frac{2}{\zeta n}\mathbb{E}_{\mathbf{X}\epsilon}\left[\sup_{\lambda\in[0,R_{\max}],S\in\mathcal{S}}\sum_{i=1}^n\epsilon_i\left(\lambda-d\left(X_i,S\right)\right)\right]\end{aligned}$$

$$
\begin{aligned}
&\leq \frac{2}{\zeta n}\mathbb{E}_{\mathbf{X}\epsilon}\left[\sup_{S\in\mathcal{S}}\sum_{i=1}^{n}\epsilon_i d\left(X_i,S\right)\right]+\frac{2}{\zeta n}\mathbb{E}_{\epsilon}\left[\sup_{\lambda\in[0,R_{\max}]}\lambda\sum_{i=1}^{n}\epsilon_i\right] \\
&\leq \frac{2}{\zeta n}\mathbb{E}_{\mathbf{X}}\mathcal{R}\left(\mathcal{F},\mathbf{X}\right)+\frac{2R_{\max}}{\zeta\sqrt{n}}.
\end{aligned}
$$

Here the third identity is a standard symmetrization argument, the second inequality the triangle inequality, followed by the contraction inequality for Rademacher averages, since $t\to\max\{t,0\}$ is a contraction. Then we used the triangle inequality again. Now let $\Psi(\mathbf{X})$ be the random variable $\sup_{S\in\mathcal{S}}\Phi_W(\mu_S)-\Phi_W(\hat{\mu}_S(\mathbf{X}))$. It then follows from Lemma 6.15 and the bounded difference inequality that with probability at least $1-\eta$ we have $\Psi(X)\leq\mathbb{E}\Psi(X)+\zeta^{-1}R_{\max}\sqrt{\ln(1/\eta)/(2n)}$.

Combined with above bound on $\mathbb{E}\Psi(X)$ this completes the proof. □

Theorem 4 follows directly from Theorems 2 and 5 in [Maurer and Pontil, 2019] and from the bias bound, Corollary 6.16 (ii).

Restatement of Theorem 5.

Theorem 6.20. *Under the conditions of the previous theorem, with probability at least $1-\eta$ in $\mathbf{X}\sim\mu^n$ we have that for all $S\in\mathcal{S}$*

$$
\left|\Phi_W(\mu_S)-\Phi_W(\hat{\mu}_S(\mathbf{X}))\right|\leq\sqrt{2V_SC}+\frac{6R_{\max}\left(\|W\|_\infty+\|W\|_{Lip}\right)C}{n}+\frac{\|W\|_\infty R_{\max}}{\sqrt{n}},
$$

where V_S is the variance of the random variable $\Phi_W(\hat{\mu}_S(\mathbf{X}))$, and C is the complexity term

$$
C=kd\ln\left(16n\|\mathcal{S}\|^2/\eta\right)
$$

if $\mathcal{S}$ is the set of sets with k elements, or convex polytopes with k vertices and $\|\mathcal{S}\|=\sup_{x\in S\in\mathcal{S}}\|x\|$, or

$$
C=kd\ln\left(16nR_{\max}^2/\eta\right)
$$

if $\mathcal{S}$ is the set of set of k-dimensional subspaces.

Proof. For any fixed $S\in\mathcal{S}$ the L-statistic $\mathbf{x}\in\mathcal{X}^n\mapsto f_S(\mathbf{x}):=\Phi_W(\hat{\mu}_S(\mathbf{x}))$ is $\left(R_{\max}\|W\|_\infty,R_{\max}\|W\|_{Lip}\right)$-weakly interacting (see [Maurer and Pontil, 2018]) and therefore satisfies the following version of Bernstein's inequality (see [Maurer et al., 2019], [Maurer and Pontil, 2018]): For $\eta\in(0,1/e)$ with probability at least $1-\eta$ in $\mathbf{X}\sim\mu^n$ we have

$$
\mathbb{E}[f_S]-f_S(\mathbf{X})\leq\sqrt{2V_S\ln(1/\eta)}+R_{\max}\left(\frac{2\|W\|_\infty}{3}+\frac{3\|W\|_{Lip}}{2}\right)\frac{\ln(1/\eta)}{n},
$$

where $\mathbb{E}[f_S]$ and V_S are expectation and variance of the random variable $f_S(\mathbf{X})=\Phi_W(\hat{\mu}_S(\mathbf{X}))$ respectively. We will make this bound uniform with a covering number argument.

Define a pseudo metric $d_{\mathcal{X}}$ on $\mathcal{S}$ by

$$
d_{\mathcal{X}}(S_1,S_2)=\sup_{x\in\mathcal{X}}\left|d(x,S_1)-d(x,S_2)\right|.
$$

It follows from Lemma 6.15 that for every $\mathbf{x}\in\mathcal{X}^n$ we have

$$
f_{S_1}(\mathbf{x})-f_{S_2}(\mathbf{x})\leq\|W\|_\infty d_{\mathcal{W}}\left(\hat{\mu}_{S_1}(\mathbf{x}),\hat{\mu}_{S_2}(\mathbf{x})\right)\leq\|W\|_\infty d_{\mathcal{X}}(S_1,S_2).
$$

In particular $\left|\mathbb{E}[f_{S_1}]-\mathbb{E}[f_{S_2}]\right|\leq\|W\|_\infty d_{\mathcal{X}}(S_1,S_2)$ and

$$
\begin{aligned}
\sqrt{V_{S_1}}-\sqrt{V_{S_2}} &= \left\|f_{S_1}-\mathbb{E}[f_{S_1}]\right\|_{L_2(\mu^n)}-\left\|f_{S_2}-\mathbb{E}[f_{S_2}]\right\|_{L_2(\mu^n)} \\
&\leq \left\|f_{S_1}-f_{S_2}\right\|_{L_2(\mu^n)}+\left|\mathbb{E}[f_{S_1}]-\mathbb{E}[f_{S_2}]\right|\leq2\|W\|_\infty d_{\mathcal{X}}(S_1,S_2).
\end{aligned}
$$

Now let $N=N(\mathcal{S},d_{\mathcal{X}},\epsilon)$ be the corresponding minimal covering number of $\mathcal{S}$ with $d_{\mathcal{X}}$-balls of radius ϵ, and let $\mathcal{S}_0\subseteq\mathcal{S}$ be such that $\forall S\in\mathcal{S}$, $\exists S'\in\mathcal{S}_0$ with $d_R(S,S')<1/n$ and $|\mathcal{S}_0|\leq N$.

Then, abbreviating $R_{\max}\left(2\left\|W\right\|_\infty/3+3\left\|W\right\|_{Lip}/2\right)$ with C, with probability at least $1-\eta$ in $\mathbf{X}$ that for every $S\in\mathcal{S}$

$$\begin{aligned}\mathbb{E}\left[f_S\right]-f_S\left(\mathbf{X}\right) &\leq \mathbb{E}\left[f_{S'}\right]-f_{S'}\left(\mathbf{X}\right)+\frac{2\left\|W\right\|_\infty}{n}\leq\sqrt{2V_{S'}\ln\left(N/\eta\right)}+\frac{C\ln\left(N/\eta\right)+2\left\|W\right\|_\infty}{n}\\ &= \sqrt{2V_S\ln\left(N/\eta\right)}+\frac{C\ln\left(N/\eta\right)+2\left\|W\right\|_\infty}{n}+\left(\sqrt{V_{S'}}-\sqrt{V_S}\right)\sqrt{2\ln\left(N/\eta\right)}\\ &\leq \sqrt{2V_S\ln\left(N/\eta\right)}+\frac{C\ln\left(N/\eta\right)+2\left\|W\right\|_\infty\sqrt{2\ln\left(N/\eta\right)}+2\left\|W\right\|_\infty}{n}.\end{aligned}$$

In the first inequality we used uniform approximation of f_S by $f_{S'}$, where S' is the nearest neighbour of S in $\mathcal{S}_0$. The next line combines Bernstein's inequality with a union bound over $\mathcal{S}_0$. Finally we again approximate $\sqrt{V_{S'}}$ by $\sqrt{V_S}$.

Next we bound the covering numbers $N\left(\mathcal{S},d_\mathcal{X},1/n\right)$, which we do separately for the case of uniformly bounded $\mathcal{S}$ and PSA. In case of the mean, k-means or sparse coding is easy to see that for $S_1,S_2\in\mathcal{S}$ and any two respective enumerations x_i and y_i or enumerations of the extreme points

$$d_\mathcal{X}\left(S_1,S_2\right)\leq 2\left\|\mathcal{S}\right\|H\left(S_1,S_2\right)\leq 2\left\|\mathcal{S}\right\|\max_i\left\|x_i-y_i\right\|.$$

It follows that $N\left(\mathcal{S},d_\mathcal{X},1/n\right)$ can be bounded by the covering number of a ball of radius $\left\|\mathcal{S}\right\|^2$ in a kd-dimensional Banach space. Use the standard result of Cucker and Smale [Cucker and Smale, 2002] we have

$$N\left(\mathcal{S},d_\mathcal{X},1/n\right)\leq\left(8n\left\|\mathcal{S}\right\|^2\right)^{kd}.$$

For PSA we can use unit vectors spanning the subspaces and instead of $\left\|\mathcal{S}\right\|^2$ we have the maximal squared norm in the support, so

$$N\left(\mathcal{S},d_\mathcal{X},1/n\right)\leq\left(8n\left\|\mathcal{X}\right\|^2\right)^{kd}.$$

$$kd\ln\left(8n\left\|\mathcal{S}\right\|^2/\eta\right).$$

Putting it all together and adding the bias bound $\Phi_W\left(\mu_S\right)-\mathbb{E}\left[\Phi_W\left(\hat{\mu}_S\left(\mathbf{X}\right)\right)\right]\leq\left\|W\right\|_\infty R_{\max}/\sqrt{n}$ (Corollary 6.16 (ii)) we get

$$\begin{aligned}&\Phi_W\left(\mu_S\right)-\Phi_W\left(\hat{\mu}_S\left(\mathbf{X}\right)\right)\\ &\leq\sqrt{2\sigma^2\left(\Phi_W\left(\hat{\mu}_S\left(\mathbf{X}\right)\right)\right)kd\ln\left(8n\left\|\mathcal{S}\right\|^2/\eta\right)}+\frac{R_{\max}\left(6\left\|W\right\|_\infty+\frac{3\left\|W\right\|_{Lip}}{2}\right)kd\ln\left(8n\left\|\mathcal{S}\right\|^2/\eta\right)}{n}+\frac{\left\|W\right\|_\infty R_{\max}}{\sqrt{n}}\end{aligned}$$

The result follows from elementary estimates and algebraic simplifications. □

6.B Algorithms

Restatement of Lemma 6.

Lemma 6.21. *For any $S\in\mathcal{S}$ and any $p\in\mathrm{Sym}_n$, if π is the ascending ordering of the $d_S(x_i)$s, then $\phi_S(\mathbf{x},p)\geq\phi_S(\mathbf{x},\pi)=\hat{\Phi}_S(\mathbf{x})$.*

Proof. Writing $w\left(i\right)=W\left(\frac{\pi(i)}{n}\right)$ and $z_i=d_S\left(x_{\pi(i)}\right)$ it is enough to show that the identity permutation is a minimizer of

$$r\left(p\right)=\sum_{i=1}^n w\left(p\left(i\right)\right)z_i \text{ for } p\in\mathrm{Sym}_n$$

This follows from the following claim, which we prove by induction:

For $k \in \{1, ..., n\}$ there is for every $p \in \mathrm{Sym}_n$ some $p' \in \mathrm{Sym}_n$ such that $r(p') \leq r(p)$ and $p'(j) = j$ for all $1 \leq j < k$. The case $k = 1$ holds trivially. If the claim holds for any $k \leq n - 1$ then there is $q \in \mathrm{Sym}_n$ such that $r(q) \leq r(p)$ and $q(j) = j$ for all $1 \leq j < k$. If $q(k) = \pi(k)$ then the claim for $k + 1$ clearly holds by defining $p' := q$. If $q(k) \neq k$ note first that both $q(k) > k$ and $q^{-1}(k) > k$. Then define $p'(j) := q(j)$ except for $p'(k) := k$ and $p'\left(q^{-1}(k)\right) := q(k)$. Then $p'(j) = j$ for all $1 \leq j < k + 1$ and

$$r(p') - r(p) \leq r(p') - r(q) = (w(k) - w(q(k)))\left(z_k - z_{q^{-1}(k)}\right) \leq 0,$$

because the first term is non-negative (since w is non-increasing) and the second non-positive. So $r(p') \leq r(p)$ which proves the claim for the case $k + 1$ and completes the induction. □

Theorem 6.22. *Minimizing $\hat{\Phi}_S(\mathbf{x})$ for the case of* KMEANS *when $k = 1$ and W is the hard threshold is NP-Hard.*

Proof. Notice that minimizing the $\hat{\Phi}_S(\mathbf{x})$ in the case of KMEANS is equivalent to minimize the following function of a subset $C \subseteq X$ of size $\lfloor zn \rfloor$

$$L(C) = \frac{1}{n} \sum_{x \in C} \|x_i - \mu_C\|_2^2$$

where $\mu_C = mean(C)$ and return μ_C. In what follow we will consider $L(C)$ as actually $L(C)n$ in order to remove the constant factor outside the objective and simplify the notation. The following lemma enables us to rewrite $L(C)$ in terms of pairwise distances.

Lemma 6.23. *Let $C \subseteq X$, then*

$$L(C) = \frac{1}{2|C|} \sum_{x,y \in C} \|x - y\|_2^2. \tag{6.13}$$

Proof. Let X and Y two i.i.d. random variables supported on C, then

$$\begin{aligned}
\mathbb{E}[\|X - Y\|_2^2] &= \mathbb{E}[\|X\|_2^2] + \mathbb{E}[\|Y\|_2^2] - 2\mathbb{E}[\langle X, Y\rangle] \\
&= \mathbb{E}[\|X\|_2^2] + \mathbb{E}[\|X\|_2^2] - 2\mathbb{E}[\|\mathbb{E}[X]\|_2^2] \\
&= 2\mathbb{E}[\|X\|_2^2] - 2\mathbb{E}[\|\mathbb{E}[X]\|_2^2] = 2\mathbb{E}[\|X - \mathbb{E}[X]\|_2^2].
\end{aligned}$$

Now assume X and Y are independent samples from the uniform distribution on C, then

$$\begin{aligned}
\mathbb{E}[\|X - \mathbb{E}[X]\|_2^2] &= \frac{1}{|C|} \sum_{x \in C} \|x - \mu_C\|_2^2 \\
&= \mathbb{E}[\|X - Y\|_2^2]/2 = \frac{1}{2|C|^2} \sum_{x,y \in C} \|x - y\|_2^2
\end{aligned}$$

from which the thesis follows. □

We recall the definition of NP-hardness for optimization problems.

Definition 6.24. *A computational problem Π is said NP-hard (optimization) if and only if the related decision problem Π_D is NP-hard. Assume Π is defined as the problem of minimizing a function $f_X(\mu)$ defined by an input instance X if the minimum exists, then Π_D is defined as the problem of determining, given in input X and a rational number c, whether there exist an assignment to the variables μ such that $f_X(\mu) \leq q$.*

In order to show hardness of an optimization problem Π, it is enough to show hardness of the related decision problem Π_D. For this reason, the following will be useful.

Definition 6.25. DECISION ROBUST 1-MEANS

Dataset	k	RKM	SD	K-MEANS++
FMNIST	2	**25.98**	33.17	34.39
EMNIST	2	38.41	**37.78**	40.29
cifar10	4	$\mathbf{31.07 \times 10^4}$	96.42×10^5	95.57×10^5
Victorian	5	**1.64**	1.66	1.76
Iris	1	**0.32**	3.71	4.75

TABLE 6.B.1: Experimental results for the case of K-MEANS clustering. In all the experiments $\zeta = 0.5$. In each row, the performance in bold corresponds to the winning algorithm.

Input: Points $X = \{x_1, \dots, x_n\} \subset \mathbb{R}^d$*, an integer* h *and a rational number* c*.*

Output: Yes *if there exist a* $C \subseteq X$ *such that* $|C| = h$ *and* $L(C) \leq c$*,* No *otherwise.*

To prove the theorem we will reduce $n/2$-CLIQUE to the decision version ROBUST 1-MEANS via a polynomial time algorithm. Since $n/2$-CLIQUE is NP-complete, hardness for ROBUST 1-MEANS will follow.

Definition 6.26. $n/2$-CLIQUE

Input: A simple undirected connected graph $G = (V, E)$ *with* $|V| = n$*.*

Output: Yes *if* G *contains a clique of size* $n/2$*,* No *otherwise.*

Given an instance of $n/2$-CLIQUE in the form of a graph $G = (V, E)$ with n vertices, we create an instance of ROBUST 1-MEANS $\Pi_D(G)$ which is equivalent to G. Let A denote the symmetric $n \times n$ adjacency matrix of G, i.e. $A_{ij} = 1$ iff $(i, j) \in E$ otherwise $A_{ij} = 0$. Consider the graph embedding given by the map $\phi : V \to \mathbb{R}^n$ such that $\phi(i) = A_{i:} + ne_i$, where $A_{i:}$ denotes the i-th row of A and e_i denotes the i-th vector of the canonical basis of $\mathbb{R}^n$. Given G we build an instance of ROBUST 1-MEANS by setting $X = \{\phi(1), \dots, \phi(n)\}$, $h = n/2$ and $c = m(2n^2 - 3n)$, where we set $m = \binom{n}{2}$ as a shortcut. Notice that it takes $O(n)$ to build such instance. The following lemma finishes the proof by showing the aforementioned equivalence.

Lemma 6.27. *G is a Yes instance iff $\Pi_D(G)$ is a Yes instance.*

Proof. Assume that G is a *Yes* instance, i.e. G contains at least clique of size $n/2$. Notice that for any $(i, j) \in E$ it holds that

$$\|\phi(i) - \phi(j)\|_2^2 \leq (n-1)^2 + (n-1)^2 = 2n^2 - 4n + 2 \leq 2n^2 - 3n$$

while for any $(i, j) \notin E$ it holds

$$\|\phi(i) - \phi(j)\|_2^2 \geq 2n^2.$$

If $\{c_1, \dots, c_{n/2}\}$ are the vertices in the clique, the cost $L(C)$, by Lemma 6.23, of the subset $C = \{\phi(c_1), \dots, \phi(c_{n/2})\}$ is at most c, since in such clique contains exactly m edges.

Now suppose that $\Pi_D(G)$ admits a cost of at most c. Lets denote by C the subsets of X achieving such cost, then the associated vertices must form a clique otherwise at least one of the m distance will be larger than $2n^2$ leading to a cost larger of c. □

Thus if we could solve in polynomial time DECISION ROBUST 1-MEANS we could solve in polynomial time $n/2$-CLIQUE.

□

6.C Experiments

In this section we discuss the additional experimental results we obtained with our method in the case of K-MEANS clustering. We tested RKM, SD and standard K-MEANS++ with the $\zeta = 0.5$, $r = 30$, and $T = 100$. Due to its cubic runtime, SD is slow even on moderate-sized datasets. Thus, we considered the randomized version of SD with M equals to the size of the training set. For this method, we repeated each experiment 5 times and reported the average reconstruction error on the test data (standard deviations resulted to be negligible in all cases).

In the following we describe each dataset, but Fashion MNIST whose experiment has already been described in the main body.

EMNIST. This dataset consists of about 814000 28×28 images of digits, lowercase and uppercase letters from the English alphabet arranged in 62 classes. The training data were generated by sampling 1000 0s and 1000 1s as inliers and sampling 33 points from each other class as outliers. We used $k = 2$ clusters. The test data consist of all the 0s and 1s in the test set and has a size of about 2000.

cifar10. The dataset consists of about 60000 100×100 images from 10 classes: airplanes, cars, trucks, ships, dogs, cats, frogs, horses, birds and deer. The training data were generated by sampling 1000 points from each of the vehicle classes as inliers and 300 points from each of the animal classes as outliers. We used $k = 4$ clusters. The test data consist of all the vehicle images from the test set and has size of about 4000.

Victorian. This dataset consists of 4500 texts from 45 authors of English language from Victorian Era, 100 texts from each author. The data have been processed as in [Ahmadian et al., 2019] and is made of 10 features. The training data were generated by sampling 50 points from each of one of the first 5 authors in the dataset as inliears and 5 points from each other class as outliers. We used $k = 5$. The test data consist of the remaining 50 points from each of the inlier authors and has a size of about 250.

Iris. This dataset consists of 150 records of iris flowers. Each record contains 4 features: sepal length, sepal width, petal length and petal width. There classes. The training data were generated by sampling 30 points from the *iris-setosa* class as inliear and 15 points from each other class as outliers. We used $k = 1$. Since the training set is small sized, we used exact version for SD. The test data consist of all the remaining *iris-setosa* points and has a size of about 20.

Bibliography

E. Abbe and C. Sandon. Community detection in general stochastic block models: Fundamental limits and efficient algorithms for recovery. In *Proc. of IEEE FOCS*, pages 670–688, 2015.

P. Afshani, E. Chiniforooshan, R. Dorrigiv, A. Farzan, M. Mirzazadeh, N. Simjour, and H. Zarrabi-Zadeh. On the complexity of finding an unknown cut via vertex queries. In *Proc. of COCOON*, pages 459–469. Springer, 2007.

S. Ahmadian, A. Epasto, R. Kumar, and M. Mahdian. Clustering without over-representation. In A. Teredesai, V. Kumar, Y. Li, R. Rosales, E. Terzi, and G. Karypis, editors, *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2019, Anchorage, AK, USA, August 4-8, 2019*, pages 267–275. ACM, 2019. doi: 10.1145/3292500.3330987. URL https://doi.org/10.1145/3292500.3330987.

S. Ahmadian, A. Norouzi-Fard, O. Svensson, and J. Ward. Better guarantees for k-means and euclidean k-median by primal-dual algorithms. *SIAM Journal on Computing*, 49(4): FOCS17–97–FOCS17–156, 2020. doi: 10.1137/18M1171321.

N. Ailon, M. Charikar, and A. Newman. Aggregating inconsistent information: Ranking and clustering. *J. ACM*, 55(5):23:1–23:27, 2008.

N. Ailon, A. Bhattacharya, and R. Jaiswal. Approximate correlation clustering using same-cluster queries. In *Proc. of LATIN*, pages 14–27, 2018a.

N. Ailon, A. Bhattacharya, and R. Jaiswal. Approximate correlation clustering using same-cluster queries. In *Proc. of LATIN*, pages 14–27, 2018b. doi: 10.1007/978-3-319-77404-6_2. URL https://doi.org/10.1007/978-3-319-77404-6_2.

N. Ailon, A. Bhattacharya, R. Jaiswal, and A. Kumar. Approximate clustering with same-cluster queries. In *Proc. of ITCS*, volume 94, pages 40:1–40:21, 2018c. doi: 10.4230/LIPIcs.ITCS.2018.40.

D. Angluin and P. D. Laird. Learning from noisy examples. *Mach. Learn.*, 2(4):343–370, 1987. doi: 10.1007/BF00116829. URL https://doi.org/10.1007/BF00116829.

H. Ashtiani, S. Kushagra, and S. Ben-David. Clustering with same-cluster queries. In *Advances in Neural Information Processing Systems 29*, pages 3216–3224, 2016.

J. Attenberg and F. Provost. Why label when you can search? Alternatives to active learning for applying human resources to build classification models under extreme class imbalance. In *Proc. of ACM KDD*, page 423–432, 2010.

P. Awasthi, A. Blum, and O. Sheffet. Center-based clustering under perturbation stability. *Information Processing Letters*, 112(1):49–54, 2012a. doi: https://doi.org/10.1016/j.ipl.2011.10.006.

P. Awasthi, A. Blum, and O. Sheffet. Center-based clustering under perturbation stability. *Information Processing Letters*, 112(1-2):49–54, 2012b.

M. F. Balcan and S. Hanneke. Robust interactive learning. In *Proc. of COLT*, volume 23, pages 20.1–20.34, 2012.

M.-F. Balcan and P. Long. Active and passive learning of linear separators under log-concave distributions. In *Proc. of COLT*, volume 30, pages 288–316, 2013a.

M.-F. Balcan and P. Long. Active and passive learning of linear separators under log-concave distributions. In *Proc. of COLT*, pages 288–316, 2013b.

M.-F. Balcan, A. Broder, and T. Zhang. Margin based active learning. In *Proc. of COLT*, pages 35–50, 2007a.

M.-F. Balcan, A. Broder, and T. Zhang. Margin based active learning. In *Proc. of COLT*, pages 35–50, 2007b.

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

A. Basu and T. Oertel. Centerpoints: A link between optimization and convex geometry. *SIAM Journal on Optimization*, 27(2):866–889, 2017.

S. Ben-David, N. Cesa-Bianchi, D. Haussler, and P. Long. Characterizations of learnability for classes of $\{0, ..., n\}$-valued functions. *Journal of Computer and System Sciences*, 50(1): 74–86, 1995.

A. Ben-Dor, R. Shamir, and Z. Yakhini. Clustering gene expression patterns. *Journal of Computational Biology*, 6(3-4):281–297, 1999.

D. Bertsimas and S. Vempala. Solving convex programs by random walks. *J. ACM*, 51(4): 540–556, July 2004. doi: 10.1145/1008731.1008733.

A. Beygelzimer, D. J. Hsu, J. Langford, and C. Zhang. Search improves label for active learning. In *Advances in Neural Information Processing Systems*, volume 29, 2016.

Y. Bilu and N. Linial. Are stable instances easy? *Comb. Probab. Comput.*, 21(5):643–660, Sept. 2012. doi: 10.1017/S0963548312000193.

F. Bonchi, D. García-Soriano, and K. Kutzkov. Local correlation clustering. *CoRR*, abs/1312.5105, 2013.

S. Boucheron, G. Lugosi, and P. Massart. *Concentration inequalities: A nonasymptotic theory of independence*. Oxford university press, 2013.

M. Bressan, N. Cesa-Bianchi, S. Lattanzi, and A. Paudice. Exact recovery of mangled clusters with same-cluster queries. In *Advances in Neural Information Processing Systems*, volume 33, pages 9324–9334, 2020a.

M. Bressan, N. Cesa-Bianchi, S. Lattanzi, and A. Paudice. Exact recovery of mangled clusters with same-cluster queries. In *Advances in Neural Information Processing Systems 33*, 2020b.

M. Bressan, N. Cesa-Bianchi, S. Lattanzi, and A. Paudice. Exact recovery of clusters in finite metric spaces using oracle queries. In *Proc. of COLT (to appear)*, 2021a.

M. Bressan, N. Cesa-Bianchi, S. Lattanzi, and A. Paudice. On margin-based cluster recovery with oracle queries. *CoRR*, abs/2106.04913, 2021b. URL https://arxiv.org/abs/2106.04913.

N. Cesa-Bianchi, C. Gentile, F. Vitale, and G. Zappella. Active learning on trees and graphs. In *Proc. of COLT*, pages 320–332, 2010.

N. Cesa-Bianchi, C. Gentile, F. Vitale, and G. Zappella. A correlation clustering approach to link classification in signed networks. In *Proc. of COLT*, pages 34.1–34.20, 2012.

N. Cesa-Bianchi, Y. Mansour, and O. Shamir. On the complexity of learning with kernels. In *Proc. of COLT*, pages 297–325, 2015.

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

S. Chawla and A. Gionis. k-means-: A unified approach to clustering and outlier detection. In *Proceedings of the 13th SIAM International Conference on Data Mining, May 2-4, 2013. Austin, Texas, USA*, pages 189–197. SIAM, 2013. doi: 10.1137/1.9781611972832.21. URL https://doi.org/10.1137/1.9781611972832.21.

S. Chawla, K. Makarychev, T. Schramm, and G. Yaroslavtsev. Near optimal LP rounding algorithm for correlation clustering on complete and complete k-partite graphs. In *Proc. of ACM STOC*, pages 219–228, 2015.

B. Chazelle. A minimum spanning tree algorithm with inverse-Ackermann type complexity. *J. ACM*, 47(6):1028–1047, Nov. 2000. ISSN 0004-5411. doi: 10.1145/355541.355562.

Y. Chen, A. Jalali, S. Sanghavi, and H. Xu. Clustering partially observed graphs via convex optimization. *The Journal of Machine Learning Research*, 15(1):2213–2238, 2014.

Y. Chen, G. Kamath, C. Suh, and D. Tse. Community recovery in graphs with locality. In *Proc. of ICML*, pages 689–698, 2016.

K.-Y. Chiang, C.-J. Hsieh, N. Natarajan, I. S. Dhillon, and A. Tewari. Prediction and clustering in signed networks: a local to global perspective. *The Journal of Machine Learning Research*, 15(1):1177–1213, 2014.

E. Chien, A. Tulino, and J. Llorca. Active learning in the geometric block model. In *Proc. of AAAI*, volume 34, pages 3641–3648, 2020.

W. W. Cohen and J. Richman. Learning to match and cluster large high-dimensional data sets for data integration. In *Proc. of ACM KDD*, pages 475–480, 2002.

F. Cucker and S. Smale. On the mathematical foundations of learning. *American Mathematical Society*, 39(1):1–49, 2002.

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

A. Daniely and S. Shalev-Shwartz. Optimal learners for multiclass problems. In *Proc. of COLT*, volume 35, pages 287–316, 2014.

G. Dasarathy, R. Nowak, and X. Zhu. S2: An efficient graph based active learning algorithm with application to nonparametric classification. In *Proc. of COLT*, pages 503–522, 2015.

S. Dasgupta. Learning mixtures of Gaussians. In *Proc. of IEEE FOCS*, page 634, 1999. ISBN 0769504094.

S. Dasgupta. Analysis of a greedy active learning strategy. In *Advances in Neural Information Processing Systems 17*, pages 337–344, 2005.

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

I. Diakonikolas and D. M. Kane. Robust high-dimensional statistics. In T. Roughgarden, editor, *Beyond the Worst-Case Analysis of Algorithms*, pages 382–402. Cambridge University Press, 2020. doi: 10.1017/9781108637435.023. URL https://doi.org/10.1017/9781108637435.023.

S. Doyle, J. Monaco, M. Feldman, J. Tomaszewski, and A. Madabhushi. An active learning based classification strategy for the minority class problem: application to histopathology annotation. *BMC Bioinformatics*, 12(1), 2011.

D. Dubhashi and A. Panconesi. *Concentration of Measure for the Analysis of Randomized Algorithms*. Cambridge University Press, New York, NY, USA, 1st edition, 2009a. ISBN 0521884276, 9780521884273.

D. Dubhashi and A. Panconesi. *Concentration of Measure for the Analysis of Randomized Algorithms*. Cambridge University Press, New York, NY, USA, 1st edition, 2009b.

F. Dzogang, C. Marsala, M.-J. Lesot, and M. Rifqi. An ellipsoidal k-means for document clustering. In *Proc. of IEEE ICDM*, pages 221–230, 2012.

R. El-Yaniv and Y. Wiener. Active learning via perfect selective classification. *Journal of Machine Learning Research*, 13(2), 2012.

R. T. Elmore, T. P. Hettmansperger, and F. Xuan. Spherical data depth and a multivariate median. *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, 72: 87, 2006.

E. Emamjomeh-Zadeh and D. Kempe. Adaptive hierarchical clustering using ordinal queries. In *Proc. of ACM-SIAM SODA*, pages 415–429. SIAM, 2018.

B. Eriksson, G. Dasarathy, A. Singh, and R. Nowak. Active clustering: Robust and efficient hierarchical clustering using adaptively selected similarities. In *Proc. of AISTATS*, pages 260–268. JMLR Workshop and Conference Proceedings, 2011.

D. Firmani, S. Galhotra, B. Saha, and D. Srivastava. Robust entity resolution using a crowd oracle. *IEEE Data Eng. Bull.*, 41(2):91–103, 2018.

H. Fournier and O. Teytaud. Lower bounds for comparison based evolution strategies using vc-dimension and sign patterns. *Algorithmica*, 59(3):387–408, Mar. 2011. ISSN 0178-4617. doi: 10.1007/s00453-010-9391-3. URL https://doi.org/10.1007/s00453-010-9391-3.

R. Fraiman, F. Gamboa, and L. Moreno. Connecting pairwise geodesic spheres by depth: DCOPS. *J. Multivar. Anal.*, 169:81–94, 2019. doi: 10.1016/j.jmva.2018.08.008. URL https://doi.org/10.1016/j.jmva.2018.08.008.

M. L. Fredman, J. Komlos, and E. Szemeredi. Storing a sparse table with O(1) worst case access time. In *Proc. of IEEE SFCS*, page 165–169, 1982.

A. Gadde, E. E. Gad, S. Avestimehr, and A. Ortega. Active learning for community detection in stochastic block models. In *Proc. of IEEE ISIT*, pages 1889–1893. IEEE, 2016.

B. Gamlath, S. Huang, and O. Svensson. Semi-supervised algorithms for approximately optimal and accurate clustering. In *Proc. of ICALP*, pages 57:1–57:14, 2018. doi: 10.4230/LIPIcs.ICALP.2018.57.

L. A. Garcia-Escudero and A. Gordaliza. Robustness properties of k means and trimmed k means. *Journal of the American Statistical Association*, 94(447):956–969, 1999.

L. Getoor and A. Machanavajjhala. Entity resolution: theory, practice & open challenges. *Proc. of the VLDB Endowment*, 5(12):2018–2019, 2012.

A. Giannopoulos. Notes on isotropic convex bodies, 2003. URL http://users.uoa.gr/~apgiannop/isotropic-bodies.pdf.

A. Gonen, S. Sabato, and S. Shalev-Shwartz. Efficient active learning of halfspaces: an aggressive approach. *The Journal of Machine Learning Research*, 14(1):2583–2615, 2013.

T. F. Gonzalez. Clustering to minimize the maximum intercluster distance. *Theoretical Computer Science*, 38:293–306, 1985.

L.-A. Gottlieb and R. Krauthgamer. Proximity algorithms for nearly doubling spaces. *SIAM Journal on Discrete Mathematics*, 27(4):1759–1769, 2013.

L.-A. Gottlieb, A. Kontorovich, and P. Nisnevitch. Nearly optimal classification for semimetrics. *The Journal of Machine Learning Research*, 18(1):1233–1254, 2017.

A. Gruenheid, B. Nushi, T. Kraska, W. Gatterbauer, and D. Kossmann. Fault-tolerant entity resolution with the crowd. *CoRR*, abs/1512.00537, 2015. URL http://arxiv.org/abs/1512.00537.

A. Guillory and J. Bilmes. Active semi-supervised learning using submodular functions. In *Proc. of UAI*, pages 274–282, 2011.

F. R. Hampel. The influence curve and its role in robust estimation. *Journal of the american statistical association*, 69(346):383–393, 1974.

F. R. Hampel. Robust statistics: A brief introduction and overview. In *Research report/Seminar für Statistik, Eidgenössische Technische Hochschule (ETH)*, volume 94. Seminar für Statistik, Eidgenössische Technische Hochschule, 2001.

B. Han, Q. Yao, X. Yu, G. Niu, M. Xu, W. Hu, I. Tsang, and M. Sugiyama. Co-teaching: Robust training of deep neural networks with extremely noisy labels. In *Advances in neural information processing systems*, pages 8527–8537, 2018.

S. Hanneke. *Theoretical Foundations of Active Learning.* PhD thesis, Carnegie Mellon University, USA, 2009. AAI3362265.

S. Hanneke. Theory of disagreement-based active learning. *Foundations and Trends® in Machine Learning*, 7(2-3):131–309, 2014. doi: 10.1561/2200000037.

S. Hanneke and L. Yang. Minimax analysis of active learning. *The Journal of Machine Learning Research*, 16(1):3487–3602, 2015.

M. Hardt and E. Price. Tight Bounds for Learning a Mixture of Two Gaussians. In *Proc. of ACM STOC*, page 753–760, 2015. ISBN 9781450335362. doi: 10.1145/2746539.2746579.

P. W. Holland, K. B. Laskey, and S. Leinhardt. Stochastic blockmodels: First steps. *Social Networks*, 5(2):109 – 137, 1983.

M. Hopkins, D. M. Kane, and S. Lovett. The power of comparisons for actively learning linear classifiers. *arXiv preprint arXiv:1907.03816*, 2019.

M. Hopkins, D. Kane, S. Lovett, and M. Moshkovitz. Bounded memory active learning through enriched queries. *CoRR*, abs/2102.05047, 2021. URL https://arxiv.org/abs/2102.05047.

W. Huleihel, A. Mazumdar, M. Médard, and S. Pal. Same-cluster querying for overlapping clusters. In *Advances in Neural Information Processing Systems 32*, pages 10485–10495, 2019.

A. T. Kalai, A. Moitra, and G. Valiant. Efficiently learning mixtures of two Gaussians. In *Proc. of ACM STOC*, pages 553–562, 2010.

K. Kameyama and Y. Kosugi. Semiconductor defect classification using hyperellipsoid clustering neural networks and model switching. In *Proc. of IJCNN'99 (Cat. No. 99CH36339)*, volume 5, pages 3505–3510. IEEE, 1999.

D. M. Kane, S. Lovett, S. Moran, and J. Zhang. Active classification with comparison queries. In *Proc. of IEEE FOCS*, pages 355–366, 2017a. doi: 10.1109/FOCS.2017.40.

D. M. Kane, S. Lovett, S. Moran, and J. Zhang. Active classification with comparison queries. In *Proc. of IEEE FOCS*, pages 355–366, 2017b. doi: 10.1109/FOCS.2017.40.

L. G. Khachiyan. Rounding of polytopes in the real number model of computation. *Mathematics of Operations Research*, 21(2):307–320, 1996a. ISSN 0364765X, 15265471. URL http://www.jstor.org/stable/3690235.

L. G. Khachiyan. Rounding of polytopes in the real number model of computation. *Mathematics of Operations Research*, 21(2):307–320, 1996b.

S. Kim, S. Nowozin, P. Kohli, and C. D. Yoo. Higher-order correlation clustering for image segmentation. In *Proc. of NeurIPS*, pages 1530–1538, 2011.

J. Kivinen. Learning reliably and with one-sided error. *Mathematical Systems Theory*, 28(2): 141–172, 1995. doi: 10.1007/BF01191474.

A. Krishnamurthy, S. Balakrishnan, M. Xu, and A. Singh. Efficient active algorithms for hierarchical clustering. In *Proc. of ICML*, pages 267–274, 2012.

B. Kulis. Metric learning: A survey. *Foundations and Trends in Machine Learning*, 5(4): 287–364, 2013.

M. P. Kumar, B. Packer, and D. Koller. Self-paced learning for latent variable models. In *Advances in neural information processing systems*, pages 1189–1197, 2010.

A. Kupavskii. The VC-dimension of k-vertex d-polytopes. *Combinatorica*, 40(6):869–874, 2020. doi: 10.1007/s00493-020-4475-4.

J. Lee and M. Raginsky. Minimax statistical learning with wasserstein distances. In *Advances in Neural Information Processing Systems*, volume 31, pages 2687–2696, 2018.

J. Lee and M. Raginsky. Learning finite-dimensional coding schemes with nonlinear reconstruction maps. *SIAM Journal on Mathematics of Data Science*, 1(3):617–642, 2019.

J. Lee, S. Park, and J. Shin. Learning bounds for risk-sensitive learning. *arXiv preprint arXiv:2006.08138*, 2020.

S. Li and O. Svensson. Approximating k-median via pseudo-approximation. *SIAM Journal on Computing*, 45(2):530–547, 2016.

E. Lloyd. Least-squares estimation of location and scale parameters using order statistics. *Biometrika*, 39(1/2):88–95, 1952.

L. Lovász and S. Vempala. Hit-and-run from a corner. *SIAM Journal on Computing*, 35(4): 985–1005, 2006.

S. T. Mai, X. He, N. Hubig, C. Plant, and C. Böhm. Active density-based clustering. In *Proc. of IEEE ICDM*, pages 508–517. IEEE, 2013.

V.-G. Marica. Hyper-ellipsoid clustering of time series: A case study for daily stock returns. *Procedia Economics and Finance*, 15:777–783, 2014.

P. Massart. The tight constant in the dvoretzky-kiefer-wolfowitz inequality. *The annals of Probability*, pages 1269–1283, 1990.

L. Massoulié. Community detection thresholds and the weak ramanujan property. In *Proc. of ACM STOC*, pages 694–703. ACM, 2014.

A. Maurer and M. Pontil. k-dimensional coding schemes in Hilbert spaces. *IEEE Transactions on Information Theory*, 56(11):5839–5846, 2010.

A. Maurer and M. Pontil. Empirical bounds for functions with weak interactions. *arXiv preprint arXiv:1803.03934*, 2018.

A. Maurer and M. Pontil. Uniform concentration and symmetrization for weak interactions. *arXiv preprint arXiv:1902.01911*, 2019.

A. Maurer et al. A bernstein-type inequality for functions of bounded interaction. *Bernoulli*, 25(2):1451–1471, 2019.

A. Mazumdar and S. Pal. Semisupervised Clustering, AND-Queries and Locally Encodable Source Coding. In *Advances in Neural Information Processing Systems 30*, pages 6489–6499, 2017.

A. Mazumdar and B. Saha. Query complexity of clustering with side information. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 4682–4693, 2017a.

A. Mazumdar and B. Saha. Clustering with noisy queries. In *Advances in Neural Information Processing Systems 30*, pages 5788–5799, 2017b.

A. McCallum and K. Nigam. Employing em and pool-based active learning for text classification. In *Proc. of ICML*, page 350–358, 1998.

M. Moshtaghi, T. C. Havens, J. C. Bezdek, L. Park, C. Leckie, S. Rajasegarar, J. M. Keller, and M. Palaniswami. Clustering ellipses for anomaly detection. *Pattern Recognition*, 44 (1):55–69, 2011.

E. Mossel, J. Neeman, and A. Sly. A proof of the block model threshold conjecture. *Combinatorica*, 38(3):665–708, 2018.

R. Motwani and P. Raghavan. *Randomized Algorithms*. Cambridge University Press, USA, 1995. ISBN 0521474655.

M. Naszódi. Approximating a convex body by a polytope using the epsilon-net theorem. *Discrete & Computational Geometry*, 61(3):686–693, Mar. 2018. doi: 10.1007/s00454-018-9977-0.

R. D. Nowak. The geometry of generalized binary search. *IEEE Transactions on Information Theory*, 57(12):7893–7906, 2011.

K. Ord. Outliers in statistical data : V. Barnett and T. Lewis, 1994, 3rd edition, (John Wiley & Sons, Chichester), 584 pp., [UK pound]55.00, ISBN 0-471-93094-6. *International Journal of Forecasting*, 12(1):175–176, March 1996. URL https://ideas.repec.org/a/eee/intfor/v12y1996i1p175-176.html.

I. M. Pelayo. *Geodesic convexity in graphs*. Springer-Verlag New York, 2013. ISBN 978-1-4614-8698-5. doi: 10.1007/978-1-4614-8699-2.

L. A. Rademacher. Approximating the centroid is hard. In *Proc. of ACM SoCG*, page 302–305, 2007. doi: 10.1145/1247069.1247123.

R. L. Rivest and R. Sloan. Learning complicated concepts reliably and usefully. In *Proc. of COLT*, pages 69–79, 1988.

B. I. Rubinstein, P. L. Bartlett, and J. H. Rubinstein. Shifting: One-inclusion mistake bounds and sample compression. *Journal of Computer and System Sciences*, 75(1):37–59, 2009.

B. Saha and S. Subramanian. Correlation clustering with same-cluster queries bounded by optimal cost. In *Proc. of ESA*, pages 81:1–81:17, 2019a.

B. Saha and S. Subramanian. Correlation Clustering with Same-Cluster Queries Bounded by Optimal Cost. In *Proc. of ESA*, pages 81:1–81:17, 2019b. doi: 10.4230/LIPIcs.ESA.2019.81.

D. Sanyal and S. Das. On semi-supervised active clustering of stable instances with oracles. *Information Processing Letters*, 151:105833, 2019.

F. Seiffarth, T. Horváth, and S. Wrobel. Maximal closed set and half-space separations in finite closure systems. In *Proc. of ECML PKDD*, pages 21–37. Springer, 2019.

R. J. Serfling. *Approximation theorems of mathematical statistics*, volume 162. John Wiley & Sons, 1980.

S. Shalev-Shwartz and S. Ben-David. *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press, USA, 2014a. ISBN 1107057132.

S. Shalev-Shwartz and S. Ben-David. *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press, New York, NY, USA, 2014b.

S. Shalev-Shwartz and S. Ben-David. *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press, USA, 2014c. ISBN 1107057132.

O. Shamir and N. Tishby. Spectral clustering on a budget. In *Proc. of AISTATS*, pages 661–669. JMLR Workshop and Conference Proceedings, 2011.

M. Shuhua, W. Jinkuan, and L. Zhigang. Ellipsoids clustering algorithm based on the hierarchical division for WSNs. In *Proc. of IEEE CCC*, pages 7394–7397. IEEE, 2013.

J. Tang, Y. Chang, C. Aggarwal, and H. Liu. A survey of signed network mining in social media. *ACM Computing Surveys (CSUR)*, 49(3):42, 2016.

M. Thiessen and T. Gärtner. Active learning on graphs with geodesically convex classes. In *Proc. of MLG*, 2020.

M. J. Todd. *Minimum-Volume Ellipsoids.* SIAM, Philadelphia, PA, 2016. doi: 10.1137/1.9781611974386. URL https://epubs.siam.org/doi/abs/10.1137/1.9781611974386.

S. Tong and E. Chang. Support vector machine active learning for image retrieval. In *Proc. of ACM ICM*, page 107–118, 2001.

D. Vainsencher, S. Mannor, and A. M. Bruckstein. The sample complexity of dictionary learning. In *Proceedings of the 24th Annual Conference on Learning Theory*, pages 773–788. JMLR Workshop and Conference Proceedings, 2011.

V. Verroios and H. Garcia-Molina. Entity resolution with crowd errors. In *Proc. of IEEE ICDE*, pages 219–230, 2015.

V. Verroios, H. Garcia-Molina, and Y. Papakonstantinou. Waldo: An adaptive human interface for crowd entity resolution. In *Proc. of ACM SIGMOD*, pages 1133–1148, 2017.

R. Vershynin. *High-Dimensional Probability: An Introduction with Applications in Data Science.* Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, 2018. doi: 10.1017/9781108231596.

F. Vitale, A. Rajagopalan, and C. Gentile. Flattening a hierarchical clustering through active learning. In *Advances in Neural Information Processing Systems 32*, pages 15263–15273, 2019.

X. Wang and I. Davidson. Active spectral clustering. In *Proc. of IEEE ICDM*, pages 561–568. IEEE, 2010.

A. Wirth. Correlation Clustering. In C. Sammut and G. I. Webb, editors, *Encyclopedia of machine learning and data mining*, pages 227–231. Springer US, 2010.

L. Xu, J. Neufeld, B. Larson, and D. Schuurmans. Maximum margin clustering. *Advances in Neural Information Processing Systems*, pages 1537–1544, 2004.

P. Zhang, C. Moore, and L. Zdeborová. Phase transitions in semisupervised clustering of sparse networks. *Physical Review E*, 90(5):052802, 2014.