Small and Stable Descriptors of Distributions for Geometric Statistical Problems

by

Jeff M. Phillips

Department of Computer Science
Duke University

Ph.D. Dissertation
2009

Abstract

This thesis explores how to sparsely represent distributions of points for geometric statistical problems. A coreset C is a small summary of a point set P such that if a certain statistic is computed on P and C, then the difference in the results is guaranteed to be bounded by a parameter ε. Two examples of coresets are ε-samples and ε-kernels. An ε-sample can estimate the density of a point set in any range from a geometric family of ranges (e.g., disks, axis-aligned rectangles). An ε-kernel approximates the width of a point set in all directions. Both coresets have size that depends only on ε, the error parameter, not the size of the original data set. We demonstrate several improvements to these coresets and how they are useful for geometric statistical problems.

We reduce the size of ε-samples for density queries in axis-aligned rectangles to nearly a square root of the size when the queries are with respect to more general families of shapes, such as disks. We also show how to construct ε-samples of probability distributions.

We show how to maintain "stable" ε-kernels, that is, if the point set P changes by a small amount, then the ε-kernel also changes by a small amount. This is useful in surveillance and tracking problems, and the stable properties leads to more efficient algorithms for maintaining ε-kernels.

We next study when the input point sets are uncertain and their uncertainty is modeled by probability distributions. Statistics on these point sets (e.g., radius of smallest enclosing ball) do not have exact answers, but rather distributions of answers. We describe data structures to represent approximations of these distributions and algorithms to compute them. We also show how to create distributions of ε-kernels and ε-samples for these uncertain data sets.

Finally, we examine a spatial anomaly detection problem: computing a spatial scan statistic. The input is a point set P and measurements on the point set. The spatial scan statistic finds the range (e.g., an axis-aligned bounding box) where the measurements inside the range are the most different from measurements outside of the range. We show how to compute this statistic efficiently while allowing for a bounded amount of approximation error. This result generalizes to several statistical models and types of input point sets.

To the memory of Andrew Mark Ladd

Contents

List of Tables

List of Figures

Acknowledgements

I want to thank my advisor Pankaj Agarwal for his consistent and pointed advice. In retrospect, it was always surprising how accurate and precise his feedback and instructions were. I greatly appreciate his patience while I learned to follow in his clear and rigorous approach to geometry and research.

Also want to thank the rest of my committee Suresh Venkatasubramanian, Herbert Edelsbrunner, Kamesh Munagala, and Alan Gelfand for providing occasional, but meaningful, guidance on the process towards my dissertation. I also want to thank Suresh for his patience and time early in my career as a theory student.

While at Duke I have interacted with many others in my research who have provided feedback, inspiration, and direction. To list a few: Hai Yu, Shashidhara Ganjugunte, Yuriy Mileyko, Bardia Sadri, Carlo Tomasi, Ron Parr, Jun Yang, Lars Arge, Don Rose, John Reif, Johannes Rudolph, Sayan Mukherjee, Dmitriy Moroziv, Amit Patel, Thomas Moelhave, Albert Yu, Sharathkumar, Kevin Yi, Andrew Danner, Esther Ezra, Jeff Headd, Nabil Mustafa, Yusu Wang, Madhu Vaidya, Ran Liu, and Badrish Chandramoulli. Thank you all for providing an interesting and collaborative research environment. I thank Diane Riggs and the rest of the Duke Computer Science support staff for watching my back on the administrative details so I could focus on research.

I also want to acknowledge Deepak Agarwal, Sariel Har-Peled, Jirka Matoušek, Subhash Suri, Maarten Löffler, Andrew McGregor, and Zhengyuan Zhu for making contributions to the ideas and content of this thesis.

I also need to thank those at Rice for helping shape my interests in computer science and research, especially Lydia Kavraki, Andrew Ladd, Oliver Brock, Naz Bedrossian, and Robin Forman. Lydia not only introduced me to research but also cultivated my interest and skills; she taught me the importance of organization. Andrew served as a personal research mentor, friend, and role model for my time at Rice. His contagious enthusiasm is what convinced me to go to graduate school and devote myself to rigorous research. Sadly, he did not make it to see me complete this thesis, and it is dedicated in his memory.

I also acknowledge the funding that made this thesis possible. I thank the National Science Foundation and the Duke Graduate School for fellowships directly funding my research. I also thank the funding I received through grants of my advisor Pankaj Agarwal from NSF, ARO, NIH, DOE, and U.S.-Israel Binational Science

Foundation.

I thank Bei Wang for the friendship, patience, feedback, inspiration, respite, and love that she provided throughout the process of creating this thesis. She looked after my health, cleanliness, and appetite when I was more concerned with other things.

Finally, I want to thank my parents John and Geri Phillips for their constant support, confidence, and pride. Their encouragement throughout this process led to resolved and constant progress. I also want to thank my dad for instilling in me respect for mathematics and science and a desire to always find out how far I could go.

Introduction

This thesis describes how to create small summaries of large data sets to answer geometric statistical questions. For several problems we decrease the size of the summary or increase the speed of the algorithm while preserving or improving accuracy. We also develop new models; in particular, we define the stability of descriptors for dynamic data, and we propose a new way to represent statistics on uncertain input data.

Sensed Data

In gathering data there is a trade-off between quantity and accuracy. The drop in the price of hard drives and other storage costs has shifted this balance towards gathering enormous quantities of data, yet with noticeable and sometimes intentional imprecision. However, often as a benefit from the large data sets, models are developed to describe the pattern of the data error. To properly analyze this data, computational and statistical-based analysis should be used to compress the data to a useful form.

Let us take as an example Light Detection and Ranging (LIDAR) data gathered for Geographic Information Systems (GIS) [76], specifically height values at millions of locations on a terrain. Each data point (x, y, z) has an x-value (longitude), a y-value (latitude), and a z-value (height). This data set is gathered by a small plane flying over a terrain with a laser aimed at the ground measuring the distance from the plane to the ground. Error can occur due to inaccurate estimation of the plane's altitude and position or artifacts on the ground distorting the laser's distance reading. But these errors are well-studied and can be modeled by replacing each data point with a probability distribution of its actual position. Greatly simplifying, we could represent each data point as a 3-variate normal distribution centered at its recorded value. Also, as artifacts (e.g., bridges, mailboxes) are discovered and corrected the data may by dynamically updated.

Similarly, large data sets are gathered and maintained for many other applications. In robotic mapping [113, 42] error models are provided for data points gathered by laser range finders and other sources. In data mining [1, 12] original data (such as published medical data) are often perturbed by a known model to preserve anonymity. In spatial databases [51, 106, 33] large data sets may be summarized as probability distributions to store them more compactly, and can be often updated. Sensor networks [38] stream in large and changing data sets collected by cheap and thus inaccurate sensors. In protein structure determination [103] every atom's position is imprecise due to inaccuracies in reconstruction techniques and the inherent flexibility in the protein. In summary, there are many large data sets with modeled errors and dynamic updates.

Coresets

In the last decade, much work has focused on computing a "small summary" of large data sets and on obtaining a good trade-off between the quality of the summary and its size. One can then compute statistics on the summary instead of the entire data set and prove that the statistic computed on the summary is a good approximation of the optimal solution on the entire data set. This approach, however, generally assumes that the original raw data are correct or that the errors generated by the summary are comparable with the sensing errors and thus are acceptable.

Coresets, such as ε-samples [31] (often referred to as ε-approximations) and α-kernels [4, 5], are examples of such approximate summaries (we described these incarnations in more detail below). Specifically, for an input set P, a *coreset* $C \subset P$ is a (hopefully small) subset of P with the property that when certain functions are calculated on C instead of P, the error is bounded in terms of ε or α. The power of a coreset-based based approach is that we can use any coreset algorithm to approximate a data set and then run the same algorithm on the coreset instead of the original set, and thereby substantially reducing the required runtime.

ε-Samples. An ε-sample (often called an ε-approximation) is defined for a data set P, often called a *ground set*, and a family of subsets of the data $\mathcal{A}$, often called *ranges*. This pairing of ground set and ranges is called a *range space*. For instance, the data set P may be a point set describing the location of every household in the US, and the family of subsets $\mathcal{A}$ may be all subsets of households within a fixed radius from a query point (i.e. within a circular disk). Such a subset is shaded in Figure 1(b). An *ε-sample* is a subset $Q \subset P$ of the ground set such that for any range $R \in \mathcal{A}$, the fraction of points from Q in R is different from the fraction of points from P in R by at most ε. Thus if Q is an ε-sample of US households and disks, and we ask what fraction of the households in the US are within 150 miles of Philadelphia, we can give an answer within ε (the answer might be 0.11 with $\varepsilon = 0.02$). Amazingly, it has been shown [116] that a random sample of households of size proportional to $(1/\varepsilon^2)\log(1/\varepsilon\delta)$ is an ε-sample with probability $\geq 1 - \delta$. Thus if we sample on the order of $10{,}000$ households, then with probability ≥ 0.99, we can answer any such query within error $\varepsilon = 0.02$. Furthermore, the same number of samples would work for the entire world with the same guarantees and for any query disk (e.g., the fraction of world households within 600 miles of Zürich).

α-Kernels. An α-kernel approximates the extreme points for a set of d-dimensional points $P \subset \mathbb{R}^d$, so that it approximately preserves the width in every direction. Let

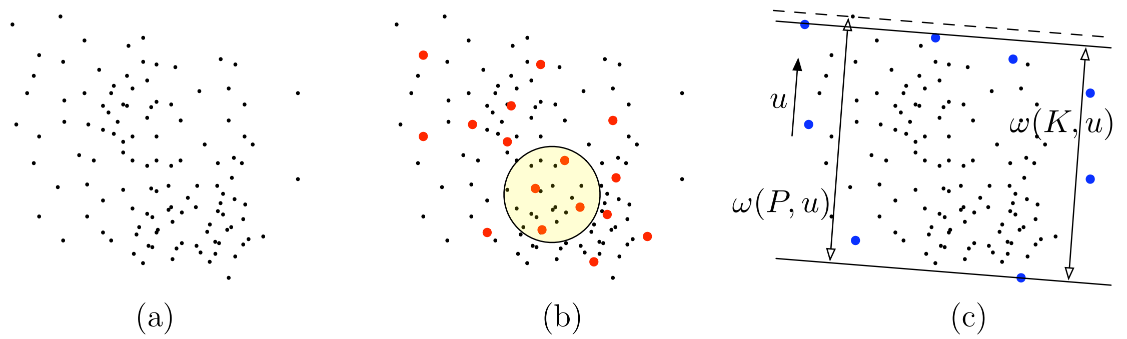
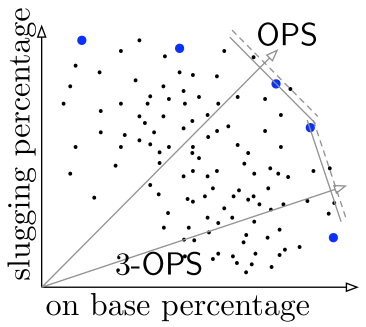


FIGURE 1: (a) A point set $P \subset \mathbb{R}^2$. (b) An ε-sample $Q \subset P$ with respect to disks, highlighted. (c) An α-kernel $K \subset P$ highlighted.

$\omega(P, u)$ be the width of a point set in direction u, see Figure 1(c). An α-*kernel* $K \subset P$ approximates P so that in any direction u the following holds $\omega(K, u) \leq \omega(P, u) \leq (1 + \alpha) \cdot \omega(K, u)$. We can construct α-kernels of size proportional to $1/\alpha^{(d-1)/2}$ in time linear in the size of the original data set [4]. For example, consider the set of major league baseball players. For a particular season we can compute statistics of how well they played, for instance, their *on base percentage* and *slugging percentage*. Then we can represent the ith player as a two-dimensional point $p_i = (b_i, s_i)$ where b_i is his on base percentage and s_i is his slugging percentage. Let P be the set of all players, each represented as a 2-dimensional point. A popular statistic is *on base plus slugging* (OPS) which for player i is described $b_i + s_i$. So the maximum OPS among all players is achieved by the extreme point in P in direction u, where $u = (1, 1)$ — this vector weights b_i and s_i equally. However, say we wanted to compute a variation of OPS called γ-OPS that weights on base percentage by γ and for player i is described as $\gamma b_i + s_i$. Then the player with the maximum γ-OPS would be the extreme point in P in direction u_γ where $u_\gamma = (\gamma, 1)$ — this vector weights b_i γ times what it weights s_i. If we have an α-kernel $K \subset P$ of all players, then we can approximate the best γ-OPS in P by the most extreme point in K in direction u_γ. This approximate maximum γ-OPS will be within a $(1 + \alpha)$ factor of the true maximum. If we keep more statistics for each player, then each player represents a higher dimensional point; and if we construct a higher dimensional α-kernel of each player, then for any linear combination of their statistics we can approximate the best value within a $(1 + \alpha)$ factor.



Coresets exist for other problems, such as the smallest enclosing ball [25], k-center clustering [10, 54], and shape fitting [119]. Additionally, ε-nets [58], a weaker form of ε-samples, are coresets.

However, it is not always realistic or possible to assume that the input data are completely accurate, an assumption made by most generic algorithms for constructing coresets. For instance, the US census does not release the exact location of all households, just the number in each zip code. And we may want to predict a baseball player's statistics for the next year using a model for each player. Incorporating these error models into the above examples could be visualized by replacing each point p in Figure 1 with a normal distribution centered at p describing the possible location of the true value of p. These normal distributions then model the uncertainty of the point sets. Alternatively, we may find an error in the statistics of a certain baseball player and may want to update our α-kernel without rebuilding the entire α-kernel.

Error Parameters

The algorithms we study in this thesis allow for three types of error parameters, which for notational convenience we consistently denote with different Greek letters: ε, α, and δ.

- Relative counting error is set by the parameter ε, such as in ε-samples. This occurs in cases where we report a fraction of the total data set and are off in this value by at most ε. For instance, we can estimate that Barack Obama received 0.53 fraction (or 53%) of the 2008 popular vote for President with an error of $\varepsilon = 0.005$, if the true value is between 0.525 and 0.535.

- Relative geometric error is set by the parameter α, such as in α-kernels. This occurs when some extent measure on a point set is off by a $(1+\alpha)$ factor. For example, we can report that the diameter (distance between two furthest points) of Earth is 12770km with an error of at most 50km, then $\alpha = 50/12770 \leq 0.004$ (i.e. the true diameter is between 12720km and 12820km).

- Randomization error is set by the parameter δ. This occurs for a randomized algorithm that may fail with probability δ. For example, two friends may have 10 identical outfits and would not want to both wear the same one on the first day of school. If each chooses an outfit at random (this is a randomized algorithm), then with probability $\delta = 1/10$ they will wear the same outfit and fail. An algorithm where $\delta = 0$ is said to be *deterministic*.

The coresets we compute will have size dependent only on these parameters, not the size of the input data. As a general rule, as the error parameters increase, the algorithms run faster and produce smaller size summaries, but allow for larger errors.

Contributions of this Thesis.

This thesis improves on the construction of ε-samples and α-kernels in several ways relevant to calculating statistics on dynamic and uncertain data. It builds several algorithmic techniques that can be used for statistical geometric problems. It also illustrates some of these techniques on a specific problem.

Chapter 1 describes a deterministic algorithm for computing ε-samples of size $(1/\varepsilon)\log^{O(1)}(1/\varepsilon)$ for range spaces where the ground set is a point set in $\mathbb{R}^d$ and the ranges are defined by a family of shapes that includes axis-aligned rectangles. We also describe how to construct small ε-samples when the ground set represents a probability distribution or a terrain. This chapter is mainly based on [96].

Chapter 2 describes an algorithm for dynamically maintaining an α-kernel under dynamic insertions and deletions of points so that when one point is changed in the original set a constant number of points change in the α-kernel. We call such an α-kernel *stable*. The described algorithm is also the fastest known dynamic algorithm for α-kernels, even if they are not required to be stable. This chapter also explores the stability of the minimal size of an α-kernel with respect to the parameter α. This chapter is based on [9] with Pankaj K. Agarwal and Hai Yu.

Chapter 3 focuses on data sets where instead of exact and precise point sets, each point is modeled by a probability distribution. For such data sets, the answers to statistical queries are not exact answers, but distributions of answers. Thus we produce answers in the form of data structures to represent different types of distributions called ε-quantizations and ε-sip functions. This work heavily uses results developed in Chapter 1 and develops the extension of α-kernels for this sort of data. This chapter is based on [79] with Maarten Löffler.

Chapter 4 calculates spatial scan statistics, a spatial anomaly detection problem. We provide algorithms with guaranteed approximation factors which perform as well as or better than heuristic algorithms. This can be extended to deal with data of the form focused on in Chapter 3. This chapter is based on two papers: The first [3] is with Deepak Agarwal and Suresh Venkatasubramanian, and the second [2] is with Deepak Agarwal, Andrew McGregor, Suresh Venkatasubramanian, and Zhengyuan Zhu.

1
Algorithms for ε-Samples of Distributions

Motivation

Representing complex objects by point sets (i.e. ε-samples) may require less storage and may make computation on them faster and easier. For instance discrete versions of terrains or probability distributions may be represented as piecewise-linear domains, but range volume queries over these domains requires complex integration. Finding ranges which optimize statistics over piecewise-linear domains can reduce to algebraic problems with no closed form solutions. Conversely, comparable queries on domains represented as discrete point sets are often well studied with efficient solutions, usually reducing to just counting the data points in a range. Because polygonal domains usually already approximate a smooth domain and because solutions on them cannot always be solved without numerical methods, it motivates the use of other simpler forms of approximation, such as using discrete point sets.

Furthermore, it may be necessary to approximate a domain with a discrete point set. Consider a domain that represents the distribution of some quantity that needs to be gathered or monitored and for which we wish to deploy a set of sentinel nodes proportional to the density of the domain. For example, given a distribution of where crimes may occur, we may wish to place a set of police precincts so that within any range, the proportions of crimes and of precincts are within some bound.

Alternatively, if the data is already a discrete point set, approximating it with a much smaller point set relative to a fixed error tolerance can be used to greatly reduce the size of a data set with limited sacrifice in quality. Other applications include sensor networks, where a set of monitoring nodes are spread over a region. We may want to monitor an area for large events that affect all nodes in a range while conserving battery power and only keeping a fraction of the nodes (sentinel nodes) on at any given time. The smaller the size of the sentinel node set, the longer the same set of nodes can monitor an area by alternating which are turned on at any given time.

This chapter focuses on algorithms for creating ε-samples, in particular those which represent distributions as discrete point sets. Our main contributions are two-fold. First, we describe for discrete distributions and range spaces using rectilinear ranges how to deterministically create ε-samples of size $(1/\varepsilon)\log^{O(1)}(1/\varepsilon)$. This produces sets smaller than those created by deterministic algorithms for general range spaces or those created by random sampling techniques. Second, we extend these techniques to domains which represent Lebesgue-measurable point sets (e.g. probability distributions and terrains), for rectilinear ranges as well as more general families of ranges. We begin with definitions and background on ε-samples, as well as on discrepancy, which is the basis for most deterministic ε-sample algorithms.

1.1 Definitions and Background

This section provides basic information on ε-samples and discrepancy. The subsection on ε-samples will be useful not only in this chapter, but also later in this thesis. Discrepancy provides the basis for deterministic ε-sample algorithms, specifically, combinatorial discrepancy for discrete points sets and Lebesgue discrepancy for Lebesgue-measureable point sets such as probability distributions and terrains.

1.1.1 ε-Samples

In this chapter we mainly study point sets, which we call ground sets and we label as P. These ground sets are usually either finite sets or are Lebesgue-measureable sets. For a given ground set P, let $\mathcal{A}$ be a set of subsets of P induced by containment in some geometric shape (such as balls or axis-aligned rectangles). The pair $(P, \mathcal{A})$ is called a *range space*. Let $\mu(\cdot)$ represent the cardinality of a discrete set or the Lebesgue measure for a Lebesgue-measurable set. We say that Q is an *ε-sample* of $(P, \mathcal{A})$ if

$$\max_{R \in \mathcal{A}} \left| \frac{\mu(R \cap Q)}{\mu(Q)} - \frac{\mu(R \cap P)}{\mu(P)} \right| \leq \varepsilon.$$

$\mathcal{A}$ is said to *shatter* a discrete set $X \subseteq P$ if each subset of X is equal to $R \cap X$ for some $R \in \mathcal{A}$. The cardinality of the largest discrete set X that $\mathcal{A}$ can shatter is known as the *VC-dimension.* A classic result of Vapnik and Chervonenkis [116], improved by Li, Long, and Srinivasan [74][1] states that for any range space $(P, \mathcal{A})$

[1] Consider the metric $d_\theta(a,b) = |a-b|/(a+b+\theta)$ [57]. Li, Long, and Srinivasan [74] show that if $Q \subset P$ is a random sample of size $O((1/\varepsilon^2\theta)(\nu\log(1/\theta) + \log(1/\delta))$ then with probability at least $1-\delta$ that for all $R \in \mathcal{A}$ $d_\theta(\mu(R \cap Q)/\mu(Q), \mu(R \cap P)/\mu(P)) \leq \varepsilon$. Thus, setting $\theta = 1$,

$$\left| \frac{\mu(R \cap Q)}{\mu(Q)} - \frac{\mu(R \cap P)}{\mu(P)} \right| \leq \varepsilon \left(\frac{\mu(R \cap Q)}{\mu(Q)} + \frac{\mu(R \cap P)}{\mu(P)} + \theta \right) \leq 3\varepsilon.$$

Hence, after scaling ε by $1/3$, if $Q \subset P$ is a random sample from P of size $O((1/\varepsilon^2)(\nu + \log(1/\delta))$ it is an ε-sample with probability at least $1-\delta$.

with constant VC-dimension ν there exists a subset $Q \subset P$ consisting of $O((\nu/\varepsilon^2))$ points that is an ε-sample for $(P, \mathcal{A})$. They also show, if each element of Q is drawn uniformly at random from P such that $|Q| = O((1/\varepsilon^2)(\nu + \log(1/\delta)))$, then Q is an ε-sample with probability at least $1 - \delta$. Thus, for a large class of range spaces random sampling produces an ε-sample of size $O((1/\varepsilon^2))$. There exist ε-samples of slightly smaller sizes [84], but efficient constructions are not known. If $(P, \mathcal{A})$ has VC-dimension ν, this also implies that $(P, \mathcal{A})$ contains at most $|P|^\nu$ sets.

Similarly, the *shatter function* $\pi_{(P,\mathcal{A})}(m)$ of a range space $(P, \mathcal{A})$ is the maximum number of sets $S \in (P, \mathcal{A})$ where $|S| = m$. The *shatter dimension* σ of a range space $(P, \mathcal{A})$ is the minimum value such that $\pi_{(P,\mathcal{A})}(m) = O(m^\sigma)$. It can be shown [55] that $\sigma \leq \nu$ and $\nu = O(\sigma \log \sigma)$. For a range space $(P, \mathcal{A}')$ where each element $A' \in \mathcal{A}'$ is described by a function of k ranges $A_1, \ldots, A_k \in \mathcal{A}$, then $(P, \mathcal{A}')$ has VC-dimension $O(\nu k \log k)$ [55].

For a range space $(P, \mathcal{A})$ the *dual range space* is defined $(\mathcal{A}, P^*)$ where P^* is all subsets $\mathcal{A}_p \subseteq \mathcal{A}$ defined for an element $p \in P$ such that $\mathcal{A}_p = \{A \in \mathcal{A} \mid p \in A\}$. If $(P, \mathcal{A})$ has VC-dimension ν, then $(\mathcal{A}, P^*)$ has VC-dimension $\leq 2^{\nu+1}$. Thus, if the VC-dimension of $(\mathcal{A}, P^*)$ is constant, then the VC-dimension of $(P, \mathcal{A})$ is also constant [82]. Hence, the standard ε-sample theorems apply to dual range spaces as well.

Let $g : \mathbb{R} \to \mathbb{R}^+$ be a function where $\int_{x=-\infty}^{\infty} g(x)\ dx = 1$. We can create an ε-sample Q_g of $(g, \mathcal{I}_+)$, where $\mathcal{I}_+$ describes the set of all one-sided intervals of the form $(-\infty, t)$, so that

$$\max_t \left| \int_{x=-\infty}^{t} g(x)\ dx - \frac{1}{|Q_g|} \sum_{q \in Q_g} 1(q < t) \right| \leq \varepsilon.$$

We can construct Q_g of size $O(1/\varepsilon)$ by choosing a set of points in Q_g so that the integral between two consecutive points is always ε. But we do not need to be so precise. Consider the set of $2/\varepsilon$ points $\{q'_1, q'_2, \ldots, q'_{2/\varepsilon}\}$ such that $\int_{x=-\infty}^{q'_i} = i\varepsilon/2$. Any set of $2/\varepsilon$ points $Q_g = \{q_1, q_2, \ldots, q_{2/\varepsilon}\}$ such that $q'_i \leq q_i \leq q'_{i+1}$ is an ε-sample.

A shape $P \subset \mathbb{R}^{d+1}$ may describe a distribution $\mu : \mathbb{R}^d \to [0, 1]$. Specifically, for a point $(p, h) \in \mathbb{R}^d \times \mathbb{R}^+$, $(p, h) \in P$ if $h \leq \mu(p)$. Given a family of subsets of $\mathbb{R}^d$, this represents a probability space $(\mathbb{R}^d, \mathcal{A}, \mu)$. For notational convenience, we sometimes represent this probability space as a range space as $(\mu, \mathcal{A})$. We note that many common distributions μ, such as multivariate Gaussian distributions, can be approximated by a polygonal shape $P \in \mathbb{R}^d$. We call such distributions *polygonally approximable* (see Section 1.4.3).

Deterministic construction of ε-samples. There exist deterministic constructions for ε-samples. When P is the unit cube $[0, 1]^d$ there are constructions which can be interpreted as ε-samples of size $O(1/\varepsilon^{2d/(d+1)})$ for half spaces [81] and $O((1/\varepsilon^{2d/(d+1)}) \cdot \log^{d/(d+1)}(1/\varepsilon))$ for balls in d-dimensions [21]. Both have the same lower bounds of $\Omega(1/\varepsilon^{2d/(d+1)})$ [13]. See Matoušek [82] or Chazelle [31] for more similar results. For a domain P, let $\mathcal{R}_d$ describe the subsets induced by axis-parallel rectangles in d dimensions, and let $\mathcal{Q}_k$ describe the subsets induced by k-oriented polygons (or more generally polytopes) with faces described by k predefined normal directions. More precisely, for $\beta = \{\beta_1, \ldots, \beta_k\} \subset \mathbb{S}^{d-1}$, let $\mathcal{G}_\beta$ describe the set of convex polytopes such that each face has an outward normal $\pm\beta_i$ for $\beta_i \in \beta$. If β is fixed, for a ground set P we will use $\mathcal{Q}_k$ to denote the subsets of P described by inclusion in a polytope from $\mathcal{G}_\beta$ since it is the size k and not the actual set β that is important. When $P = [0,1]^d$, then the range space $(P, \mathcal{R}_d)$ has an ε-sample of size $O((1/\varepsilon)\log^{d-1}(1/\varepsilon))$ [52]. Also, for range space $(\mathbb{R}^d, \mathcal{A})$ where $\mathcal{A}$ is defined as all homothets (translations and uniform scalings) of any particular $G \in \mathcal{G}_k$, Skriganov constructs an ε-sample of size $O((1/\varepsilon)\log^{d-1}(1/\varepsilon)\ \mathrm{polylog}(\log(1/\varepsilon)))$. When P is a discrete point set of size n, ε-samples of size $O(((1/\varepsilon)\log(1/\varepsilon))^{2-2/(\nu+1)})$ exist for bounded VC-dimension ν [84], but there are not efficient constructions known for sets of this size. Alternatively, ε-samples of size $O((\nu/\varepsilon^2)\log(\nu/\varepsilon))$ can be constructed in time $O(n \cdot ((\nu^3/\varepsilon^2)\log(\nu/\varepsilon))^\nu)$. In this spirit, for $\mathcal{R}_2$ and a discrete point set of size n, Suri, Toth, and Zhou [112] construct an ε-sample of size $O((1/\varepsilon)\log(\varepsilon n)\log^4((1/\varepsilon)\log(\varepsilon n)))$ in the context of a streaming algorithm which can be analyzed to run in time $O(n((1/\varepsilon)\log^4(1/\varepsilon))^3)$.

1.1.2 Lebesgue and Combinatorial Discrepancy

Lebesgue discrepancy. The Lebesgue discrepancy is defined for an n-point set $Q \subset [0,1]^d$ relative to the volume of a unit cube $[0,1]^d$. [2] Given a range space $([0,1]^d, \mathcal{A})$ and a point set Q, the *Lebesgue discrepancy* is defined

$$D(Q, \mathcal{A}) = \sup_{R \in \mathcal{A}} |\bar{D}(Q, R)|, \quad \text{where} \quad \bar{D}(Q,R) = n \cdot |R \cap [0,1]^d| - |R \cap Q|.$$

Optimized over all n-point sets, define the *optimal Lebesgue discrepancy* of $([0,1]^d, \mathcal{A})$ as

$$\hat{D}(n, \mathcal{A}) = \inf_{Q \subset [0,1]^d, |Q| = n} D(Q, \mathcal{A}).$$

The study of Lebesgue discrepancy arguably began with the Van der Corput set C_n [114], which satisfies $D(C_n, \mathcal{R}_2) = O(\log n)$. This was generalized to higher dimensions by Hammersley [53] and Halton [52] so that $D(C_n, \mathcal{R}_d) = O(\log^{d-1} n)$. However, it was shown that many lattices also provide $O(\log n)$ discrepancy in the plane [82]. This is generalized to $O(\log^{d-1} n \log^{1+\tau} \log n)$ for $\tau > 0$ over $\mathcal{R}^d$ [108, 109, 22]. For a more in-depth history of the progression of these results we refer to the notes in Matoušek's book [82]. For application of these results in numerical integration see Niederreiter's book [91]. The results on lattices extend to homothets of any $G_k \in \mathcal{Q}_k$ for $O(\log n)$ discrepancy in the plane [108] and $O(\log^{d-1} n \log^{1+\tau} \log n)$

[2] Although not common in the literature, this definition can replace $[0,1]^d$ with an hyper-rectangle $[0, w_1] \times [0, w_2] \times \ldots \times [0, w_d]$.

discrepancy, for $\tau > 0$, in $\mathbb{R}^d$ [110], for some constant k. A wider set of geometric families which include half planes, right triangles, rectangles under all rotations, circles, and predefined convex shapes produce $\Omega(n^{1/4})$ discrepancy.

Lebesgue discrepancy describes an ε-sample of $([0,1]^d, \mathcal{A})$, where $\varepsilon = f(n) = \hat{D}(n, \mathcal{A})/n$. Thus we can construct an ε-sample for $([0,1]^d, \mathcal{A})$ of size $g_D(\varepsilon, \mathcal{A})$ as defined below. (Solve for n in $\varepsilon = \hat{D}(n, \mathcal{A})/n$).)

$$g_D(\varepsilon, \mathcal{A}) = \begin{cases} O((1/\varepsilon)\log^\tau(1/\varepsilon)) & \text{for } \hat{D}(n, \mathcal{A}) = O(\log^\tau n) \\ O((1/\varepsilon)^{1/(1-\tau)}) & \text{for } \hat{D}(n, \mathcal{A}) = O(n^\tau) \end{cases} \tag{1.1}$$

Combinatorial discrepancy. Given a range space $(P, \mathcal{A})$ where P is a finite point set and a coloring function $\chi : P \to \{-1, +1\}$ we say the *combinatorial discrepancy* of $(P, \mathcal{A})$ colored by χ is

$$\operatorname{disc}_\chi(P, \mathcal{A}) = \max_{R \in \mathcal{A}} \overline{\operatorname{disc}}_\chi(P \cap R) \quad \text{where}$$

$$\overline{\operatorname{disc}}_\chi(P) = \sum_{p \in P} \chi(p) = |\{p \in P \ : \ \chi(p) = +1\}| - |\{p \in P \ : \ \chi(p) = -1\}|.$$

Taking this over all colorings and all point sets of size n we say

$$\widehat{\operatorname{disc}}(n, \mathcal{A}) = \max_{\{P : |P| = n\}} \min_{\chi : P \to \{-1,+1\}} \operatorname{disc}_\chi(P, \mathcal{A}).$$

Results about combinatorial discrepancy are usually proved using the partial coloring method [20] or the Beck-Fiala theorem [24]. The partial coloring method usually yields lower discrepancy by some logarithmic factors, but is nonconstructive. Alternatively, the Beck-Fiala theorem actually constructs a low discrepancy coloring, but with a slightly weaker bound. The Beck-Fiala theorem states that for a family of ranges $\mathcal{A}$ and a point set P such that $\max_{p \in P} |\{A \in \mathcal{A} \ : \ p \in A\}| \leq t$, $\operatorname{disc}(P, \mathcal{A}) \leq 2t - 1$. So the discrepancy is only a constant factor larger than the largest number of sets any point is in.

Srinivasan [111] shows that $\widehat{\operatorname{disc}}(n, \mathcal{R}_2) = O(\log^{2.5} n)$, using the partial coloring method. An earlier result of Beck [19] showed $\widehat{\operatorname{disc}}(n, \mathcal{R}_2) = O(\log^4 n)$ using the Beck-Fiala theorem [24]. The construction in this approach reduces to $O(n)$ Gaussian eliminations on a matrix of constraints that is $O(n) \times O(n)$. Each Gaussian elimination step requires $O(n^3)$ time. Thus the coloring χ in the construction for $\widehat{\operatorname{disc}}(n, \mathcal{R}_2) = O(\log^4 n)$ can be found in $O(n^4)$ time. For more results on discrepancy see Beck and Chen's book [23].

Similar to Lebesgue discrepancy, the set $Q = \{q \in X \mid \chi(q) = +1\}$ generated from the coloring χ for combinatorial discrepancy $\widehat{\operatorname{disc}}(n, \mathcal{A})$ describes an ε-sample of $(X, \mathcal{A})$ where $\varepsilon = f(n) = \widehat{\operatorname{disc}}(n, \mathcal{A})/n$. Thus, given this value of ε, we can say that Q is an ε-sample for $(X, \mathcal{A})$ of size

$$g(\varepsilon, \mathcal{A}) = \begin{cases} O((1/\varepsilon)\log^\tau(1/\varepsilon)) & \text{for } \widehat{\operatorname{disc}}(n, \mathcal{A}) = O(\log^\tau n) \\ O((1/\varepsilon)^{1/(1-\tau)}) & \text{for } \widehat{\operatorname{disc}}(n, \mathcal{A}) = O(n^\tau). \end{cases} \tag{1.2}$$

In the next section we generalize this result.

1.2 Deterministic Construction of ε-Samples for Discrete Point Sets

Lemma 1.1. $\widehat{\mathrm{disc}}(n, \mathcal{Q}_k) = O(\log^{2k} n)$ *for points in* $\mathbb{R}^d$ *and the coloring that generates this discrepancy can be constructed in* $O(n^4)$ *time, for* k *constant.*

The proof combines techniques from Beck [19] and Matoušek [83].

Proof. Given a class $\mathcal{Q}_k$, each potential face is defined by a normal vector from $\{\beta_1, \ldots, \beta_k\}$. For $j \in [1, k]$ project all points along β_j. Let a *canonical interval* be of the form $\left[\frac{t}{2^q}, \frac{t+1}{2^q}\right)$ for integers $q \in [1, \log n]$ and $t \in [0, 2^q)$. For each direction β_j choose a value $q \in [1, \log n]$ creating 2^q canonical intervals induced by the ordering along β_j. Let the intersection of any k of these canonical intervals along a fixed β_j be a *canonical subset*. Since there are $\log n$ choices for the values of q for each of the k directions, it follows that each point is in at most $(\log n)^k$ canonical subsets. Using the Beck-Fiala theorem, we can create a coloring for X so that no canonical subset has discrepancy more than $O(\log^k n)$.

Each range $R \in \mathcal{Q}_k$ is formed by at most $O(\log^k n)$ canonical subsets. For each ordering by β_i, the interval in this ordering induced by R can be described by $O(\log n)$ canonical intervals. Thus the entire range R can be decomposed into $O(\log^k n)$ canonical subsets, each with at most $O(\log^k n)$ discrepancy.

Applying the Beck-Fiala construction of size n, this coloring requires $O(n^4)$ time to construct. $\square$

Corollary 1.1. $\widehat{\mathrm{disc}}(n, \mathcal{R}_d) = O(\log^{2d} n)$ *and the coloring that generates this discrepancy can be constructed in* $O(n^4)$ *time, for* d *constant.*

A better nonconstructive bound exists due to Matoušek [83], using the partial coloring method. For polygons in $\mathbb{R}^2$ $\widehat{\mathrm{disc}}(n, \mathcal{Q}_k) = O(k \log^{2.5} n \sqrt{\log(k + \log n)})$, and for polytopes in $\mathbb{R}^d$ $\widehat{\mathrm{disc}}(n, \mathcal{Q}_k) = O(k^{1.5\lfloor d/2 \rfloor} \log^{d+1/2} n \sqrt{\log(k + \log n)})$.

Next we will describe how to iteratively apply this process efficiently to achieve these bounds for any value of ε.

1.2.1 From Combinatorial Discrepancy to ε-Samples

We generalize the framework of Chazelle and Matoušek [32] describing an algorithm for creating an ε-sample of a range space $(P, \mathcal{A})$. Consider any range space $(P, \mathcal{A})$, with $|P| = n$, for which there is an algorithm to generate a coloring χ that yields the combinatorial discrepancy $\mathrm{disc}_\chi(P, \mathcal{A})$ and can be constructed in time $O(n^w \cdot l(n))$ where $l(n) = o(n)$. For simplicity, we refer to the combinatorial discrepancy we can construct $\mathrm{disc}_\chi(P, \mathcal{A})$ as $\widehat{\mathrm{disc}}(n, \mathcal{A})$ to emphasize the size of the ground set, and we use equation (1.2) to describe $g(\varepsilon, \mathcal{A})$, the size of the ε-sample it corresponds to.

The values $\widehat{\text{disc}}(n, \mathcal{A})$, w, and $l(n)$ are dependent on the range space $(P, \mathcal{A})$ (i.e., see Lemma 1.1), but not necessarily its VC-dimension as in [32]. As used above, let $f(n) = \widehat{\text{disc}}(n, \mathcal{A})/n$ be the value of ε in the ε-sample generated by a single coloring of a set of size n — the relative error. We require that, $f(2n) \leq (1 - \delta)f(n)$, for constant $0 < \delta \leq 1$; thus it is a geometrically decreasing function.

The algorithm will compress a set P of size n to a set Q of size $O(g(\varepsilon, \mathcal{A}))$ such that Q is an ε-sample of $(P, \mathcal{A})$ by recursively creating a low discrepancy coloring. We note that an ε-sample of an ε'-approximation is an $(\varepsilon + \varepsilon')$-approximation of the original set.

We start by dividing P into sets of size $O(g(\varepsilon, \mathcal{A}))$,[3] here ε is a parameter. The algorithm proceeds in two stages. The first stage alternates between merging pairs of sets and halving sets by discarding points colored $\chi(p) = -1$ by the combinatorial discrepancy method described above. The exception is after every $w + 2$ halving steps, we then skip one halving step. The second stage takes the one remaining set and repeatedly halves it until the error $f(|Q|)$ incurred in the remaining set Q exceeds $\varepsilon/(2 + 2\delta)$. This results in a set of size $O(g(\varepsilon, \mathcal{A}))$.

Algorithm 1.2.1 Creates an ε-sample for $(P, \mathcal{A})$ of size $O(g(\varepsilon, \mathcal{A}))$.

```
1: Divide $P$ into sets $\{P_0, P_1, P_2, \ldots\}$ each of size $4(w + 2)g(\varepsilon, \mathcal{A})$. [2]
2: **repeat** {*Stage 1*}
3:   **for** $w + 2$ steps **do** {or stop if only one set is left}
4:     MERGE: Pair sets arbitrarily (i.e. $P_i$ and $P_j$) and merge them into a single
         set (i.e. $P_i := P_i \cup P_j$).
5:     HALVE: Halve each set $P_i$ using the coloring $\chi$ from disc$(P_i, \mathcal{A})$ (i.e. $P_i =$
         $\{p \in P_i \mid \chi(p) = +1\}$).
6:   MERGE: Pair sets arbitrarily and merge each pair into a single set.
7: **until** only one set, $Q$, is left
8: **repeat** {*Stage 2*}
9:   HALVE: Halve $Q$ using the coloring $\chi$ from disc$(Q, \mathcal{A})$.
10: **until** $f(|P|) \geq \varepsilon/(2 + 2\delta)$
```

Theorem 1.1. *For a finite range space $(P, \mathcal{A})$ with $|P| = n$ and an algorithm to construct a coloring $\chi : P \to \{-1, +1\}$ such that*

- *the set $\{p \in P \ : \ \chi(p) = +1\}$ is an α-approximation of $(P, \mathcal{A})$ of size $g(\alpha, \mathcal{A})$ with $\alpha = disc_\chi(P, \mathcal{A})/n$ (see equation (1.2)).*
- *χ can be constructed in $O(n^w \cdot l(n))$ time where $l(n) = o(n)$.*

then Algorithm 1.2.1 constructs an ε-sample for $(P, \mathcal{A})$ of size $O(g(\varepsilon, \mathcal{A}))$ in time $O(w^{w-1} n \cdot g(\varepsilon, \mathcal{A})^{w-1} \cdot l(g(\varepsilon, \mathcal{A})) + g(\varepsilon, \mathcal{A}))$.

[3] If the sets do not divide equally, artificially increase the size of the sets when necessary. These points can be removed later.

Proof. Let $2^j = 4(w+2)g(\varepsilon, \mathcal{A})$, for an integer j, be the size of each set in the initial dividing stage (adjusting by a constant if $\delta \leq 1/4$). Each round of Stage 1 performs $w+3$ MERGE steps and $w+2$ HALVE steps on sets of the same size and each subsequent round deals with sets twice as large. The union of all the sets is an γ-sample of $(P, \mathcal{A})$ (to start $\gamma = 0$) and γ only increases in the HALVE steps. The ith round increases γ by $f(2^{j-1+i})$ per HALVE step. Since $f(n)$ decrease geometrically as n increases, the size of γ at the end of the first stage is asymptotically bounded by the increase in the first round. Hence, after Stage 1 $\gamma \leq 2(w+2)f(4(w+2)g(\varepsilon, \mathcal{A})) \leq \varepsilon/2$. Stage 2 culminates the step before $f(|P|) \geq \varepsilon/(2+2\delta)$. Thus the final HALVE step creates an $(\varepsilon\delta/(2+2\delta))$-approximation and the entire second stage creates an $\varepsilon/2$-approximation, hence overall Algorithm 1.2.1 creates an ε-sample. The relative error caused by each HALVE step in stage 2 is equivalent to a HALVE step in a single round of stage 1.

The running time is also dominated by Stage 1. Each HALVE step of a set of size 2^j takes $O((2^j)^w l(2^j))$ time and runs on $n/2^j$ sets. In between each HALVE step within a round, the number of sets is divided by two, so the running time is asymptotically dominated by the first HALVE step of each round. The next round has sets of size 2^{j+1}, but only $n/2^{j+w+2}$ of them, so the runtime is at most 1/2 that of the first HALVE step. Thus the running time of a round is less than half of that of the previous one. Since $2^j = O(wg(\varepsilon, \mathcal{A}))$ the running time of the HALVE step, and hence the first stage is bounded by $O(n \cdot (w \cdot g(\varepsilon, \mathcal{A}))^{w-1} \cdot l(g(\varepsilon, \mathcal{A})) + g(\varepsilon, \mathcal{A}))$. Each HALVE step in the second stage corresponds to a single HALVE step per round in the first stage, and does not affect the asymptotics. $\square$

We can invoke Theorem 1.1 along with Lemma 1.1 and Corollary 1.1 to compute χ in $O(n^4)$ time (notice that $w = 4$ and $l(\cdot)$ is constant), so $g(\varepsilon, \mathcal{Q}_k) = O((1/\varepsilon)\log^{2k}(1/\varepsilon))$ and $g(\varepsilon, \mathcal{R}_d) = O((1/\varepsilon)\log^{2d}(1/\varepsilon))$. We obtain the following important corollaries.

Corollary 1.2. *For a set of size n and over the ranges $\mathcal{Q}_k$ an ε-sample of size $O((1/\varepsilon)\log^{2k}(1/\varepsilon))$ can be constructed in time $O((n/\varepsilon^3)\log^{6k}(1/\varepsilon))$.*

Corollary 1.3. *For a set of size n and over the ranges $\mathcal{R}_d$ an ε-sample of size $O((1/\varepsilon)\log^{2d}(1/\varepsilon))$ can be constructed in time $O((n/\varepsilon^3)\log^{6d}(1/\varepsilon))$.*

Weighted case. These results can be extended to the case where each point $p \in P$ is given a weight $\mu_0(p)$ and for any $Q \subseteq P$ let $\mu(Q) = \sum_{q \in Q} \mu_0(q)$. Now an ε-sample of a range space $(P, \mathcal{A})$ is a set $Q \subseteq P$ and a weighting $\mu_0 : P \to \mathbb{R}$ such that

$$\max_{R \in \mathcal{A}} \left| \frac{\mu(Q \cap R)}{\mu(Q)} - \frac{\mu(P \cap R)}{\mu(P)} \right| \leq \varepsilon.$$

The weights on Q may differ from those on P. A result from Matoušek [80], invoking the unweighted algorithm several times at a geometrically decreasing cost, creates a weighted ε-sample of the same asymptotic size and with the same asymptotic runtime as for an unweighted algorithm. This extension is important when we combine ε-samples representing regions of different total measure. For this case we weight each point relative to the measure it represents.

1.3 Sampling from Polygonal Ground Sets

We will prove a general theorem for deterministically constructing small ε-samples for polygonal ground sets which will have direct consequences on polygonal terrains. A key observation of Matoušek [80] is that the union of ε-samples of disjoint ground sets forms an ε-sample of the union of the ground sets. Thus for any polygonal ground set P we first divide it into pieces for which we can create ε-samples. Then we merge all of these point sets into an ε-sample for the entire domain. Finally, we use Theorem 1.1 to reduce the sample size.

Instead of restricting ourselves to ground sets which we can divide into cubes of the form $[0,1]^d$, thus allowing the use of Lebesgue discrepancy results, we first expand on a result about lattices and polygons.

Lattices and polygons. A *lattice* Λ in $\mathbb{R}^d$ is an infinite set of points defined such that for d vectors $V_\Lambda = \{v_1, \ldots, v_d\}$ that form a basis, for any point $p \in \Lambda$, $p + v_h$ and $p - v_h$ are also in Λ for any $h \in [1,d]$. We also specify an origin point $v_0 \in \Lambda$. Λ is *irrational* with respect to any polytope in $\mathcal{G}_\beta$ if for all $\beta_i \in \beta$, for all $v_h \in V_\Lambda$, and for all $j, l \leq d$, the fraction $\beta_{i,j}/v_{h,l}$ is irrational. (Note that $\beta_{i,j}$ (resp. $v_{h,l}$) represents the jth (resp. lth) element of the vector β_i (resp. v_h).) Lattices with Λ irrational (relative to the face normals) generate low discrepancy sets.

Theorem 1.2. *Let $\beta, \beta' \subset \mathbb{S}^{d-1}$ be sets of k and h directions, respectively. Let $G_h \in \mathcal{G}_{\beta'}$ be a fixed convex polytope. Let $\mathcal{Q}_k$ be the set of ranges defined by directions β. We can choose a lattice Λ such that $\Lambda \cap G_h$ is an ε-sample of $(G_h, \mathcal{Q}_k)$ of size $O(((k+h)/\varepsilon)\log^{d-1}(1/\varepsilon)\ \mathrm{polylog}(\log(1/\varepsilon)))$.*

Proof. Consider polytope $sG_h \in \mathcal{G}_{\beta'}$ and lattice Λ, where the uniform scaling factor s is treated as an asymptotic quantity, and let $\mu(G_h) = O(1/n)$ (i.e. the Lebesgue-measure in $\mathbb{R}^d$ of polytope G_h is $\Theta(1/n)$) for some integer n. For notational simplicity let $M_\Lambda = \Lambda \cap [0,1]^d$, and assume $sG_h + t \subset [0,1]^d$. Skriganov's Theorem 6.1 in [110] claims

$$\max_{t \in \mathbb{R}^d} \bar{D}(M_\Lambda, sG_h + t \cap M_\Lambda) = O\left(s^{d-1}\rho^{-\theta} + \sum_f S_f(\Lambda, \rho)\right)$$

where

$$S_f(\Lambda, \rho) = O(\log^{d-1}\rho \log^{1+\tau}\log\rho)$$

for $\tau > 0$, as long as Λ is irrational with respect to the normal of the face f of G_h and infinite otherwise, where $\theta \in (0,1)$ and ρ can be arbitrarily large. Note that this is a simplified form yielded by invoking Theorem 3.2 and Theorem 4.5 from [110]. By setting $\rho^\theta = s^{d-1}$,

$$\max_{t \in \mathbb{R}^d} \bar{D}(M_\Lambda, sG_h + t \cap M_\Lambda) = O(h \log^{d-1} s \log^{1+\tau} \log s). \tag{1.3}$$

Now by noting that as s grows, the number of lattice points in sG_h grows by a factor of s^d, and we can set $s = n^{1/d}$ so (1.3) implies that $\bar{D}(M_\Lambda, sG_h \cap M_\Lambda) = O(h \log^{d-1} n \log^{1+\tau} \log n)$ for $|M_\Lambda| = n$.

The discrepancy is a sum over the set of h terms, one for each face f, each of which is small as long as Λ is irrational with respect to f's normal β_f. Hence this lattice gives low discrepancy for any polytope in the analogous family $\mathcal{G}_\beta$ such that Λ is irrational with respect to $\mathcal{G}_\beta$. Finally we realize that any subset $G_h \cap G_k$ for $G_h \in \mathcal{G}_{\beta'}$ and $G_k \in \mathcal{G}_\beta$ is a polytope defined by normals from $\beta' \cup \beta$. Let $\mathcal{Q}_{k+k'}$ be the ranges defined by directions $\beta \cup \beta'$, and then refer to $g_D(\varepsilon, \mathcal{Q}_{k+k'})$ in (1.1) to bound the size of the ε-sample from the given Lebesgue discrepancy. $\square$

Remark. Skriganov's result [110] is proved under the *whole space* model where the lattice is infinite (tG_h is not confined to $[0,1]^d$), and the relevant error is the difference between the Lebesgue measure of tG_h versus the cardinality $|tG_h \cap \Lambda|$, where each $p \in \Lambda$ represents 1 unit of measure. Skriganov's main results is summarized in equation (1.3) and only pertains to a fixed polytope G_h instead of, more generally, a family of polytopes $\mathcal{G}_\beta$, as shown in Theorem 1.2.

Samples for polygonal terrains. Combining the above results and weighted extension of Theorem 1.1 implies the following results.

Theorem 1.3. *We can create a weighted ε-sample of size $O((k+k')(1/\varepsilon) \cdot \log^{2k}(1/\varepsilon))$ of $(P, \mathcal{Q}_k)$ in time $O((k+k')(n/\varepsilon^4)$ polylog$(1/\varepsilon))$ for any d-dimensional ground set P which can be decomposed into n d-dimensional convex k'-oriented polytopes.*

Proof. We divide the domain into n k'-oriented polytopes and then approximate each polytope $G_{k'}$ with a point set $\Lambda \cap G_{k'}$ using Theorem 1.2. We observe that the union of these point sets is a weighted ε-sample of $(P, \mathcal{Q}_k)$, but is quite large. Using the weighted extension of Theorem 1.1 we can reduce the point sets to the size and in the time stated. $\square$

This has applications to terrain ground sets P defined with a piecewise-linear base B and height function $h : B \to \mathbb{R}$. We decompose the terrain so that each linear piece of h describes one 3-dimensional polytope, then apply Theorem 1.3 to get the following result.

Corollary 1.4. *For terrain ground set P with piecewise-linear base B and height function $h : B \to \mathbb{R}$ with n linear pieces, we construct a weighted ε-sample of $(P, \mathcal{Q}_k)$ of size $O(k(1/\varepsilon) \log^4(1/\varepsilon))$ in time $O(k(n/\varepsilon^4)$ polylog$(1/\varepsilon))$.*

1.3.1 ε-Samples for Distributions and Shapes with Bounded VC-Dimension.

In this subsection we extend the results described in this section to ranges $\mathcal{A}$ so that $(\mathbb{R}^d, \mathcal{A})$ has bounded VC-dimension.

The general scheme to create an ε-sample for $(P, \mathcal{A})$, where $P \in \mathbb{R}^d$ is a polygonal shape, is to use a lattice Λ of points. We first create a discrete $(\varepsilon/2)$-sample $M = \Lambda \cap P$ of $(P, \mathcal{A})$ and then create an $(\varepsilon/2)$-sample Q of $(M, \mathcal{A})$ using standard techniques [32] or Corollary 1.2. Then Q is an ε-sample of $(P, \mathcal{A})$. For a shape P with m $(d-1)$-faces on its boundary, any subset $A' \subset \mathbb{R}^d$ that is described by a subset from $(P, \mathcal{A})$ is an intersection $A' = A \cap P$ for some $A \in \mathcal{A}$. Since P has m $(d-1)$-dimensional faces, when $\nu_{\mathcal{A}}$ is the VC-dimension of $(\mathbb{R}^d, \mathcal{A})$, we can bound the VC-dimension of $(P, \mathcal{A})$ as $\nu = O((m + \nu_{\mathcal{A}}) \log(m + \nu_{\mathcal{A}}))$. Finally the set $M = P \cap \Lambda$ is determined by choosing an arbitrary initial origin point in Λ and then uniformly scaling all vectors $\{v_1, \ldots, v_d\}$ until $|M| = \Theta((\nu/\varepsilon^2)\log(\nu/\varepsilon))$ [82].

It follows that we can create such an ε-sample of size $|M|$ in time $O(|M| m \log |M|)$ by starting with a scaling of the lattice so a constant number of points are in S and then doubling the scale until we get to within a factor of d of $|M|$. If there are n points inside S, it takes $O(nm)$ time to count them.

Lemma 1.2. *For a polygonal shape $P \subset \mathbb{R}^d$ with m facets, we can construct an ε-sample for $(P, \mathcal{A})$ of size $O((\nu/\varepsilon^2)\log(\nu/\varepsilon))$ in time $O(m(\nu/\varepsilon^2)\log^2(\nu/\varepsilon))$, where $(P, \mathcal{A})$ has VC-dimension $\nu_{\mathcal{A}}$ and $\nu = O((\nu_{\mathcal{A}} + m)\log(\nu_{\mathcal{A}} + m))$.*

Remark. An important part of the above construction is the arbitrary choice of the origin points of the lattice Λ. This allows us to arbitrarily shift the lattice defining M and thus the set Q. In Section 3.5 we need to construct n ε-samples $\{Q_1, \ldots, Q_n\}$ for n range spaces $\{(P_1, \mathcal{A}), \ldots, (P_n, \mathcal{A})\}$. In Algorithm 3.5.2 we examine sets of $\nu_{\mathcal{A}}$ points, each from separate ε-samples that define a minimal shape $A \in \mathcal{A}$. It is important that we do not have two such (possibly not disjoint) sets of $\nu_{\mathcal{A}}$ points that define the same minimal shape $A \in \mathcal{A}$. (Note, this does not include cases where say two points are antipodal on a disk and any other point in the disk added to a set of $\nu_{\mathcal{A}} = 3$ points forms such a set; it refers to cases where say four points lie (degenerately) on the boundary of a disc.) We can guarantee this by enforcing a property on all pairs of origin points p and q for $(P_i, \mathcal{A})$ and $(P_j, \mathcal{A})$. For the purpose of construction, it is easiest to consider only the lth coordinates p_l and q_l for any pair of origin points or lattice vectors (where the same lattice vectors V_Λ are used for each lattice). We enforce a specific property on every such pair p_l and q_l, for all l and all distributions and lattice vectors.

First, consider the case where $\mathcal{A} = \mathcal{R}_d$ describes axis-aligned bounding boxes. It is easy to see that if for all pairs p_l and q_l that $(p_l - q_l)$ is irrational, then we cannot have $> 2d$ points on the boundary of an axis-aligned bounding box, hence the desired property is satisfied.

Now consider the more complicated case where $\mathcal{A} = \mathcal{B}$ describes smallest enclosing balls. There is a polynomial of degree 2 that describes the boundary of the ball, so we can enforce that for all pairs p_l and q_l that $(p_l - q_l)$ is of the form $c_1(r_{p_l})^{1/3} + c_2(r_{q_l})^{1/3}$ where c_1 and c_2 are rational coefficients and r_{p_l} and r_{q_l} are distinct integers that are not multiple of cubes. Now if $\nu = d + 1$ such points satisfy (and in fact define) the equation of the boundary of a ball, then no $(d + 2)$th point which has this property with respect to the first $d + 1$ can also satisfy this equation.

More generally, if $\mathcal{A}$ can be described with a polynomial of degree p with ν variables, then enforce that every pair of coordinates are the sum of $(p + 1)$-roots. This ensures that no $\nu+1$ points can satisfy the equation, and the undesired situation cannot occur.

1.4 Sampling from Smooth Ground Sets

We can create an ε-sample for a smooth Lebesgue-measureable ground set P (one which cannot be decomposed into polytopes) in a three stage process. The first stage approximates any ground set with a set of polytopes. The second approximates each polytope with a point set. The third merges all point sets and uses Theorem 1.1 to reduce their size.

This section mainly focuses on the first stage, however, we also offer an improvement for the second stage in a relevant special case. More formally, we can approximate a non-polygonal ground set P with a set of disjoint polygons Q such that Q has properties of an ε-sample.

Lemma 1.3. *If $\mu(P \setminus Q) \leq (\varepsilon/2)\mu(P)$ and $Q \subseteq P$ then*

$$\max_{R\in\mathcal{A}} \left| \frac{\mu(R \cap Q)}{\mu(Q)} - \frac{\mu(R \cap P)}{\mu(P)} \right| \leq \varepsilon.$$

Proof. No range $R \in \mathcal{A}$ can have

$$\left| \frac{\mu(R \cap Q)}{\mu(Q)} - \frac{\mu(R \cap P)}{\mu(P)} \right| > \varepsilon$$

because if $\mu(P) \geq \mu(Q)$ (w.l.o.g.), then $\mu(R\cap P) - (\mu(P)/\mu(Q))\mu(R\cap Q) \leq \varepsilon\mu(P)$ and $\mu(R\cap Q)(\mu(P)/\mu(Q)) - \mu(R\cap P) \leq \varepsilon\mu(P)$. The first part follows from $\mu(P)/\mu(Q) \geq 1$ and is loose by a factor of 2. For the second part we can argue

$$\begin{aligned}
\mu(R \cap Q)\frac{\mu(P)}{\mu(Q)} - \mu(R \cap P) &\leq \mu(R \cap Q)\frac{1}{1-\varepsilon/2} - \mu(R \cap P) \\
&\leq \mu(R \cap P)\frac{1}{1-\varepsilon/2} - \mu(R \cap P) \\
&= \frac{\varepsilon/2}{1-\varepsilon/2}\mu(R \cap P) \\
&\leq \varepsilon\mu(R \cap P) \leq \varepsilon\mu(P).
\end{aligned}$$

□

We say a ground set $P \subset \mathbb{R}^d$ is *polygonal approximable* if there exists a polygonal shape $Q \subset P$ with m facets such that $\mu(P \setminus Q) \leq \varepsilon\mu(P)$ for any $\varepsilon > 0$. Usually, m is dependent on ε.

For terrain ground set P defined with a base B and a height function $h : B \to \mathbb{R}$, if B is polygonal we can decompose it into polygonal pieces, otherwise we can approximate it with constant-size polygonal pieces according to Lemma 1.3. Then, similarly, if h is polygonal we can approximate the components invoking Corollary 1.4; however, if it is smooth, then we can approximate each piece according to Lemma 1.3.

Section 1.4.1 improves on Theorem 1.2 for the second stage and gives a more efficient way to create an ε-sample for $(P, \mathcal{R}_d \times \mathbb{R})$ of a terrain when B is a rectangle and h is linear. Ranges from the family $\mathcal{R}_d \times \mathbb{R}$ are generalized hyper-cylinders in $d+1$ dimensions where the first d dimensions are described by an axis-parallel rectangle and the $(d+1)$st dimension is unbounded. Section 1.4.2 focuses on the first stage and uses this improvement as a base case in a recursive algorithm (akin to a fair split tree) for creating an ε-sample for $(P, \mathcal{R}_d \times \mathbb{R})$ when B is rectangular and h is smooth.

1.4.1 Stretching the Van der Corput Set

The Van der Corput set [114] is a point set $Q_n = \{q_0, \ldots, q_{n-1}\}$ in the unit square defined for $q_i = (i/n, b(i))$ where $b(i)$ is the bit reversal sequence. For simplicity we assume n is a power of 2. The function $b(i)$ writes i in binary, then reverses the ordering of the bits, and places them after the decimal point to create a number in $[0, 1)$. For instance for $n = 16$, $i = 13 = 1101$ in binary and $b(13) = 0.1011 = 11/16$. Formally, if $i = \sum_{i=0}^{\log n} a_i 2^i$ then $b(i) = \sum_{i=0}^{\log n} (a_i/2^{i+1})$.

Halton [52] showed that the Van der Corput set Q_n satisfies $D(Q_n, \mathcal{R}_2) = O(\log n)$. We can extend this to approximate any rectangular domain. For a rectangle $[0, w] \times [0, l]$ (w.l.o.g.) we can use the set $Q_{n,w,l}$ where $q_i = (w \cdot i/n, l \cdot b(i))$ and a version of the Lebesgue discrepancy over a stretched domain is still $O(\log n)$.

We can stretch the Van der Corput set to approximate a rectangle $r = [0, w] \times [0, l]$ with a weighting by an always positive linear height function $h(x, y) = \alpha x + \beta y + \gamma$. Let $\Delta(w, \alpha, \gamma, i)$ be defined such that the following condition is satisfied

$$\int_0^{\Delta(w,\alpha,\gamma,i)} (\alpha x + \gamma) dx = \frac{i}{n} \int_0^w (\alpha x + \gamma) dx.$$

Note that we can solve for Δ explicitly and because h is linear it can simultaneously be defined for the x and y direction. Now define the *stretched Van der Corput set* $S_{n,w,l,h} = \{s_0, \ldots, s_n\}$ for $s_i = (\Delta(w, \alpha, \gamma, i), \Delta(l, \beta, \gamma, b(i) \cdot n))$.

Theorem 1.4. *For the stretched Van der Corput set $S_{n,w,l,h}$, $D(S_{n,w,l,h}, \mathcal{R}_2) = O(\log n)$ over the domain $[0, w] \times [0, l]$ with $h : [0, w] \times [0, l] \to \mathbb{R}^+$ a linear weighting function.*

The proof follows the proof in Matoušek [82] for proving logarithmic discrepancy for the standard Van der Corput set in the unit square.

Proof. Let a *canonical interval* be of the form $[\Delta(l,\beta,\gamma,k)/2^q,\ \Delta(l,\beta,\gamma,k+1)/2^q)$ for integers $q \in [1,n]$ and $k \in [0,2^q)$. Let any rectangle $r = [0,a) \times I$ where I is canonical and $a \in (0,1]$ be called a *canonical rectangle.*

Claim 1.1. *For any canonical rectangle r, $\bar{D}(S_{n,w,l,h}, r) \leq 1$.*

Proof. Like in the Van der Corput set, every subinterval of r such that $h(r) = 1/n$ has exactly 1 point. Let $I = [\Delta(l,\beta,\gamma,k)/2^q,\ \Delta(l,\beta,\gamma,k+1)/2^q)$. Thus each rectangle $r_j = [\Delta(l,\beta,\gamma,j2^q/n), \Delta(l,\beta,\gamma,(j+1)2^q/n)) \times I$ contains a single point from $S_{n,w,l,h}$ and $h(r_j) = 1/n$, where $h(r) = \int_r h(p)dp$.

So the only part which generates any discrepancy is the canonical rectangle r_j which contains the segment $a \times I$. But since $|S_{n,w,l,h} \cap r_j \cap r| \leq 1$ and $h(r_j \cap r) \leq 1/n$, the claim is proved. □

Let $\mathcal{C}_d$ be the family of ranges defined by d-dimensional rectangles with the lower left corner at the origin. Let $C_{(x,y)} \in \mathcal{C}_2$ be the corner rectangle with upper right corner at (x,y).

Claim 1.2. *Any corner rectangle $C_{(x,y)}$ can be expressed as the disjoint union of at most $O(\log n)$ canonical rectangles plus a rectangle M with $|\bar{D}(S_{n,w,l,h}, M)| \leq 1$.*

Proof. Starting with the largest canonical rectangle $r_0 = [0,a) \times I$ within $C_{(x,y)}$ such that $I = [\Delta(l,\beta,\gamma,0)/2^q,\ \Delta(l,\beta,\gamma,1)/2^q)$ for the smallest value possible of q, keep including the next largest disjoint canonical rectangle within $C_{(x,y)}$. Each consecutive one must increase q by at least 1. Thus there can be at most $O(\log n)$ of these.

The left over rectangle $M = [m_x, x] \times [m_y, y]$, must be small enough such that $\int_0^w \int_{m_y}^y h(p,q)\ dqdp < 1/n$, thus it can contain at most 1 point and $\bar{D}(S_{n,w,l,h}, M) \leq 1$. □

It follows from Claim 1.1 and Claim 1.2 that $\text{disc}(S, \mathcal{C}_2) = O(\log n)$. We conclude by using the classic result [82] that $D(S,\mathcal{C}_2) \leq D(S,\mathcal{R}_2) \leq 4D(S,\mathcal{C}_2)$ for any point set S. □

This improves on the discrepancy for this problem attained by using Theorem 1.2 by a factor of $\log(1/\varepsilon)$.

Corollary 1.5. *A stretched Van der Corput set $S_{n,w,l,h}$ forms an ε-sample of $(P,\mathcal{R}_2)$ of size $n = O((1/\varepsilon)\log(1/\varepsilon))$ for P defined by a rectangle $[0,w] \times [0,l]$ with a linear height function h.*

Remark. This extends to higher dimensions. A stretched b-ary Van der Corput set [82] forms an ε-sample of $(P,\mathcal{R}_d)$ of size $O((1/\varepsilon)\log^{d-1}(1/\varepsilon))$ for P defined by $\times_{i=1}^d [0,w_i]$ with a linear height function.

1.4.2 Approximating Smooth Terrains

Given a terrain ground set P where $B \subset \mathbb{R}^2$ is rectangular and $h : B \to \mathbb{R}^+$ is a C^2-continuous height function we can construct an ε-sample based on a parameter ε and properties z_P^-, d_P, and λ_P. Let $z_P^- = \min_{p \in B} h(p)$. Let $d_P = \max_{p,q \in B} ||p - q||$ be the diameter of P. Let λ_P be the largest eigenvalue of H_h where

$$H_h = \begin{bmatrix} \frac{d^2h}{dx^2} & \frac{d^2h}{dxdy} \\ \frac{d^2h}{dydx} & \frac{d^2h}{dy^2} \end{bmatrix}$$

is the Hessian of h.

We first create a set of linear functions to approximate h with a recursive algorithm. If the entire domain cannot be approximated with a single linear function, then we split the domain by its longest direction (either x or y direction) evenly. This decreases d_P by a factor of $1/\sqrt{2}$ each time. We recur on each subset domain.

Lemma 1.4. *For a ground set P with rectangular base $B \subset \mathbb{R}^2$ and with a C^2-continuous height function $h : B \to \mathbb{R}$ we can approximate h with $O((\lambda_P d_P^2 / z_P^- \varepsilon))$ linear pieces h_ε so that for all $p \in B$ $h_\varepsilon(p) \leq h(p) \leq h_\varepsilon(p) + \varepsilon$.*

Proof. First we appeal to Lemma 4.1 which says that the error of a first order linear approximation at a distance d is bounded by $\lambda_P d^2$. Thus we take the tangent at the point in the middle of the range and this linear (first order) approximation has error bounded by $\lambda_P (d_P/2)^2 = \lambda_P d_P^2/4$. The height of the linear approximation is lowered by $\lambda_P d_P^2/4$ from the tangent point to ensure it describes a subset of P. Thus, as long as the upper bound on the error $\lambda_P d_P^2/2$ is less than $z_P^- \varepsilon$ then the lemma holds. The ratio $(\lambda_P d_P^2 / 2 z_P^- \varepsilon)$ is halved every time the domain is split until it is less than 1. Thus it has $O(\lambda_P d_P^2 / z_P^- \varepsilon)$ base cases. □

After running this decomposition scheme so that each linear piece L has error $\varepsilon/2$, we invoke Corollary 1.5 to create an $(\varepsilon/2)$-approximation point set of size $O((1/\varepsilon)\log(1/\varepsilon))$ for each $(L, \mathcal{R}_2 \times \mathbb{R})$. The union creates a weighted ε-sample of $(P, \mathcal{R}_2 \times \mathbb{R})$, but it is quite large. We can then reduce the size according to Corollary 1.3 to achieve the following result.

Theorem 1.5. *For a ground set P with rectangular base $B \subset \mathbb{R}^2$ and with a C^2-continuous height function $h : B \to \mathbb{R}$ we can deterministically create a weighted ε-sample of $(P, \mathcal{R}_2 \times \mathbb{R})$ of size $O\left(\left(\lambda_P d_P^2 / z_P^- \varepsilon\right)((1/\varepsilon)\log(1/\varepsilon))\right)$. We reduce the size to $O((1/\varepsilon)\log^4(1/\varepsilon))$ in time $O\left(\left(\lambda_P d_P^2 / z_P^-\right)(1/\varepsilon^5)\log^{13}(1/\varepsilon)\right)$.*

This generalizes in a straightforward way for $B \in \mathbb{R}^d$. Similar results are possible when B is not rectangular or when B is not even piecewise-linear. The techniques of Section 1.3 are necessary if $\mathcal{Q}_k$ is used instead of $\mathcal{R}_2$, and are slower by a factor $O(1/\varepsilon)$.

1.4.3 ε-Samples for a Normal Distribution

A *normal distribution*, often referred to as a Gaussian distribution, is a family of continuous distributions often used to model error. The *central limit theorem* highlights its power by showing that the sum of independent and identically distributed distributions with bounded variance converge to a normal distribution as the set size grows. A normal distribution $\mathcal{N}(m, \sigma^2)$ is defined by two parameters, a *mean* m marking the center of the distribution and a *variance* σ^2 scaling the spread of the distribution. Specifically, we can analyze a random variable X distributed according to a normal distribution, denoted $X \sim \mathcal{N}(m, \sigma^2)$. We then say that

$$\varphi_{m,\sigma^2}(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-m)^2}{2\sigma^2}}$$

describes the probability that a point X drawn randomly from a normal distribution $\mathcal{N}(m, \sigma^2)$ is located at $x \in \mathbb{R}$. Since it is a distribution, then $\int_{x\in\mathbb{R}} \varphi_{m,\sigma^2}(x) = 1$. A *standard normal distribution* $\mathcal{N}(0, 1)$ has mean 0 and variance 1. As the variance changes, the normal distribution is stretched symmetrically and proportional to σ so the integral is still 1. The inflection points of the curve describing the height of the distribution are at the points $m - \sigma$ and $m + \sigma$.

A *multivariate normal distribution* is a higher dimensional extension to the normal distribution. A d-dimensional random variable $X = [X_1, \ldots, X_d]^T$ is drawn from a multivariate normal distribution if for every linear combination $Y = a_1X_1 + \ldots + a_dX_d$ (defined by any set of d scalar values a_i) is normally distributed. Thus for a d-dimensional random variable X defined over the domain $\mathbb{R}^d$, any projection of X to a one dimensional subspace of $\mathbb{R}^d$ is normally distributed.

We now discuss how to create ε-samples for $(P, \mathcal{R}_2 \times \mathbb{R})$ where P is a 2-variate normal distribution with domain $\mathbb{R}^2$. Extensions to higher dimensions will follow easily. We primarily follow the techniques outlined in Section 1.4.2 for a smooth terrain with properties z_P^-, d_P, and λ_P. What remains is to approximate P with another ground set P' such that P' has better bounds on the quantities $z_{P'}^-$ and $d_{P'}$. The approximation will obey Lemma 1.3, replacing P with P' by just truncating the domain of P.

The cumulative distribution function $\Phi_{m,\sigma^2}(x)$ for a normal distribution φ_{m,σ^2} describes the probability that a random variable $X \sim \mathcal{N}(m, \sigma^2)$ is less than or equal to x. We can write

$$\Phi_{m,\sigma^2}(x) = \int_{-\infty}^{x} \varphi_{m,\sigma^2}(t)\, dt = \frac{1}{\sigma\sqrt{2\pi}} \int_{-\infty}^{x} e^{-\frac{(t-m)^2}{2\sigma^2}}\, dt = \frac{1}{2}\left(1 + \mathsf{erf}\left(\frac{x-m}{\sigma\sqrt{2}}\right)\right),$$

where $\mathsf{erf}(x) = (2/\sqrt{\pi}) \int_0^x e^{-t^2}\, dt$. W.l.o.g. we can set $m = 0$. We want to find the value of x such that $\Phi_{0,\sigma^2}(x) \geq 1 - \varepsilon/4$ so that if we truncate the domain of $\varphi_{0,\sigma^2}(x)$ which is being approximated, the result will still be within $\varepsilon/2$ of the original domain. We can bound $\mathsf{erf}(x)$ with the following inequality which is very close to equality as

x becomes large:

$$1 - \mathsf{erf}(x) \leq \frac{e^{-x^2}}{x\sqrt{\pi}}.$$

For a $\gamma \leq 1/(e\sqrt{\pi})$ then

$$1 - \mathsf{erf}(x) \leq \gamma \quad \text{when} \quad x \geq \sqrt{-\ln(\gamma\sqrt{\pi})}.$$

We say that the tail of Φ_{0,σ^2} is sufficiently small when

$$1 - \Phi_{0,\sigma^2} = (1/2) - (1/2)\mathsf{erf}(x/(\sigma\sqrt{2})) \leq \varepsilon/4.$$

Thus letting $\gamma = \varepsilon/4$, this is satisfied when

$$x \geq \sigma\sqrt{2\ln(1/(\varepsilon\sqrt{\pi/2}))}.$$

Thus

$$\int_{m-\sigma\sqrt{2\ln(1/(\varepsilon\sqrt{\pi/2})}}^{m+\sigma\sqrt{2\ln(1/(\varepsilon\sqrt{\pi/2}))}} \varphi_{m,\sigma^2}(x)\, dx \geq 1 - \varepsilon/2$$

and bounding the domain of the normal distribution φ_{m,σ^2} to

$$\left[m - \sigma\sqrt{2\ln(1/(\varepsilon\sqrt{\pi/2}))}, m + \sigma\sqrt{2\ln(1/(\varepsilon\sqrt{\pi/2}))}\right]$$

will approximate the distribution within $\varepsilon/2$.

For a multivariate normal distribution, we truncate in the directions of the x- and y-axis according to the variance in each direction. Letting P' be the normal distribution with this truncated domain, then the diameter of the space is $d_{P'} = O(\sigma_{\max}\sqrt{\log(1/\varepsilon)})$, where $\sigma_{\max}$ is the maximum variance over all directions. ($\sigma_{\min}$ is the minimum variance.) In d-dimensions, the diameter is $d_{P'} = O(\sigma_{\max}\sqrt{d\log(1/\varepsilon)})$.

The lower bound $z^-_{P'}$ is now on the boundary of the truncation. In one dimension, the value at the boundary is

$$\varphi_{0,\sigma^2}\left(\sigma\sqrt{2\ln(1/(\varepsilon\sqrt{\pi/2}))}\right) = \frac{1}{\sigma\sqrt{2\pi}}e^{-\left(\sigma\sqrt{2\ln(1/(\varepsilon\sqrt{\pi/2}))}\right)^2/(2\sigma^2)} = \frac{\varepsilon}{2\sigma}.$$

For a 2-variate normal distribution the lower bound occurs at the corner of the rectangular boundary where in the 1-variate normal distribution that passes through that point and $m = 0$ the value of $x = \sqrt{2}\sigma\sqrt{2\ln(1/(\varepsilon\sqrt{\pi/2}))}$. Thus the value at the lowest point is

$$\begin{aligned} z^-_{P'} = \varphi_{0,\sigma^2}\left(\sqrt{2}\sigma\sqrt{2\ln(1/(\varepsilon\sqrt{\pi/2}))}\right) &= \frac{1}{\sigma_c\sqrt{2\pi}}e^{-\left(\sqrt{2}\sigma_c\sqrt{2\ln(1/(\varepsilon\sqrt{\pi/2}))}\right)^2/(2\sigma_c^2)} \\ &= \frac{\varepsilon^2\sqrt{\pi/2}}{2\sigma_c} \\ &= \Omega(\varepsilon^2/\sigma_c), \end{aligned}$$

where σ_{c}^2 is the variance in the direction of the corner. In the d-variate case, the lowest point is $\Omega(\varepsilon^d/\sigma_{\mathrm{c}})$.

We calculate a bound for $\lambda_{P'}$ by examining the largest second derivative along the 1-dimensional normal distribution and along an ellipse defined by the minimum and maximum variance. In the first case we can write

$$\frac{d^2\varphi_{0,\sigma^2}(x)}{dx^2} = \varphi_{0,\sigma^2}\left(\frac{x^2-\sigma^2}{\sigma^4}\right)$$

which is maximized at $x = \sqrt{3}\sigma$. And

$$\frac{d^2\varphi_{0,\sigma^2}(\sqrt{3}\sigma)}{dx^2} = \frac{1}{\sigma^3}\sqrt{\frac{2}{\pi e^3}} = O(1/\sigma^3).$$

For a bivariate normal distribution the largest eigenvalue of the Hessian of the extension of φ_{0,σ^2} is similarly not large in the tail. Thus our choice of ε does not effect this value.

Hence, we can write $\left(\lambda_{P'} d_{P'}^2/z_{P'}^-\right) = O((1/\varepsilon^2)\log(1/\varepsilon))$ for constant σ. And, using Theorem 1.5, we can state the following theorem.

Theorem 1.6. *For a d-variate normal distribution φ with constant variance, we can deterministically create an ε-sample of the range space $(\varphi, \mathcal{R}_d \times \mathbb{R})$ of size $O((1/\varepsilon^{d+2}) \cdot \log^2(1/\varepsilon))$. This can be improved to a set of size $O((1/\varepsilon)\log^{2d}(1/\varepsilon))$ in additional time $O((1/\varepsilon^{d+5})\log^{14}(1/\varepsilon))$.*

1.5 Applications

Creating smaller ε-samples improves several existing algorithms, including those described in Chapter 3 and Chapter 4.

1.5.1 Terrain Analysis

After creating an ε-sample of a terrain we are able to approximately answer questions about the integral over certain ranges. For instance, a terrain can model the height of a forest. A foresting company may deem a region ready to harvest if the average tree height is above some threshold. Computing the integral on the ε-sample will be much faster than integrating over the full terrain model.

More interesting analysis can be done by comparing two terrains. These can represent the forest height and the ground height or the elevation of sand dunes at two snapshots or the distribution of a population and a distribution of a trait of that population. Let T_1 and T_2 be two terrains defined by piecewise-linear height functions h_1 and h_2, respectively, over a subset of $\mathbb{R}^2$. The height $h = h_1 - h_2$ may be negative in some situations. This can be handled by dividing it into two disjoint terrains, where one is the positive parts of h and the other is the negative parts. Each triangle can be split by the $h = 0$ plane at most once, so this does not asymptotically change the number of piecewise-linear sections.

Once an ε-sample has been created for the positive and negative terrain, the algorithms of Chapter 4 can be used to find the rectangle with the largest positive integral. For n points this takes $O(n^2 \log n)$ time. The same can be done for finding the rectangular range with the most negative integral. The range returned indicates the region of largest difference between the two terrains. The runtime is dominated by the time to create the ε-sample in Corollary 1.4.

1.5.2 Cuts in Sensor Networks

Sensor networks geometrically can be thought of as a set of n points in a region P. These points (or nodes) need to communicate information about their environment, but have limited power. Shrivastava *et. al.* [107] investigates the detection of large disruptions to the domain that affect at least εn nodes. They want to detect these significant events but with few false positives. In particular, they do not want to report an event unless it affects at least $(\varepsilon/2)n$ nodes.

We say $Q \subseteq P$ is an *ε-sentinel* of $(P, \mathcal{A})$ if for all $R \in \mathcal{A}$

- if $\mu(R \cap P) \geq \varepsilon\mu(P)$ then $\mu(R \cap Q) \geq \varepsilon(3/4)\mu(Q)$, and
- if $\mu(R \cap Q) \geq \varepsilon(3/4)\mu(Q)$ then $\mu(R \cap P) \geq \varepsilon(1/2)\mu(P)$.

Shrivastava *et. al.* [107] construct ε-sentinels for half spaces of size $O(1/\varepsilon)$ and in expected time $O((n/\varepsilon)\log n)$. They note that an $\varepsilon/4$-approximation can be used as an ε-sentinel, but that the standard upper bound for ε-samples [116] requires roughly $O((1/\varepsilon^2)\log(1/\varepsilon))$ points which is often impractical. They pose the question: *For what other classes of ranges can an ε-sentinel be found of size* $O((1/\varepsilon)\ \text{polylog}(1/\varepsilon))$?

Followup work by Gandhi *et. al.* [45] construct ε-sentinels for any $\mathcal{A}$ with bounded VC-dimension v (such as disks or ellipses) of size $O((1/\varepsilon)\log(1/\varepsilon))$ and in time $O(n(1/\varepsilon^{2v})\log^v(1/\varepsilon))$.

As an alternative to this approach, by invoking Corollary 1.2 we show that we can construct a small ε-sentinel for $\mathcal{Q}_k$.

Theorem 1.7. *For a discrete point set P of size n, we can compute ε-sentinels for $(P, \mathcal{Q}_k)$ of size $O((1/\varepsilon)\log^{2k}(1/\varepsilon))$ in time $O(n(1/\varepsilon^3)\log^{6k}(1/\varepsilon))$.*

In fact, if we can choose where we place our nodes we can create an ε-sentinel of size $O((1/\varepsilon)\log^{2k}(1/\varepsilon))$ to monitor some region P, where P is Lebesgue-measureable. We can invoke Theorem 1.2 or Theorem 1.5, depending on the nature of P.

Additionally, by modifying the techniques of this paper, we can create $O(n\varepsilon/\log^{2k}(1/\varepsilon))$ disjoint sets of ε-sentinels. At every HALVE step of Algorithm 1.2.1 we make a choice of which points to discard. By branching off with the other set into a disjoint ε-sample, we can place each point into a disjoint ε-sentinel of size $O((1/\varepsilon)\log^{2k}(1/\varepsilon))$. Since the HALVE step now needs to be called $O(n\varepsilon/\log^{2k}(1/\varepsilon))$ times on each of the $O(\log(n\varepsilon))$ levels, this takes $O(n(1/\varepsilon^3)\log(n\varepsilon)\log^{6k}(1/\varepsilon))$ time.

Theorem 1.8. *For a discrete point set P of size n, we can create $O(n\varepsilon/\log^{2k}(1/\varepsilon))$ disjoint sets of ε-sentinels in $O(n(1/\varepsilon^3)\log(n\varepsilon)\log^{6k}(1/\varepsilon))$ total time.*

The advantage of this approach is that the nodes can alternate which sensors are activated, thus conserving power. If instead a single node is used in multiple ε-sentinels it will more quickly use up its battery supply, and when its batter runs out, the ε-sentinels using that node can no longer make the appropriate guarantees.

2

Stability of α-Kernels

2.1 Motivation

In a number of applications, the input data is being updated periodically or one may want to change some algorithm parameter slightly. Thus dynamic algorithms have been developed to update coresets on the fly [15, 29, 118, 61, 112]. Since the known algorithms focus on minimizing the size of the coreset, a weakness of most of them is that changing a single point in the input set P may drastically change the resulting coreset, e.g., this is the case for the algorithms in [15, 112, 29, 4]. This is particularly undesirable when the computed summary is the input to another dynamic data structure for maintaining another information: the dynamic data structure running on the coreset will not benefit from the small change in P as one may have to reconstruct the entire data structure. For example, kinetic data structures (KDS) based on coresets have been proposed to maintain various extent measures of a set of moving points [5]. If an insertion or deletion of an object changes the entire summary, then one has to reconstruct the entire KDS instead of locally updating it. It is also not well-understood how the coreset may change when the error parameter α changes.

In this chapter we study the *stability* of α-kernels: how α-kernels change as we update the input set or vary the value of α.

α-Kernels. For any direction $u \in \mathbb{S}^{d-1}$ let $\langle p, u\rangle$ describe the inner product of p and u. And let $P[u] = \arg\max_{p \in P} \langle p, u\rangle$ be the extreme point from P in the direction u. Let $\omega(P, u) = \langle P[u] - P[-u], u\rangle$ describe the directional width of P along the direction u. For a given $\alpha > 0$, we say that $K \subset P \subset \mathbb{R}^d$ is an α-kernel [4] of P if for all directions $u \in \mathbb{S}^{d-1}$

$$\langle P[u] - K[u], u\rangle \leq \alpha\omega(P, u).$$

Note that this is a slightly stronger version than defined in [4] and that an α-kernel K gives a relative $(1+2\alpha)$-approximation of $\omega(P,u)$ for all $u \in \mathbb{S}^{d-1}$ (i.e. $\omega(K,u) \leq \omega(P,u) \leq (1+2\alpha)\omega(K,u)$). It has been shown in [4, 5, 28, 29] that α-kernels lead to efficient approximation algorithms for a wide range of problems. For simplicity, we will assume that $\alpha \in (0,1)$, because when $\alpha \geq 1$, it is always possible to choose a constant number of points to form an α-kernel.

Agarwal et al. [4] showed that there exists an α-kernel of size $O(1/\alpha^{(d-1)/2})$ and it can be computed in time $O(n + 1/\alpha^{3d/2})$. The running time was improved by Chan [28] to $O(n + 1/\alpha^{d-3/2})$ (see also [119]). An α-kernel of size $O(1/\alpha^{(d-1)/2})$ can be maintained dynamically by spending $O((1/\alpha^{(d-1)/2}) \log n + 1/\alpha^{d-3/2})$ time per insertion or deletion [29] and can be easily adapted to have size $O((1/\alpha^{d-1}) \log n)$ and update time $O(\log n)$. Alternatively, in a streaming setting where points are only inserted, an α-kernel of size $O(1/\alpha^{(d-1)/2})$ in $O(1/\alpha^{(d-1)/2} \log(1/\alpha))$ space can be maintained in $O((1/\alpha^{(d-1)/2}) \log 1/\alpha)$ time per insertion [120]. In $\mathbb{R}^2$ this can be improved to optimal space $O(1/\sqrt{\alpha})$ with update time $O(\log 1/\alpha)$ [118]. These streaming algorithms can be converted to maintain a stable insertion-only α-kernel of size $O((1/\alpha^{(d-1)/2}) \log 1/\alpha)$ in $\mathbb{R}^d$ or $O(1/\sqrt{\alpha})$ in $\mathbb{R}^2$ with the same update times by extending standard deamortization techniques [93] in the same way as our algorithms for maintaining insertions and deletions, described below.

Dynamic stability. We call an α-kernel γ-*stable* if the insertion or deletion of a point causes the α-kernel to change by at most γ points. For brevity, if $\gamma = O(1)$, we call the α-kernel to be *stable*. An interesting question is whether there is an efficient algorithm for maintaining a stable α-kernel of size $O(1/\alpha^{(d-1)/2})$, as points are being inserted or deleted. Maintaining a stable α-kernel dynamically is difficult for two main reasons. First, for an input set P, many algorithms compute α-kernels in two or more steps. They first construct a large α-kernel K', and then use a more expensive algorithm to create a small α-kernel of K'. However, if the first algorithm is unstable, then K' may change completely each time P is updated. Second, all of the known α-kernel algorithms rely on first finding a "rough shape" of the input set P (e.g., finding a small box that contains P), estimating its fatness [18]. This rough approximation is used crucially in the computation of α-kernel. However, this shape is itself very unstable under insertions or deletions to P. Circumventing these difficulties, we prove the following (Section 2.2):

Theorem 2.1. *Given a parameter $0 \leq \alpha \leq 1$, we can maintain a stable α-kernel of size $O(1/\alpha^{(d-1)/2})$ of a point set of n points in $\mathbb{R}^d$ under insertions and deletions in $O(1/\alpha^{(d-1)/2} + \log n)$ time.*

Approximation stability. If the size of an α-kernel K is $O(1/\alpha^{(d-1)/2})$, then decreasing α changes K quite predictably. However, this is the worst-case bound, and it is possible that the size of K may be quite small, e.g., $O(1)$, or in general much smaller than the $1/\alpha^{(d-1)/2}$ maximum (efficient algorithms are known for computing

α-kernels of near-optimal size [5]). Then how much the size can increase as we reduce the allowable error from α to $\alpha/2$? Many shape simplification problems suffer from this unstability, i.e., the size of simplification can change drastically as we reduce the allowable approximation error. We prove the following result for α-kernels (Section 2.3). For any $\alpha > 0$, let $\kappa(P, \alpha)$ denote the minimum size of an α-kernel of P.

Theorem 2.2. *There are constants $c_1, c_2 > 0$ for which the following hold:*

(1) there exist a point set P and some $\alpha > 0$, such that

$$\kappa(P, \alpha/2) \geq c_1 \cdot \min\left\{\left\lceil 1/\alpha^{(d-1)/2} \right\rceil, \kappa(P, \alpha)^{\lfloor d/2 \rfloor}\right\};$$

(2) for any point set P and for any $\alpha > 0$,

$$\kappa(P, \alpha/2) \leq c_2 \cdot \min\left\{\left\lceil 1/\alpha^{(d-1)/2} \right\rceil, \kappa(P, \alpha)^{\lfloor d/2 \rfloor} \log^{d-2}(1/\alpha)\right\}.$$

2.2 Dynamic Stability

In this section we describe an algorithm that proves Theorem 2.1. Then we show in Section 2.2.1 how to maintain a stable α-kernel when the input point set retains its rough shape. Next, we extend in Section 2.2.2 our algorithm to handle the case when the rough shape of the point set changes. Finally, Section 2.2.3 composes three dynamic stable algorithms to achieve the main result, which also improves on the previous best result for dynamic (non-stable) α-kernels!

We begin by noting a powerful property of α-kernels: If K is an α-kernel of a point set P, and K' is an α'-kernel of K, then K' is an $(\alpha + \alpha')$-kernel of P. However, this property cannot, in general, be used for dynamic α-kernel algorithms because a change to P may change all of K and thus cause $O(|K|)$ updates to the dynamic algorithm creating an α'-kernel of K. However, stable α-kernels have precisely the property needed to compose dynamic α-kernel algorithms.

Lemma 2.1 (Composition Lemma). *If K is an s-stable α-kernel of P and K' is an s'-stable α'-kernel of K, then K' is an $(s \cdot s')$-stable $(\alpha + \alpha')$-kernel of P.*

Proof. The $(\alpha + \alpha')$-kernel is true by the above property. The stability results can be seen because every 1 change to P causes s changes to K, in turn, each of the s changes to K causes s' changes to K'. Thus each 1 change to P causes $s \cdot s'$ changes to K'. □

Fatness of point sets. We say a point set P is *β-fat* if

$$\frac{\max_{u \in \mathbb{S}^{d-1}} \omega(P, u)}{\min_{u \in \mathbb{S}^{d-1}} \omega(P, u)} \leq \beta.$$

If β is a constant, we sometimes just say a point set is *fat*. To make a point set P fat, we first choose a set of $d+1$ *anchor points* $A = \{a_0, a_1, \ldots, a_d\}$ using the following procedure of Barequet and Har-Peled [18]. Choose a_0 arbitrarily. Let a_1 be the farthest point from a_0. Then inductively, let a_i be the furthest point from the flat $\mathsf{span}(a_0, \ldots, a_{i-1})$. (See Figure 2.1.) The anchor points A define a bounding box I_A with center at a_0 and orthogonal directions defined by vectors from the flat $\mathsf{span}(a_0, \ldots, a_{i-1})$ to a_i. The extents of I_A in each orthogonal direction is defined by placing each a_i on a bounding face and extending I_A the same distance from a_0 in the opposite direction. Next we perform an affine transform T_A on P such that the vector from the flat $\mathsf{span}(a_0, \ldots, a_{i-1})$ to a_i is equal to e_i, where $e_0 = (0, \ldots, 0), e_1 = (1, 0, \ldots, 0), \ldots,\ e_d = (0, \ldots, 0, 1)$. This ensures that $T_A(P) \subseteq T_A(I_A) = [-1, 1]^d$.

Lemma 2.2. *For all $u \in \mathbb{S}^{d-1}$*

$$\omega(T_A(A), u) \leq \omega(T_A(P), u) \leq \omega(T_A(I_A), u) \leq \beta_d \cdot \omega(T_A(A), u), \tag{2.1}$$

for a constant $\beta_d \leq 2^d d^{3/2} d!$. Hence $T_A(P)$ is β_d-fat.

Proof. The first two inequalities follow by $A \subset P \subset I_A$. We can upper bound $\max_u \omega(T_A(I_A), u) = \omega([-1, 1]^d, u) \leq \sqrt{d}$. The volume of the convex hull $\mathsf{conv}(T_A(A))$ is $1/d!$ since it is a d-simplex and $\langle T_A(a_0) - T_A(a_i), e_i \rangle = 1$ for each direction e_i. We can then scale $\mathsf{conv}(T_A(A))$ by a factor $1/2$ (shrinking the volume by factor $1/2^d$) so it fits in $[0, 1]^d$. Now we can apply a lemma from [56] that the minimum width of a convex shape that is contained in $[0, 1]^d$ is at least $1/d$ times its d-dimensional volume, which is $1/(2^d d!)$. The fatness of $T_A(P)$ follows from the fatness of $T_A(I_A)$. $\square$

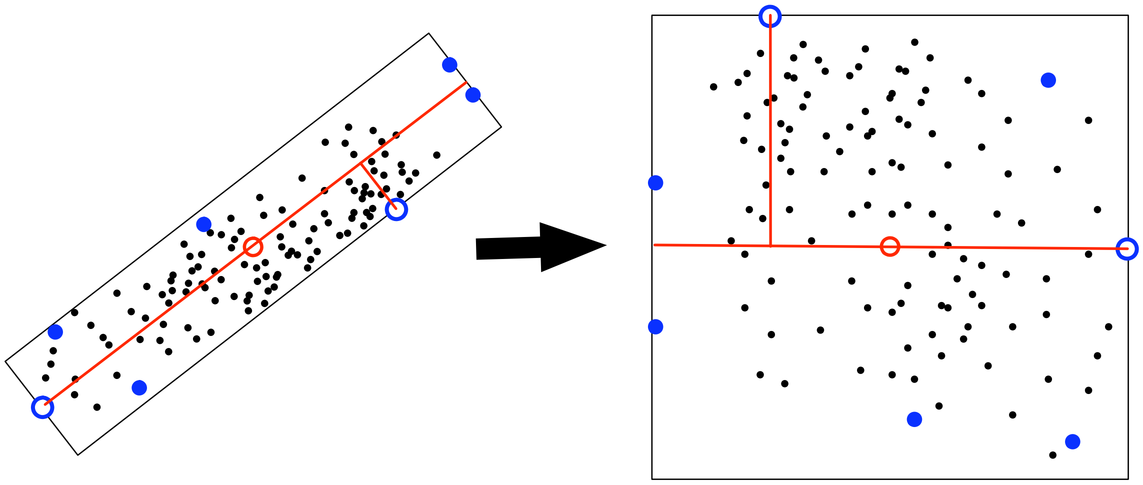


FIGURE 2.1: Transform T_A applied to points in $\mathbb{R}^2$. Anchor points are hollow.

Agarwal *et al.* [4] show if K is an α-kernel of P, then $T(K)$ is an α-kernel of $T(P)$ for any affine transform T. For any points set P and anchor points A, if

$$\langle P[u] - K[u], u \rangle \leq \alpha \omega(A, u)$$

then we say K is an *α-kernel of P with respect to A*.

2.2.1 Stable α-Kernels of Fat Point Sets

This subsection deals with algorithms that assume the input set $P \in \mathbb{R}^d$ to be fat relative to a fixed set of anchor points $A = \{e_0, \dots e_d\}$ and that $I_A = [-1, 1]^d$. We first infer some important lemmas from known α-kernel algorithms of Agarwal *et al.* [4] and Chan [28]. We then extend the framework of Chan to create an efficient, optimal-size stable α-kernel for a fat point set.

Agarwal *et al.* describes an orthogonal grid construction for an α-kernel. It starts by placing a d-dimensional orthogonal grid over the hypercube $[-1, 1]^d$ along directions $e_0, e_1, \dots, e_d$. Each side of the hypercube is divided into $2/\alpha$ intervals of width α, and the cross product of these intervals for each side forms the grid. The total number of grid cells is $O(1/\alpha^d)$. We choose an arbitrary point of P from each grid cell (if it contains a point of P) to form an α-kernel K'. We can create an α-kernel K of size $(1/\alpha^{d-1})$ by fixing one grid direction e_i and only choosing the first and last point of K' along direction e_i in K from each grid column. The set K' can be maintained in $O(1)$ time, and K can the be maintained in $O(\log 1/\alpha)$ time.

Lemma 2.3. *Let $P \subset \mathbb{R}^d$ be a point set of size n that is updated by insertions and deletions but remains of size $O(n)$ and for A, a set of anchor points, for all $u \in \mathbb{S}^{d-1}$ $\omega(P, u) \leq c\omega(A, u)$ for $c > 0$. For a given $\alpha \in (0, 1)$, we can preprocess P in $O(n)$ time using $O(n)$ space so that a 2-stable α-kernel of P with respect to A of size $O(1/\alpha^{d-1})$ can be maintained in $O(\log 1/\alpha)$ time per update.*

Chan introduces a technique called discrete Voronoi diagrams. Let $[E] = \{1, 2, \dots, E\}$ and let $E^\tau \leq F \leq E$ for some $\tau \in (0, 1)$. For a finite set of points $P \subset \mathbb{R}^d$ and a query point $q \in \mathbb{R}^d$, let $\psi_P(q)$ be the closest point to q in P.

Lemma 2.4 (Chan [28])**.** *Let $P \subset [E]^{d-1} \times \mathbb{R}$ be a set of at most E^{d-1} points. For all grid points in $b \in [F]^{d-1} \times \{0\}$, we can compute $\psi_P(b)$ in total time $O(E^{d-2}F)$.*

Chan [28] describes a framework for quickly creating an optimal size α-kernel using Lemma 2.4. He first applies the orthogonal grid construction d times for each direction e_i. That is, he constructs d α-kernels $K'_1, \dots, K'_d$, where K'_i contains the first and last grid point along each grid column in direction e_i. Let the *grid base points* B_i^s for $s \in \{-, +\}$ be a scaled and translated set from $[F]^{i-1} \times \{0\} \times [F]^{d-i}$, where $F = \lceil 2\sqrt{d/\alpha} \rceil$, so that $B_i^s \subset H_i^s = [-2, 2]^{i-1} \times \{s2\} \times [-2, 2]^{d-i}$ and no point on the hypercube H_i^s is further than $\sqrt{\alpha}$ from a point of B_i^s. Chan then uses a discrete Voronoi diagram to compute $\psi_{K'_i}(b)$ for each $b \in B_i^+ \cup B_i^-$, in $O(1/\alpha^{d-3/2})$ time. The set

$$K = \bigcup_{i=1}^{d} \bigcup_{b \in B_i^+ \cup B_i^-} \psi_{K'_i}(b)$$

forms an α-kernel of size $O(1/\alpha^{(d-1)/2})$.

A stable version of Chan's algorithm. We describe the following for one direction (w.l.o.g. e_d) and set of grid base points B_d^+, but it is repeated for all d directions $e_1, \ldots, e_d$ and corresponding sets of grid base points B_i^+ and B_I^-. Invoke Lemma 2.3 along direction e_d to create a set P_d, the *available grid input points*. The grid cells are indexed as $[i_1, \ldots, i_d]$ indicating the number of grid cells along each direction it is from the corner of I at $(-1, \ldots, -1)$. Create a set of $O(1/\alpha^{(d-1)/2})$ groups of columns along direction e_d. For $(j_1, \ldots, j_{d-1}) = J \in [F]^{d-1}$ and $F = \lceil 2\sqrt{d/\alpha}\rceil$, let group

$$G_J = [j_1\lceil\sqrt{\alpha/d}\rceil, (j_1+1)\lceil\sqrt{\alpha/d}\rceil] \times \ldots \times [j_{d-1}\lceil\sqrt{\alpha/d}\rceil, (j_{d-1}+1)\lceil\sqrt{\alpha/d}\rceil] \times [-1, 1].$$

Each group G_J accounts for up to $O(1/\alpha^{(d-1)/2})$ points $P'_J \subset P_d$.

For all grid base points $b \in B_d^+$ and for each group G_J, find a *deputy point* $a_J = \psi_{P'_J}(b)$; Set $D_b = \{a_J \mid J \in [F]^{d-1}\}$. The set $\bigcup_{i=1}^{d} \bigcup_{s\in\{-,+\}} \{\psi_{D_b}(b) \mid b \in B_i^s\}$ is an α-kernel of P, where D_b is defined symmetrically for each direction and sign, but we add points to the kernel iteratively to make it stable.

Assign an arbitrary order to the points $\langle b_1, b_2, \ldots\rangle = B_d^+$, and process them one-by-one as follows, starting with $b_i = b_1$. For b_i assign $a^*_{b_i} = \psi_{D_{b_i}}(b_i)$ to K. For the group G_J such that $a^*_{b_i} \in P'_J$, remove $a^*_{b_i}$ from P'_J and update $a^*_{b_i}$'s column in the grid as in Lemma 2.3. Then recalculate the deputy point a_J for all $b \in B_d^+$ in $O(1/\alpha^{(d-1)/2})$ time. Then process b_{i+1}. This guarantees the following two properties

(P1) $a^*_{b_i} \neq a^*_{b_j}$ for $i \neq j$

(P2) each $a^*_{b_i}$ is closer to b_i than any other point in $P_d \setminus \{a^*_{b_1}, \ldots, a^*_{b_{i-1}}\}$.

This preprocessing runs in a total time of $O(1/\alpha^{d-1})$.

We can now handle insertions and deletions stably in $O(1/\alpha^{(d-1)/2})$ time as long as inserted points are within I_A and the point set remains fat with respect to A. If a point p is inserted, we process all grid base points B_d^+ one-by-one in the same order as above, starting with $b_i = b_1$ and $q = p$. If $||q - b_i|| < ||a^*_{b_i} - b_i||$ then we replace $a^*_{b_i}$ in K with q and set $q = a^*_{b_i}$, then process b_{i+1} with the new value of q. In the end, only one point is removed from K, and all deputy points for that point's group can be updated in $O(1/\alpha^{(d-1)/2})$ time as above. Properties (P1) is maintained because we only allow each q to replace one $a^*_{b_i}$ and (P2) are maintained because if any q is closer than $a^*_{b_i}$ to b_i, then it is replaced.

If a point p is deleted from P'_J, we recompute all deputies a_J for all points B_d^+. If p was in K as the closest point to $b \in B_d^+$, then we recompute $a^*_b = \psi_{D_b}(b)$ and place it in K, and we move b to the end of the order of points in B_d^+. For the group G_J such that $a^*_b \in P'_J$ we recompute the deputies a_J for all points B_d^+. Property (P1) is maintained because a^*_b is replaced with a new point if needed. We maintain property (P2) by moving b to the end of the order of points because all other points already conformed to (P2) ignoring b, then since b is at the end of the list we just need to find $\psi_{P'\setminus K}(b)$. This process takes $O(1/\alpha^{(d-1)/2})$ time because deputy points are recomputed for up to two groups and $\psi_{P_J}(b)$ is calculated at most once.

Lemma 2.5. *Let $P \subset \mathbb{R}^d$ be a point set of size n that is updated by insertions and deletions but remains of size $O(n)$ and for A, a set of anchor points, for all $u \in \mathbb{S}^{d-1}$ $\omega(P,u) \leq c\omega(A,u)$ for $c > 0$. For a given $\alpha \in (0,1)$, we can preprocess P in $O(n + 1/\alpha^{d-1})$ time so that a stable α-kernel of P with respect to A of size $O(1/\alpha^{(d-1)/2})$ can be maintained in $O(1/\alpha^{(d-1)/2})$ time per update.*

2.2.2 Stable α-Kernels with Varying Fatness

In this subsection we describe two related algorithms for stable α-kernels that do not require the point set to remain fat. The first divides the point set into inner and outer points so an α-kernel with respect to a fixed set of anchor points can be maintained for the inner points for a specified number of updates. The second technique, a modification of Chan's dynamic coreset algorithm [29], creates several layers of point sets where the more inner layers maintain α-kernels with respect to fixed anchor points for a larger specified number of updates. We apply techniques from the previous subsection on the inner points for both techniques.

Outer kernels. We create an *outer kernel* by pealing off m "layers" of anchor points, we will chose m to be $1/\alpha^{d-1}$ or $1/\alpha^{(d-1)/2}$. We first find an approximate center point a_0 of P. We then select d other anchor points according to Barequet and Har-Peled [18]. Let A_1 denote this first set of anchor points, where a_0 is the first anchor point, and set $P_1 = P \setminus A_1$. We repeat this m times, finding each set of anchor points A_i from P_{i-1} and always using a_0 as the first anchor point. The last set of anchor points is labeled $A' = A_m$ and the remaining points $P' = P_m$. Let $A = \bigcup_{i=1}^m A_i$ be called the outer kernel.

If $m = 1/\alpha^{d-1}$, then we construct a stable α-kernel K_I for P' with respect to A' using Lemma 2.3, and if $m = 1/\alpha^{(d-1)/2}$ we construct a stable α-kernel K_I for P' with respect to A' using Lemma 2.5. We set $K = K_I \cup A$ to be the α-kernel of P.

We maintain K under insertion or deletion of points from P as follows. If a point in $I_{A'}$ is removed or inserted, we maintain K_I using Lemma 2.5 or Lemma 2.3. If we insert or delete a point outside of $I_{A'}$ we insert it to or delete it from A. After $O(m)$ insertions or deletions outside of $I_{A'}$ we rebuild the entire structure (outer and inner kernel). Since, rebuilding the α-kernel requires we update K with at most $O(m)$ points and only occurs once every at least $\Omega(m)$ updates, in an amortized sense these algorithms are stable.

Naively we can construct the outer kernel $A \subset P$ of size $O(m)$ in $O(mn)$ time, where $|P| = n$. Each set A_i takes $O(n)$ time to construct with $O(d)$ scans over P_{i-1} to find the furthest point. Alternatively, we can build the outer kernel using Chan's dynamic α-kernel algorithm. We construct a τ-kernel K of P of size $O(\frac{1}{\tau^{d-1}} \log n)$ in $O(n)$ time, for a constant $\tau \leq 1$. We then build a set of anchor points A_1 from K in $O(\frac{1}{\tau^{d-1}} \log n)$ time, and remove each point in $O(\log n)$ time. This causes $T_{A_1}(P_1)$ to be (β_d/τ)-fat. Thus to build m sets of anchor points takes $O(n + m\frac{1}{\tau^{d-1}} \log n)$ time, which when $\tau \leq 1$ and $1/\tau = O(1)$ is $O(n + m \log n)$ time.

Lemma 2.6. *If fewer than m points have been inserted or deleted from P since the last time $K = K_I \cup A$ was built, then K is an $(\alpha\beta_d)$-kernel of P of size $O(m)$.*

Proof. We can guarantee that at least one set of anchor points (w.l.o.g. A_i) is still completely in K. Thus we can show that $A_i \cup K_I$ forms an α-kernel of $A_i \cup P'$. For any direction $u \in \mathbb{S}^{d-1}$

$$\begin{aligned}
\omega(K_I, u) \leq \omega(P', u) &\leq \omega(K_I, u) + \alpha \cdot \omega(P' \cup A', u) \\
&\leq \omega(K_I, u) + \alpha \cdot \omega(P_i, u) \\
&\leq \omega(K_I, u) + \alpha \cdot \omega(I_i, u) \\
&\leq \omega(K_I, u) + \alpha\beta_d \cdot \omega(A_i, u) \qquad \text{[via (2.1)]} \\
&\leq \omega(K_I, u) + \alpha\beta_d \cdot \omega(P' \cup A_i, u).
\end{aligned}$$

Thus, for any direction $u \in \mathbb{S}^{d-1}$ the extreme point of $P' \cup A_i$ is either in P' or A_i. In the first case, K_I approximates the width within a factor of $\alpha\beta_d \cdot \omega(P' \cup A_i, u)$. In the second case, the extreme point is in K because all of A_i is in K. Thus the set $P' \cup A_i$ has an α-kernel in K, and the rest of the points are also in K, so K is an α-kernel of the full set P.

The size of K starts at $O(m)$ because both K_I and A are of size $O(m)$. At most m points are inserted outside of $I_{A'}$ and hence into A, thus the size of $K = K_I \cup A$ is still $O(m)$ after m steps. □

Lemma 2.7. *For a point set $P \subset \mathbb{R}^d$ of size n and a parameter $\alpha \in (0, 1)$, we can maintain a stable α-kernel of P under insertions and deletions with*

(a) size $O(1/\alpha^{(d-1)/2})$ and update time $O(n\alpha^{(d-1)/2} + 1/\alpha^{(d-1)/2} + \log n)$.

(b) size $O(1/\alpha^{d-1})$ and update time $O(n\alpha^{d-1} + \log n)$.

Proof. We build an outer kernel of size $O(m)$ in $O(n + m \log n)$ time. It lasts for $\Omega(m)$ insertions or deletions, so its construction time can be amortized over that many steps, and thus it costs $O(n/m + \log n)$ time per insertion or deletion.

In maintaining the inner kernel the preprocessing time can be amortized over m steps, but the update time cannot. In case (a) we maintain the inner kernel of size $m = O(1/\alpha^{(d-1)/2})$ with Lemma 2.5. The update time is $O(n\alpha^{(d-1)/2} + 1/\alpha^{(d-1)/2})$. In case (b) we maintain the inner kernel of size $m = O(1/\alpha^{d-1})$ with Lemma 2.3. The update time is $O(n\alpha^{d-1} + \log(1/\alpha))$.

The update time can be made worst case using a standard deamortization techniques [93]. More specifically, we start rebuilding the inner and outer kernels after $m/2$ steps and spread out the cost over the next $m/2$ steps. We put all of the needed insertions in a queue, inserting a constant number of points to K each update to P. Then after the new kernel is built, we enqueue required deletions from K and perform a constant number each update to P over the next $m/2$ steps. □

Chan's dynamic coresets. Chan [29] describes a dynamic α-kernel algorithm for a set of n points P that decomposes P into $h = O(\log n)$ disjoint point sets $\langle P_1, \ldots, P_h \rangle$, where $|P_i| \geq \gamma \sum_{j=i+1}^{h} |P_j|$ for some constant $\gamma > 1$ so $|P_i| \geq 2|P_{i+1}|$, and $|P_h| = 1/\alpha^{d-1}$. P_1 is constructed first, and recursively each P_i are the "inner" remaining points. For each set P_i (for $i < h$) there exists a set of anchor points A_i defining a bounding box I_{A_i} (i.e., $P_i \subset I_{A_i}$) and for which an α-kernel of P_i can be maintained with respect to for at least $\Omega(|P_i|)$ insertions or deletions to P. The insertion of a point p into P can be placed in P_i such that $p \in I_{A_i}$ for the smallest i. We can then invoke Lemma 2.3 to maintain a stable α-kernel K_i for P_i with respect to A_i, and amortize the cost of rebuilding K_i to the number of insertions or deletions we construct a new set A_i. The last set P_h has size $O(1/\alpha^{d-1})$ so we set $K_h = P_h$. The set $K = \bigcup_{i=1}^{h} K_i$ is an α-kernel of P.

Lemma 2.8. *For any $\alpha \in (0, 1)$, Chan's dynamic coreset algorithm can maintain a stable α-kernel of size $O((1/\alpha^{d-1}) \log n)$ that can be updated in time $O(\log n)$ per insertion or deletion.*

Proof. First, we clarify that the kernel K_i of P_i needs to be rebuilt if $c|P_i|$ insertions or deletions have taken place, for some constant c, or if K_l needs to be rebuilt, for $l < i$. We can handle the second case by rebuilding K_i when K_{i+1} is rebuilt and K_i would need to be rebuilt before K_{i+1} would otherwise need to be rebuilt again. Since $|P_i| \geq 2|P_{i+1}|$, this only decreases the number of updates to P before K_i is rebuilt be a constant fraction because K_{i+1} will be rebuilt at least twice before K_i is rebuilt.

Given this adjusted updating plan it takes at least twice as many updates to P before K_i is rebuilt, compared to K_{i+1}. And K_h is rebuilt after c/α^{d-1} updates, for a constant c. Since the entire system is rebuilt after $\Theta(|P_1|) = \Theta(n)$ updates, we call this interval a round. We can bound the updates to K in a round by charging $O(1/\alpha^{d-1})$ each time a K_i is rebuilt.

$$\sum_{i=1}^{h} \frac{\Theta(n)}{\Theta(|P_i|)} \cdot O(1/\alpha^{d-1}) \leq \sum_{i=1}^{h} \frac{\Theta(n) \cdot O(1/\alpha^{d-1})}{\Omega(|P_h| 2^{h-i})} = \sum_{i=1}^{h} \frac{\Theta(n)}{\Omega(2^i)} = O(n).$$

Thus there are $O(n)$ updates to the α-kernel K for every $\Theta(n)$ updates to P. Thus, in an amortized sense, for each update to P there are $O(1)$ updates to K.

This process can be deamortized, in a similar way as with outer kernels. If a kernel K_i is valid for k insertions or deletions to P, then we start construction on the next kernel K_i after $k/2$ insertions or deletions have taken place since the last time K_i was rebuilt. All insertions can be put in a queue and added to K by the time $k/2$ steps have transpired. All deletions from old K_i to new K_i are then queued and removed from K before another $k/2$ insertions or deletions. This can be done by performing $O(1)$ queued insertions or deletions from K each insertion or deletion from P. □

2.2.3 Putting It All Together

For a point set $P \subset \mathbb{R}^d$ of size n, we can produce the best size and update time tradeoff for stable α-kernels by invoking Lemma 2.1 to compose three stable α-kernel algorithms, as illustrated in Figure 2.2. We first apply Lemma 2.8 to maintain a stable $(\alpha/3)$-kernel K_1 of P of size $O((1/\alpha^{d-1})\log n)$ with update time $O(\log n)$. We then apply Lemma 2.7(b) to maintain a stable $(\alpha/3)$-kernel K_2 of K_1 of size $O(1/\alpha^{d-1})$ with update time $O(|K_1|\alpha^{d-1}+\log|K_1|) = O(\log n+\log(1/\alpha))$. Finally we apply Lemma 2.7(a) to maintain a stable $(\alpha/3)$-kernel K of K_2 of size $O(1/\alpha^{(d-1)/2})$ with update time $O(|K_2|\alpha^{(d-1)/2} + 1/\alpha^{(d-1)/2} + \log|K_2|) = O(1/\alpha^{(d-1)/2})$. K is a stable α-kernel of P of size $O(1/\alpha^{(d-1)/2})$ with update time $O(\log n + 1/\alpha^{(d-1)/2})$. This completes the proof of Theorem 2.1.

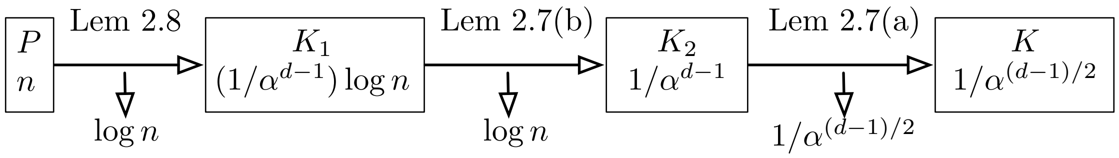


FIGURE 2.2: Composing stable α-kernel algorithms

2.3 Approximation Stability

In this section we prove Theorem 2.2. We first give a short proof for the lower-bound and then a more involved proof of the upper bound.

2.3.1 Lower Bound

Take a cyclic polytope with n vertices and $\Omega(n^{\lfloor d/2 \rfloor})$ facets and convert it into a fat polytope $\mathcal{P}$ using standard procedures [4]. For a parameter $\alpha > 0$, we add, for each facet f of $\mathcal{P}$, a point p_f that is α far away from the facet. Let P be the set of vertices of $\mathcal{P}$ together with the collection of added points. We choose α sufficiently small so that points in P are in convex position and all non-facet faces of $\mathcal{P}$ remain as faces of $\mathrm{conv}(P)$. Then the size of an optimal α-kernel of P is at most n (by taking the vertices of $\mathcal{P}$ as an α-kernel), but the size of an optimal $\alpha/2$-kernel is at least the number of facets of $\mathcal{P}$, because every point of the form p_f has to be present in the kernel. The first half of the lower bound is realized with $O(1/\alpha^{(d-1)/2})$ evenly-spaced points on a sphere, and hence the full lower bound is proved.

2.3.2 Upper Bound

By [4], it suffices to consider the case in which P is fat and the diameter of P is normalized to 1. Let C be an α-kernel of P of the smallest size. Let $\mathcal{P} = \mathrm{conv}(C)$, and $\mathcal{P}_\alpha = \mathcal{P} \oplus \alpha\mathbb{B}^d$. We have $\mathcal{P} \subseteq \mathrm{conv}(P) \subseteq \mathcal{P}_\alpha$ by the definition of α-kernels. It suffices to show that there is a set $C' \subseteq P$ such that for $\mathcal{P}' = \text{conv}(C')$, $\mathcal{P}' \subseteq \text{conv}(P) \subseteq \mathcal{P}'_{\alpha/2}$, and $|C'| = O(|C|^{\lfloor d/2 \rfloor} \log^{d-2}(1/\alpha))$ [4].

For convenience, we assume that C' is not necessarily a subset of points in P; instead, we only require C' to be a subset of points in $\text{conv}(P)$. By Caratheodory's theorem, for each point $x \in C$, we can choose a set $P_x \subseteq P$ of at most $d+1$ points such that $x \in \text{conv}(P_x)$. We set $\bigcup_{x \in C'} P_x$ as the desired $(\alpha/2)$-kernel of P; $|P_x| \leq (d+1)|C'| = O(\kappa(P,\alpha)^{\lfloor d/2 \rfloor} \log^{d-2}(1/\alpha))$.

We also assume that $\mathcal{P}$ is a simplicial polytope. The proof can be extended to a non-simplicial polytope by triangulating each non-simplicial face into simplicies and then applying a similar argument.

Construction of C'. For each face f of $\mathcal{P}$, we denote by $f^* \subseteq \mathbb{S}^{d-1}$ the dual of f in the Gaussian diagram of $\mathcal{P}$. Recall that if f has dimension k $(0 \leq k \leq d-1)$, then f^* has dimension $d-1-k$. The region $\mathcal{P}_\alpha \setminus \text{int}\, \mathcal{P}$ is partitioned into a collection of $|\mathcal{P}|$ regions (where $|\mathcal{P}|$ is the number of faces of all dimensions in $\mathcal{P}$) as follows (see Figure 2.3):

$$\{\sigma(f) = f \times (\alpha f^*) \mid f \text{ is a face of } \mathcal{P}\}.$$

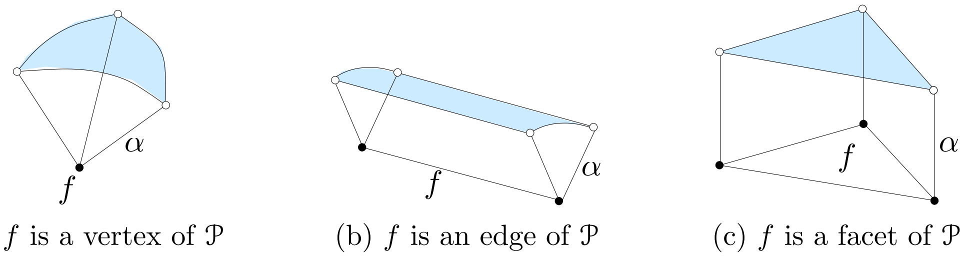

α
f
(a) f is a vertex of $\mathcal{P}$
α
f
(b) f is an edge of $\mathcal{P}$
α
f
(c) f is a facet of $\mathcal{P}$

FIGURE 2.3: An illustration of different types of regions in the partition of $\mathcal{P}_\alpha \setminus \text{int}\, \mathcal{P}$ in three dimensions.

The number of regions in the decomposition is equal to $|\mathcal{P}| = O(|C|^{\lfloor d/2 \rfloor})$ by the Upper Bound Theorem [85]. For each region $\sigma(f)$, we next describe a procedure to choose a set $K_f \subseteq \text{conv}(P)$ of $O(\log^{d-2}(1/\alpha))$ points such that

$$\text{conv}(K_f) \subseteq \text{conv}(P) \cap \sigma(f) \subseteq \text{conv}(K_f) \oplus (\alpha/2)\mathbb{B}^d.$$

The proof is completed by setting $C' = \bigcup_f K_f$.

To provide better intuition, we first give an informal overview of the construction of K_f. At a high level, we sample a constant number of directions in f^*, and for each sample direction u, a constant number of slices of $f \times \alpha u$ along direction u. For each slice (which is a translated copy of f), we recursively construct a certain exponentially distributed grid inside the slice (see Figure 2.4 for an illustration). Intuitively, the grid points that fall inside $\text{conv}(P)$ provide a good approximation

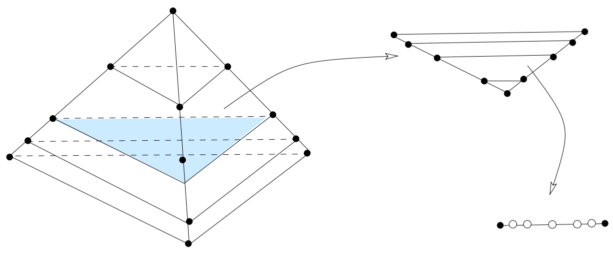


FIGURE 2.4: Recursively constructing an exponentially distributed grid in a simplex.

and are retained in K_f. For technical reasons, the actual construction of K_f is more involved.

For convenience, in the following we write a point $q = \bar{q} + z \cdot u \in \sigma(f)$, where $\bar{q} \in f, 0 \le z \le \alpha$, and $u \in f^*$, in the form of $\bar{q}^z[u]$ (which intuitively reads, the point whose projection onto f is $\bar{q}$ and which is at a distance z above f in direction u). We also write $q[v] = \bar{q} + z \cdot v$ (intuitively, $q[v]$ is obtained by rotating q w.r.t. f from direction u to direction v). Similarly, we write a simplex $\bar{\Delta}^z[u] = \bar{\Delta} \oplus z \cdot u$, where $\bar{\Delta}$ is a simplex inside f, $0 \le z \le \alpha$, and $u \in f^*$, and write $\Delta[v] = \bar{\Delta} \oplus z \cdot v$.

Let k be the dimension of f. Set $\delta = \alpha/10d$. In order to construct K_f, we construct a set Σ of point-simplex pairs, each of the form $\langle p, \Delta \rangle$ with $p \in \Delta$. The set K_f is the collection of points in these point-simplex pairs. Σ is the union of at most k sets $\Sigma_k, \Sigma_{k-1}, \cdots$, which are defined in a recursive manner: each Σ_j is defined by a set $\mathcal{L}_j$ of simplexes, and $\mathcal{L}_j$ is in turn defined by Σ_{j+1}. In more detail:

- Initially, we choose an arbitrary direction $v \in f^*$ and set $\mathcal{L}_k = \{f^\alpha[v]\}$.
- Suppose $\mathcal{L}_j$ has been defined for some index $j \le k$. We construct Σ_j as follows. For each simplex $\Delta \in \mathcal{L}_j$, and each $i = 5d, 5d+1, \cdots, 10d-1$, we set $\Delta_i = \Delta^{i\delta}$ and proceed as follows. We find a $(1/10d)$-*net* N_{Δ_i} of the set $U_{\Delta_i} = \{u \in f^* \mid \Delta_i[u] \cap \mathrm{conv}(P) \ne \emptyset\}$, so that for any $u \in U_{\Delta_i}$ there exists some $v \in N_{\Delta_i}$ such that the angle spanned between u and v is at most $1/10d$. Note that $|N_{\Delta_i}| = O(1)$. For each $u \in N_{\Delta_i}$, choose an arbitrary point p (called *support point*) from $\Delta_i[u] \cap \mathrm{conv}(P)$, and add $\langle p, \Delta_i \rangle$ into the set Σ_j.
- Finally, for each $\langle p, \Delta \rangle \in \Sigma_j$, we construct a set $H_{p,\Delta}$ of $O(\log(1/\alpha))$ simplexes within Δ (to be described shortly). We set $\mathcal{L}_{j-1} = \bigcup_{\langle p, \Delta \rangle \in \Sigma_j} H_{p,\Delta}$ and continue the recursion.
- For $k \le d-2$, we stop the recursion naturally at $j = 0$. For $k = d-1$, we stop the recursion at $j = 1$ as follows: for each line segment $\ell \in \mathcal{L}_1$ that intersects $\mathrm{conv}(P)$, we find the two endpoints q_1, q_2 of $\ell \cap \mathrm{conv}(P)$ and add $\langle q_1, \ell \rangle, \langle q_2, \ell \rangle$ into Σ_1.

The sizes of $\mathcal{L}_j$ and Σ_j satisfy the following recurrence for some constants $c_3, c_4 > 0$:

$$|\Sigma_j| \le c_3 |\mathcal{L}_j|, \quad |\mathcal{L}_{j-1}| \le c_4 \log(1/\alpha) |\Sigma_j|, \qquad \text{for } j \le k.$$

A simple calculation shows that $|K_f| \le \sum_{j \le k} |\Sigma_j| = O(\log^k(1/\alpha))$ for $k \le d-2$, and $|K_f| \le \sum_{j \le k} |\Sigma_j| = O(\log^{k-1}(1/\alpha))$ for $k = d-1$. In both cases, $|K_f| = O(\log^{d-2}(1/\alpha))$.

It remains to describe the construction of $H_{p,\Delta}$, for a point p and a j-simplex Δ, with $p \in \Delta$ and the diameter of Δ bounded by 1. For each facet H of Δ, we proceed as follows (see Figure 2.5). Let $q_1, \cdots, q_j$ be the vertices of H. For $1 \le i \le j$, let x_i be the point on the line segment $q_i p$ that is δ away from q_i, and let d_i be the distance from x_i to the facet H. Set $d_H = \min_{1 \le i \le j} d_i$. The value d_H has the following property:

Lemma 2.9. *For any point $q \in \Delta$ which is at most d_H distance away from H, let q' be the intersection of the line segment qp with the $(j-1)$-flat at distance d_H from H; then $\|qq'\| \le \delta$.*

Consider $(j-1)$-flats in the span of Δ that are at distances $d_H, (1+1/10d^2)d_H, (1+1/10d^2)^2 d_H, \cdots$ from H and have nonempty intersections with the j-simplex formed by H and p. The set of nonempty intersections of these flats with Δ is called *a family of $(j-1)$-simplicies induced by the facet H*. Because the diameter of Δ is at most 1, it can be shown that the number of such simplexes is $O(\log(1/\delta)) = O(\log(1/\alpha))$. Repeat this construction for each facet of Δ, and the collection of all resulting simplexes constitutes $H_{p,\Delta}$. We have $|H_{p,\Delta}| = O(\log(1/\alpha))$.

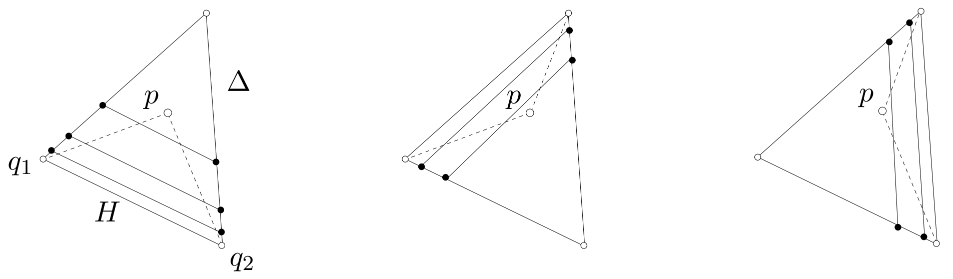


FIGURE 2.5: The construction of $H_{p,\Delta}$.

Proof of correctness. Let f be a k-face of $\mathcal{P}$ ($0 \le k \le d-1$). We need to show $\mathrm{conv}(P) \cap \sigma(f) \subseteq \mathrm{conv}(K_f) \oplus (\alpha/2)\mathbb{B}^d$, or in other words, for any point $p \in \mathrm{conv}(P) \cap \sigma(f)$, there is a point $q \in \mathrm{conv}(K_f)$ such that $\|pq\| \le \alpha/2$. If $p \in f \times (\alpha/2) f^*$, then clearly the projection q of p onto f, which belongs to $\mathrm{conv}(K_f)$ as $f \subseteq \mathcal{P} \subseteq \mathrm{conv}(K_f)$, satisfies $\|pq\| \le \alpha/2$. So in the following, we assume $p = \bar{p}^z[u]$ for some $\bar{p} \in f$, $u \in f^*$ and $\alpha/2 < z \le \alpha$. Let h be the largest multiple of δ with $h \le z$.

Before describing the technical details, we first provide some intuition regarding the proof. Eventually, we need to find a nearby point of p in $\mathrm{conv}(K_f)$. Our basic strategy is to find some "helper point" p_{k-1} for p so that if p_{k-1} has a nearby point in conv(K_f), then so does p. If p_{k-1} itself is in conv(K_f), then we are done. Otherwise, we recursively find a "helper point" p_{k-2} for p_{k-1}, and a "helper point" p_{k-3} for p_{k-2}, and so on. We stop the recursion when we have found a "helper point" which itself is in conv(K_f). Tracing back along the recursion, we can then prove that p has a nearby point in conv(K_f).

Formally, for $j = k, k-1, \cdots$, we now make a sequence of recursive definitions, for a simplex $\bar{\Delta}_j \subseteq f$, a direction u_j, a point $\bar{p}_j \in f$, and a real value $h_j \leq \alpha$. As a convention, for each j, Δ_j is implicitly defined as $\bar{\Delta}_j^{h_j}[u_j]$, and p_j is implicitly defined as $\bar{p}_j^{h_j}[u_j]$. We will maintain the following invariants for each index j:

(a) $p_j \in \text{conv}(P)$ and $p_j \in \Delta_j$;

(b) h_j is a multiple of δ, and $h_{j+1} - \delta \leq h_j \leq h_{j+1}$ (for $j \neq k$);

(c) the dimension of $\bar{\Delta}_j$ is j; and

(d) $\|u_j - u_{j+1}\| \leq 1/10d$ (for $j \neq k$).

Initially, we set $\bar{\Delta}_k = f, u_k = u, \bar{p}_k = \bar{p}$, and $h_k = h$. Note that $p_k \in \text{conv}(P)$ (because $p_k \in p\bar{p}$) and $p_k \in \Delta_k$.

Assume that for an index $j \leq k$, $\bar{\Delta}_j$, u_j, $\bar{p}_j$, and h_j have been defined. Since (a) implies $\Delta_j \cap \text{conv}(P) \neq \emptyset$ and as such $u_j \in U_{\Delta_j}$, there is a direction $u'_{j-1} \in N_{h_j, \bar{\Delta}_j}$ such that the angle between u_j and u'_{j-1} is at most $1/10d$. Set $p''_{j-1} = p_j[u'_{j-1}]$. Let η_{j-1} be the support point selected from $\Delta_j[u'_{j-1}] \cap \text{conv}(P)$. Let H be the facet of $\Delta_j[u'_{j-1}]$ that is intersected by the ray $\overrightarrow{\eta_{j-1}p''_{j-1}}$. Let Δ'_{j-1} be the first simplex in the family of $(j-1)$-simplexes induced by H inside $\Delta_j[u'_{j-1}]$ that are intersected by the ray $\overrightarrow{p''_{j-1}\eta_{j-1}}$, and let p'_{j-1} be their intersection. (See Figure 2.6). If the ray $\overrightarrow{\eta_{j-1}p''_{j-1}}$ does not intersect any simplex in the family of $(j-1)$-simplexes induced by H inside $\Delta_j[u'_{j-1}]$, we say that p_j *is close to the boundary.*

Next we consider two cases:

Case 1: (Figure 2.6 (a)) If p'_{j-1} does not exist, we set $p_{j-1} = \eta_{j-1}$ and terminate the recursion early.

Case 2: (Figure 2.6 (b)) Otherwise (i.e., p'_{j-1} lies on the segment $p''_{j-1}\eta_{j-1}$). We set $\bar{p}_{j-1}$ as the projection of p'_{j-1} onto f and $\bar{\Delta}_{j-1}$ as the projection of Δ'_{j-1} onto f. It remains to define u_{j-1} and h_{j-1}. Observe that $p_j, \eta_{j-1} \in \text{conv}(P)$, and as such $\overline{p_j\eta_{j-1}} \subseteq \text{conv}(P)$. There is a point q on $\overline{p_j\eta_{j-1}}$ whose projection onto f is $\bar{p}_{j-1}$. Let $q = \bar{p}_{j-1} + h'u_{j-1}$ for some h' and $u_{j-1} \in f^*$. Observe that $\|u_{j-1} - u_j\| \leq \|u'_{j-1} - u_j\| \leq 1/10d$. Using this observation, it can be shown that $h_j - \delta \leq h' \leq h_j$. Let h_{j-1} be the largest multiple of δ with $h_{j-1} \leq h'$. This completes the definitions of $\bar{\Delta}_{j-1}$, u_{j-1}, $\bar{p}_{j-1}$ and h_{j-1}, and transitively Δ_{j-1} and p_{j-1}. As for the invariants, we have shown that (b)–(d) are all properly maintained. For (a), observe that p_{j-1} lies on $q\bar{p}_{j-1}$, and $q, \bar{p}_{j-1} \in \text{conv}(P)$. Hence $p_{j-1} \in \text{conv}(P)$. It is also clear that $p_{j-1} \in \Delta_{j-1}$ by the above construction.

Assume the recursion does not terminate early. For the case $k < d-1$, we stop the recursion at $j = 0$ and set $p_0 = \eta_0$. For the case $k = d-1$, we stop the recursion

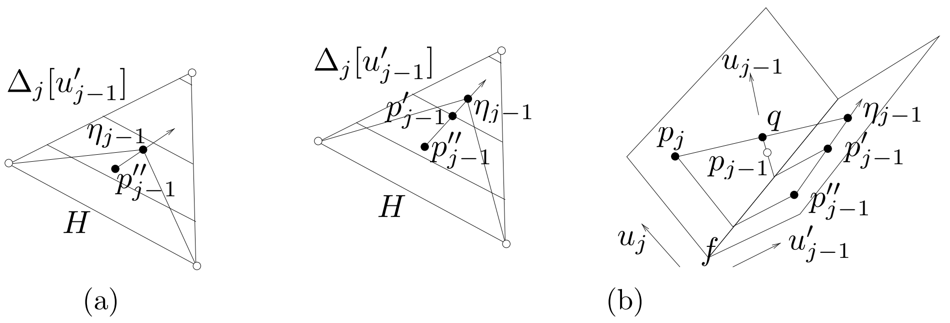


FIGURE 2.6: (a) Case 1: p'_{j-1} does not exist. (b) Case 2: p'_{j-1} lies on the segment $p''_{j-1}\eta_{j-1}$.

at $j = 1$. Note that for $k = d - 1$, since f^* is a singleton set, it can be proved inductively that $p_j \in \mathrm{conv}(P)$ for all $1 \le j \le k$.

This completes the description of the recursive definition.

Let p_m be the last point defined in the sequence. By construction, $p_m \in \mathrm{conv}(P)$. For each $m \le j \le k$, let $q_j = p_j[u_m]$ and $\Xi_j = \Delta_j[u_m]$. We have the following key lemma.

Lemma 2.10. *For each $j \ge m$, there is a point $q'_j \in \Xi_j$ such that*

(1) $\|q'_j q_j\| \le j\delta$; and
(2) $q''_j = q'_j - 2(j-m)\delta u_m \in \mathrm{conv}(P)$.

Proof. We prove the lemma by induction on j. For $j = m$, since $q_m = p_m \in \mathrm{conv}(P)$, the claim is trivially true by setting $q'_m = q_m$. Assume the claim is true for some $j \ge m$. Now consider the case $j+1$. Let y be the intersection of the ray $\overrightarrow{\bar{p}_j\bar{p}_{j+1}}$ with ∂f. Let x_j be the projection of q'_j onto f, and let x_{j+1} be the intersection of yx_j with the line passing through $\bar{p}_{j+1}$ and parallel to $\overline{\bar{p}_j x_j}$ (see Figure 2.7). There are two cases:

Case 1: p_{j+1} is close to the boundary. Then by Lemma 2.9, we know that $\|\bar{p}_{j+1}\bar{p}_j\| \le \delta$. We set $q'_{j+1} = {x_j}^{h_{j+1}}[u_m]$. As such,

$$\|q'_{j+1}q_j\| = \|\bar{p}_{j+1}x_j\| \le \|\bar{p}_{j+1}\bar{p}_j\| + \|\bar{p}_j x_j\| \le \delta + j\delta = (j+1)\delta.$$

Moreover, since $h_{j+1} - 2(j+1-m)\delta \le h_j - 2(j-m)\delta$, q''_{j+1} lies on the segment $q''_j x_j$ and therefore $q''_{j+1} \in \mathrm{conv}(P)$.

Case 2: Otherwise. In this case, we have

$$\frac{\|x_j x_{j+1}\|}{\|x_j y\|} = \frac{\|\bar{p}_j\bar{p}_{j+1}\|}{\|\bar{p}_j y\|} \le \frac{1/10d^2}{1 + 1/10d^2} \le 1/10d^2.$$

We set $q'_{j+1} = {x_{j+1}}^{h_{j+1}}[u_m]$. First observe that

$$\|q'_{j+1}q_{j+1}\| = \|x_{j+1}\bar{p}_{j+1}\| \le \|x_j\bar{p}_j\| \le j\delta \le (j+1)\delta.$$

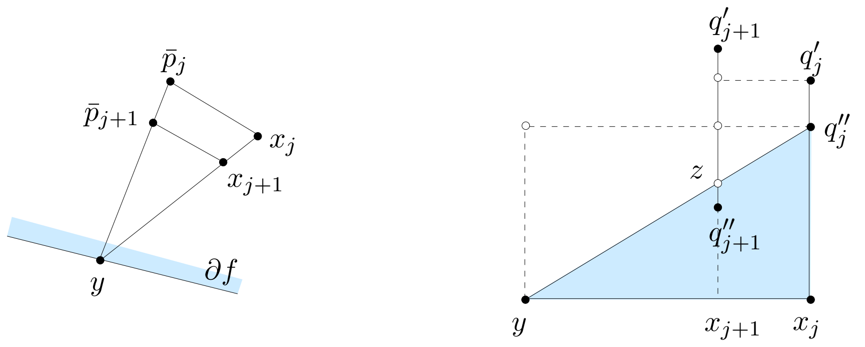


FIGURE 2.7: The inductive step for proving $\|q'_{j+1}q_{j+1}\| \leq (j+1)\delta$ and $q''_{j+1} \in \text{conv}(P)$.

Furthermore, let z be the intersection of yq''_j with $q'_{j+1}x_{j+1}$. We then have

$$\|q'_{j+1}z\| \leq (h_{j+1}-h_j)+\|q'_jq''_j\|+(1/10d^2)\cdot\|q''_jx_j\| < \delta+2(j-m)\delta+\delta = 2(j+1-m)\delta.$$

Therefore q''_{j+1} lies below z and as such $q''_{j+1} \in \triangle x_jyq''_j$. Since $\triangle x_jyq''_j \subseteq \text{conv}(P)$ by the induction hypothesis, we have $q''_{j+1} \in \text{conv}(P)$. □

Lemma 2.11. $\|pq''_k\| \leq \alpha/2$.

Proof. For $j = k$, Lemma 2.10 implies that $\|q'_kq_k\| \leq k\delta$, and $q''_k \in \text{conv}(P)$. It follows that

$$\begin{aligned}\|pq''_k\| &\leq \|pp_k\|+\|p_kq_k\|+\|q_kq'_k\|+\|q'_kq''_k\| \\ &\leq \delta+d\delta+k\delta+2(k-m)\delta \\ &\leq 5d\delta = \alpha/2,\end{aligned}$$

as desired. □

The above lemma concludes the proof of Theorem 2.2.

2.3.3 Remarks

(1) For $d = 2$ and 3, the theorem indicates that $\kappa(P, \alpha/2)$ is only a factor of $O(1)$ and $O(\log(1/\alpha))$, respectively, larger than $\kappa(P, \alpha)$; therefore, the sizes of optimal α-kernels in these dimensions are relative stable. However, for $d \geq 4$, the stability drastically reduces in the worst case because of the superlinear dependency on $\kappa(P, \alpha)$.

(2) Neither the upper nor the lower bound in the theorem is tight. For $d = 3$, we can prove a tighter lower bound of $\Omega\big(\kappa(P, \alpha)\log(1/(\alpha\cdot\kappa(P, \alpha)))\big)$. We conjecture that $\kappa(P, \alpha/2) = \Theta\big(\kappa(P, \alpha)^{\lfloor d/2\rfloor}\log^{d-2}(1/(\alpha^{(d-1)/2}\cdot\kappa(P, \alpha)))\big)$ in $\mathbb{R}^d$.

3 Shape Fitting on Point Sets with Probability Distributions

Motivation

This chapter deals specifically with data sets where each "data point" is actually a distribution of where a data point may be. In this chapter we focus on geometric problems on this data, however, many of the ideas, data structures, and algorithms extend to non-geometric problems.

Since the input data we consider has uncertainty, given by probability distributions, we argue that computing exact answers may not be worth the effort. Furthermore, many problems we consider may not have compact closed form solutions. As a result, we produce approximate answers.

However, as discussed in the introduction, much raw data is not immediately given as a set of probability distributions, rather as a set of points, each drawn from a probability distribution itself. Approximate algorithms may treat this data as exact, construct an approximate answer, and then postulate that since the raw data is not exact and has inaccuracies, the approximation errors made by the algorithm may be similar to the inaccuracies of the imprecise input data. This is a very dangerous postulation, as demonstrated by the following example.

Example. *Consider a robot trying to determine the boundary of a convex room. Its strategy is to use a laser range finder to get data points on objects in the room (hopefully boundary walls), and then take the convex hull of these points.*

However, large errors may occur if the room has windows; a few laser scans may not bounce off the window, and thus return data points (say, 100 *meters) outside the room. Standard techniques (e.g., α-kernels) would include those points in the convex hull, but may allow some approximation (say, up to* 10 *meters). Hence, the outlier data points could still dramatically warp the shape of the room, outside the error tolerance.*

However, an error model on these outlier data points, through regression to the mean, *would assign some probability to them being approximately correct and some probability to them actually being inside (or much closer to) the true room. An algorithm which took this error model into account would assign some probability of a shape near the true room shape and some probability to the oblong room that extends out through the window.*

It is clear from this example, that an algorithm can only provide answers as good as the raw data *and* the models for error on that data. This chapter is not about how to construct error models, but how to take error models into account. While many existing algorithms produce approximations with respect only to the raw input data, algorithms in this chapter approximate with respect to the raw input data and the error models associated with them.

Other error models. In the 1980s and 1990s, many data sets in computational geometry could be assumed to be precise as they we often hand-constructed for computer graphics or simulations. An early model to quantify imprecision in geometric data, motivated by finite precision of coordinates, is *ε-geometry*, introduced by Guibas *et al.* [49]. In this model, the input is given by a traditional point set P, where the imprecision is modeled by a single extra parameter ε. The true point set is not known, but it is certain that for each point in P there is a point in the disk of radius ε around it. (See also the work of Milenkovic [86], Hoffman [62], and Joskowitz *et al.* [67].) This model has proven fruitful and is still used due to its simplicity. To name a few, Guibas *et al.* [50] define *strongly convex* polygons: polygons that are guaranteed to stay convex, even when the vertices are perturbed by ε. (See also [75].) Bandyopadhyay and Snoeyink [16] compute the set of all potential simplices in $\mathbb{R}^2$ and $\mathbb{R}^3$ that could belong to the Delaunay triangulation. Held and Mitchell [59] and Löffler and Snoeyink [78] study the problem of preprocessing a set of imprecise points under this model, so that when the true points are specified later some computation can be done faster. Cabello and van Krevald consider matching of point sets under this model [26].

A more involved model for imprecision can be obtained by not specifying a single ε for all the points, but allowing a different radius for each point, or even other shapes of imprecision regions. This allows for modeling imprecision that comes from different sources, independent imprecision in different dimensions of the input, etc. This extra freedom in modeling comes at the price of more involved algorithmic solutions, but still many results are available. Nagai and Tokura [87] compute the union and intersection of all possible convex hulls to obtain bounds on any possible solution, as does Ostrovsky-Berman and Joskowicz [92] in a setting allowing some dependence between points. Van Kreveld and Löffler [115] study the problem of computing the smallest and largest possible values of several geometric extent measures, such as the diameter or the radius of the smallest enclosing ball, where the points are restricted to lie in given regions in the plane. Kruger [69] extends some of these results to higher dimensions.

These models, in general, give worst case bounds on error, for instance upper and lower bounds on the radius of minimum enclosing ball. When the error is derived entirely from precision errors, this information can be quite useful (as much of theoretical computer science is based on worst case bounds). However, when data is sensed, the maximum error range used as input are often manufactured by truncating a probability distribution, so the probability that a point is outside that range is below some threshold. Since the above models usually produce algorithms and answers very dependent on boundary cases, these artificial (and sometimes arbitrary) thresholds play large roles in the answers. Furthermore, the true location of the data points are often not near the boundary of the error range, but near the center. Hence, it makes more sense to use the original probability distributions, and then if needed, we can apply a threshold based on probability to the final solution. This ensures that the truncation errors have not accumulated.

3.1 Problem Statement

Let $\mu_p : \mathbb{R}^d \to \mathbb{R}^+$ describe the probability distribution of a point p where the integral $\int_{x \in \mathbb{R}^d} \mu_p(x)\, dx = 1$. Let $\mu_P : \mathbb{R}^d \times \mathbb{R}^d \times \ldots \times \mathbb{R}^d \to \mathbb{R}^+$ describe the distribution of a point set P by the joint probability over each $p \in P$. For brevity we write the space $\mathbb{R}^d \times \ldots \times \mathbb{R}^d$ as $\mathbb{R}^{dn}$. For this chapter we will assume $\mu_P(q_1, q_2, \ldots, q_n) = \prod_{i=1}^{n} \mu_{p_i}(q_i)$, so the distribution for each point is independent, although for some of our algorithms this restriction can be easily circumvented.

Given a distribution μ_P we ask a variety of shape fitting questions about the uncertain point set. For instance, we can ask what is the radius smallest enclosing ball or what is the smallest axis-aligned bounding box of an uncertain point set. In the presence of imprecision, the answer to such a question is not a single value or structure, but also a *distribution* of answers. The focus of this chapter is not just how to answer such shape fitting questions about these distributions, but how to concisely represent them. As a result, we introduce two types of approximate distributions as answers, and a technique to construct coresets for these answers.

ε-Quantizations. Let $f : \mathbb{R}^{dn} \to \mathbb{R}^k$ be a function on a fixed point set. Examples include the radius of the minimum enclosing ball where $k = 1$ and the width of the minimum enclosing axis-aligned rectangle in along the x-axis and y-axis where $k = 2$. Define the "dominates" binary operator $\preceq$ so that $(p_1, \ldots, p_k) \preceq (v_1, \ldots, v_k)$ is true if for every coordinate $p_i \le v_i$. Let $\mathbb{X}_f(v) = \{Q \in \mathbb{R}^{dn} \mid f(Q) \preceq v\}$. For a query value v define,

$$F_{\mu_P}(v) = \int_{Q \in \mathbb{X}_f(v)} \mu_P(Q)\, dQ.$$

Then F_{μ_P} is the cumulative density function of the distribution of possible values that f can take[1]. Ideally, we would return the function F_{μ_P} so we could quickly

[1] In this chapter, for a function f and a distribution of point sets μ_P, we will always represent the cumulative density function of f over μ_P by F_{μ_P}.

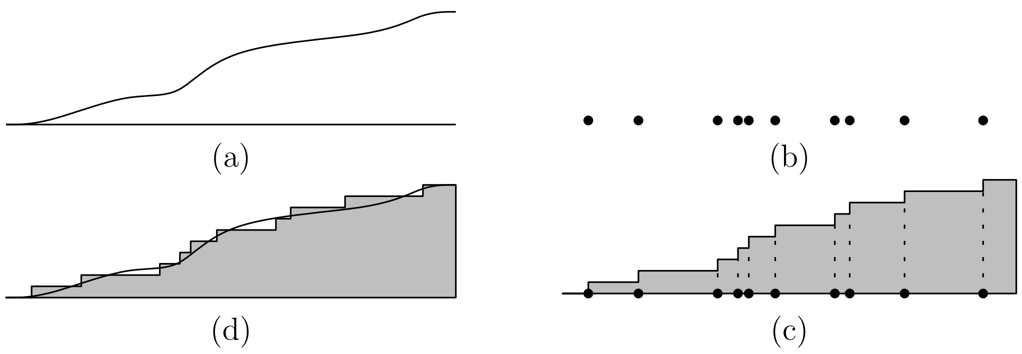


FIGURE 3.1: (a) The true form of the function. (b) The ε-quantization R as a point set in $\mathbb{R}$. (c) The inferred curve h_R in $\mathbb{R}^2$. (d) Overlay of the two images.

answer any query exactly, however, it is not clear how to calculate $F_{\mu_P}(v)$ exactly for even a single query value v. Rather, we introduce a data structure, which we call an ε-quantization, to answer any such query approximately and efficiently, illustrated in Figure 3.1 for $k = 1$. An *ε-quantization* is a point set $R \subset \mathbb{R}^k$ which induces a function h_R where $h_R(v)$ describes the fraction of points in R that v dominates. Let $R_v = \{r \in R \mid r \preceq v\}$. Then $h_R(v) = |R_v|/|R|$. For an isotonic (monotonically increasing in each coordinate) function F_{μ_P} and any value v, an ε-quantization, R, guarantees that

$$|h_R(v) - F_{\mu_P}(v)| \leq \varepsilon.$$

When $k = 1$, we say R is a univariate ε-quantization, otherwise we say R is a k-variate ε-quantization. A 2-variate ε-quantization is illustrated in Figure 3.2. The space required to store the data structure for R is dependent only on ε and k, not on $|P|$ or μ_P.

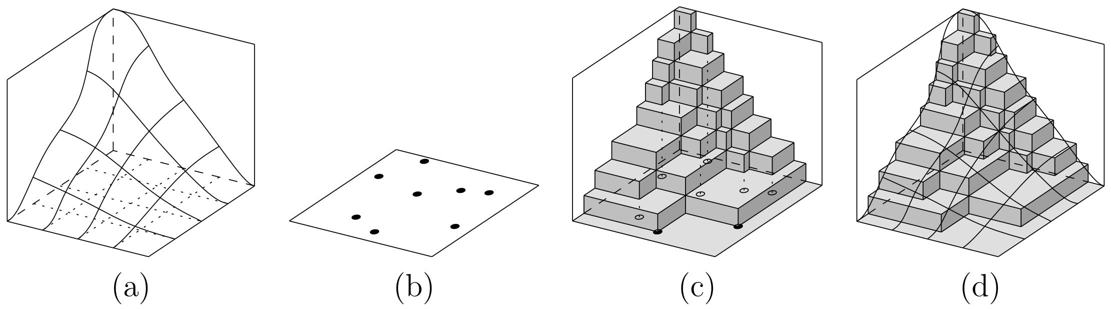


FIGURE 3.2: (a) The true form of the 2-variate function. (b) The ε-quantization R as a point set in $\mathbb{R}^2$. (c) The inferred surface h_R in $\mathbb{R}^3$. (d) Overlay of the two images.

$(\varepsilon, \delta, \alpha)$-**Kernels.** Rather than compute a new data structure for each measure we are interested in, we can also compute a single data structure (a coreset) that allows us to answer many types of questions. For an isotonic function $F_{\mu_P} : \mathbb{R}^+ \to [0,1]$, an (ε, α)-*quantization* data structure M describes a function $h_M : \mathbb{R}^+ \to [0,1]$ so for any $x \in \mathbb{R}^+$, there is an $x' \in \mathbb{R}^+$ such that (1) $|x - x'| \leq \alpha x$ and (2) $|h_M(x) - F_{\mu_P}(x')| \leq \varepsilon$. An $(\varepsilon, \delta, \alpha)$-*kernel* is a data structure that can produce an (ε, α)-quantization, with probability at least $1 - \delta$, for F_{μ_P} where f measures the width in any directions and whose size depends only on $1/\varepsilon$, $1/\alpha$, and $1/\delta$. The notion of (ε, α)-quantizations is generalized to a k-variate version, as are $(\varepsilon, \delta, \alpha)$-kernels, in Section 3.3.

Shape inclusion probabilities. A *summarizing shape* of a point set $P \subset \mathbb{R}^d$ is a Lebesgue-measureable subset of $\mathbb{R}^d$ that is determined by P. Examples include the smallest enclosing ball, the minimum-area axis-aligned bounding rectangle, or the convex hull. We consider some class of shapes $\mathcal{S}$ and the summarizing shape $S(P) \in \mathcal{S}$ is the shape from $\mathcal{S}$ that is optimized in some aspect with respect to P. For a family of summarizing shapes $\mathcal{S}$ we can study the *shape inclusion probability function* $s_{\mu_P} : \mathbb{R}^d \to [0,1]$ (or sip function), where $s_{\mu_P}(q)$ describes the probability that a query point $q \in \mathbb{R}^d$ is included in the summarizing shape[2]. There does not seem to be a closed form for many of these functions. Rather we calculate an ε-sip function $\hat{s} : \mathbb{R}^d \to [0,1]$ such that $\forall_{x \in \mathbb{R}^d} |s_{\mu_P}(x) - \hat{s}(x)| \leq \varepsilon$. The space required to store an ε-sip function depends only on ε and the complexity of the summarizing shape.

3.1.1 Contributions

We describe simple and practical randomized algorithms for the computation of ε-quantizations, $(\varepsilon, \delta, \alpha)$-kernels, and ε-sip functions. Let $T_f(n)$ be the time it takes to calculate a summarizing shape of a set of n points $Q \subset \mathbb{R}^d$, which generates a statistic $f(Q)$ (e.g., radius of smallest enclosing ball). We can calculate an ε-quantization of F_{μ_P}, with probability at least $1 - \delta$, in time $O(T_f(n)(1/\varepsilon^2)\log(1/\delta))$. For univariate ε-quantizations the size is $O(1/\varepsilon)$, and for k-variate ε-quantizations the size is $O(k^2(1/\varepsilon)\log^{2k}(1/\varepsilon))$. We can calculate an $(\varepsilon, \delta, \alpha)$-kernel of size $O((1/\alpha^{(d-1)/2}) \cdot (1/\varepsilon^2)\log(1/\delta))$ in $O((n + (1/\alpha^{d-3/2}))(1/\varepsilon^2)\log(1/\delta))$ time. With probability at least $1 - \delta$, we can calculate an ε-sip function of size $O((1/\varepsilon^2)\log(1/\delta))$ in time $O(T_f(n)(1/\varepsilon^2)\log(1/\delta))$. All of these randomized algorithms are simple and practical, as demonstrated by some experimental results.

In addition, we provide deterministic algorithms for computing ε-quantizations of a specific class of functions. Let $\mathcal{S}$ be a family of summarizing shapes such that $(\mathbb{R}^d, \mathcal{S})$ has bounded VC-dimension. Let $f : \mathbb{R}^{dn} \to \mathbb{R}^k$ be a function that for a point set $P \in \mathbb{R}^{dn}$ describes a statistic on summarizing shape $S(P) \in \mathcal{S}$. An ε-quantization for F_{μ_P} can be computed in deterministic time $O(\text{poly}(n, 1/\varepsilon))$, as described in Table 3.2.

[2] For technical reasons, if there are (degenerately) multiple optimal summarizing shapes, we say each are equally likely to be the summarizing shape of the point set.

This chapter describes results for shape fitting problems for distributions of point sets in $\mathbb{R}^d$, in particular, we will use the smallest enclosing ball and the axis-aligned bounding box as running examples in the algorithm descriptions. The concept of ε-quantizations extends to many other problems with uncertain data. In fact, variations of our randomized algorithm will work for a more general array of problems.

3.2 Randomized Algorithm for ε-Quantizations

We start with a general algorithm (Algorithm 3.2) which will be made specific in several places in this chapter. We assume we can draw a point from μ_p for each $p \in P$ in constant time; if the time depends on some other parameters, the time complexity of the algorithms can be easily adjusted.

Algorithm 3.2.1 Approximate μ_P with regard to a family of shapes $\mathcal{S}$ or function $f_{\mathcal{S}}$

```
1: for $i = 1$ to $m = O((1/\varepsilon^2)\log(1/\delta))$ do
2:     for $p_j \in P$ do
3:         Generate $q_j \in \mu_{p_j}$.
4:     Set $V_i = f_{\mathcal{S}}(\{q_1, q_2, \ldots, q_n\})$.
5: Reduce or Simplify the set $V = \{V_i\}_{i=1}^m$.
```

Algorithm for ε-quantizations. For a function f on a point set P of size n, it takes $T_f(n)$ time to evaluate $f(P)$. We now construct an approximation to F_{μ_P} by adapting Algorithm 3.2.1 as follows. First draw a sample point q_j from each μ_{p_j} for $p_j \in P$, then evaluate $V_i = f(\{q_1, \ldots, q_n\})$. The fraction of trials of this process that produces a value dominated by v is the estimate of $F_{\mu_P}(v)$. In the univariate case we can reduce the size of V by returning $2/\varepsilon$ evenly spaced points according to the sorted order.

Theorem 3.1. *Let $T_f(n)$ be the time it takes to compute $f(Q)$ for any point set Q of size n. For a distribution μ_P of n points, with success probability at least $1-\delta$, there exists an ε-quantization of size $O(1/\varepsilon)$ for F_{μ_P}, and it can be constructed in $O(T_f(n)(1/\varepsilon^2)\log(1/\delta))$ time.*

Proof. Because $F_{\mu_P} : \mathbb{R} \to [0,1]$ is an isotonic function, there exists another function $g : \mathbb{R} \to \mathbb{R}^+$ such that $F_{\mu_P}(t) = \int_{x=-\infty}^{t} g(x)\,dx$ where $\int_{x\in\mathbb{R}} g(x)\,dx = 1$. Thus g is a probability distribution of the values of f given inputs drawn from μ_P. This implies that an ε-sample of $(g, \mathcal{I}_+)$ is an ε-quantization of F_{μ_P}, since both estimate within ε the fraction of points in any range of the form $(-\infty, x)$.

By drawing a random sample q_i from each μ_{p_i} for $p_i \in P$, we are drawing a random point set Q from μ_P. Thus $f(Q)$ is a random sample from g. Hence, using the standard randomized construction for ε-samples, $O((1/\varepsilon^2)\log(1/\delta))$ such samples will generate an $(\varepsilon/2)$-sample for g, and hence an $(\varepsilon/2)$-quantization for F_{μ_P}, with probability at least $1-\delta$.

Since in an $(\varepsilon/2)$-quantization R every value $h_R(v)$ is different from $F_{\mu_P}(v)$ by at most $\varepsilon/2$, then we can take an $(\varepsilon/2)$-quantization of the function described by $h_R(\cdot)$ and still have an ε-quantization of F_{μ_P}. Thus, we can reduce this to an ε-quantization of size $O(1/\varepsilon)$ by taking a subset of $\varepsilon/2$ points spaced evenly according to their sorted order. □

We can construct k-variate ε-quantizations using the same basic procedure as in Algorithm 3.2. The output V_i of $f_{\mathcal{S}}$ is k-variate and thus results in a k-dimensional point. As a result, the reduction of the final size of the point set requires more advanced procedures.

Theorem 3.2. *Let $T_f(n)$ be the time it takes to compute $f(Q)$ for any point set Q of size n. Given a distribution μ_P of n points, with success probability at least $1-\delta$, we can construct a k-variate ε-quantization for F_{μ_P}*

(a) of size $O((k^2/\varepsilon^2)\log(1/\delta))$ and in time $O(T_f(n)(k/\varepsilon^2)\log(1/\delta))$,

(b) of size $O((k^2/\varepsilon)\log^{2k}(1/\varepsilon))$ and in time $O(T_f(n)(k/\varepsilon^2)\log(1/\delta)+(k^2/\varepsilon^5)\log^{6k}(1/\varepsilon)\log(1/\delta))$, or

(c) of size $O((k^2/\varepsilon^2)\log(1/\varepsilon))$ and in time $O(T_f(n)(k/\varepsilon^2)\log(1/\delta)+(k^{3k+1}/\varepsilon^{2k+2})\log^{k}(k/\varepsilon)\log(1/\delta))$.

Proof. Let $\mathcal{R}_+$ describe the family of ranges where a range $A_p=\{q\in\mathbb{R}^k \mid q\preceq p\}$. In the k-variate case there exists a function $g:\mathbb{R}^k\to\mathbb{R}^+$ such that $F_{\mu_P}(v)=\int_{x\preceq v} g(x)\,dx$ where $\int_{x\in\mathbb{R}^k} g(x)\,dx=1$. Thus g describes the probability distribution of the values of f given inputs drawn randomly from μ_P. Hence a random point set Q from μ_P, evaluated as $f(Q)$, is still a random sample from the k-variate distribution described by g. Thus, with probability at least $1-\delta$, a set of $O((k/\varepsilon^2)\log(1/\delta))$ such samples is an ε-sample of $(g,\mathcal{R}_+)$, which has VC-dimension k, and the samples are also a k-variate ε-quantization of F_{μ_P}.

We can then reduce the size of the ε-quantization R to $O((k^2/\varepsilon)\log^{2k}(1/\varepsilon))$ in time $O(|R|(k/\varepsilon^3)\log^{6k}(1/\varepsilon))$ [96] or to $O((k^2/\varepsilon^2)\log(1/\varepsilon))$ in time $O(|R|(k^{3k}/\varepsilon^{2k})\cdot\log^k(k/\varepsilon))$ [32], since the VC-dimension is k and each data point requires $O(k)$ storage. □

There exist weighted variants of ε-quantizations, where each $r\in R$ contributes some value $w(r)$ to the total fraction and $\sum_{r\in R} w(r)=1$. For a weighted ε-quantization (R,w)

$$h_R(v)=\sum_{r\in R_v} w(r).$$

3.3 $(\varepsilon, \delta, \alpha)$-Kernels

The above construction works for a fixed family of summarizing shapes. This section builds a single data structure, an $(\varepsilon, \delta, \alpha)$-kernel, for a distribution μ_P in $\mathbb{R}^d$ that can be used to construct (ε, α)-quantizations for several families of summarizing shapes. In particular, an $(\varepsilon, \delta, \alpha)$-kernel of μ_P is a data structure such that in any query direction $u \in \mathbb{S}^{d-1}$ we can create an (ε, α)-quantization for the cumulative density function of $\omega(\cdot, u)$, the width in direction u, with probability at least $1 - \delta$. This data structure introduces a parameter α, which deals with relative geometric error, in addition to the error parameter ε, which deals with relative counting error and error parameter δ which accounts for potential error due to randomization.

We follow the randomized framework described above as follows. The desired $(\varepsilon, \delta, \alpha)$-kernel $\mathcal{K}$ consisting of a set of $m = O((1/\varepsilon^2)\log(1/\delta))$ $(\alpha/2)$-kernels, $\{K_1, K_2, \ldots, K_m\}$, where each K_j is an $(\alpha/2)$-kernel of a point set Q_j drawn randomly from μ_P. Given $\mathcal{K}$, with probability at least $1 - \delta$ we can then create an (ε, α)-quantization for the cumulative density function of width over μ_P in any direction $u \in \mathbb{S}^{d-1}$. Specifically, let $M = \{\omega(K_j, u)\}_{j=1}^m$.

Lemma 3.1. *With probability at least $1 - \delta$, M is an (ε, α)-quantization for the cumulative density function of width of μ_P in direction u.*

Proof. The width $\omega(Q_j, u)$ of a random point set Q_j drawn from μ_P is a random sample from the distribution over widths of μ_P in direction u. Thus, with probability at least $1 - \delta$, m such random samples would create an ε-quantization. Using the width of the α-kernels K_j instead of Q_j induces an error on each random sample of at most $2\alpha \cdot \omega(Q_j, u)$. Then for a query width w, say there are γm point sets Q_j that have width at most w and $\gamma' m$ α-kernels K_j with width at most w; see Figure 3.3. Note that $\gamma' > \gamma$. Let $\hat{w} = w - 2\alpha w$. For each point set Q_j that has width greater than w it follows that K_j has width greater than $\hat{w}$. Thus the number of α-kernels K_j that have width at most $\hat{w}$ is at most γm, and thus there is a width w' between w and $\hat{w}$ such that the number of α-kernels at most w' is exactly γm. $\square$

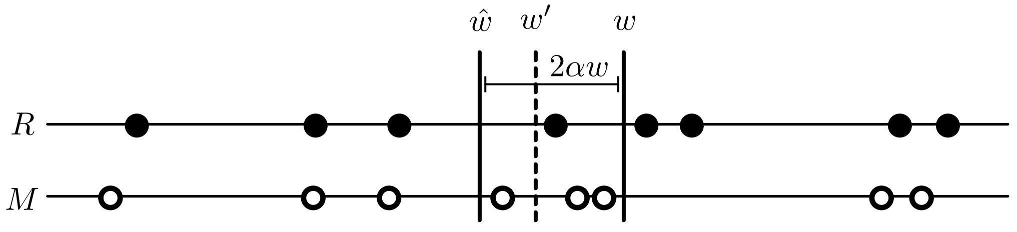


FIGURE 3.3: (ε, α)-quantization M (white circles) and ε-quantization R (black circles) given a query width w.

Since each K_j can be computed in $O(n + 1/\alpha^{d-3/2})$ time, we obtain the following.

Theorem 3.3. *We can construct an $(\varepsilon, \delta, \alpha)$-kernel for μ_P on n points in $\mathbb{R}^d$ of size $O((1/\alpha^{(d-1)/2})(1/\varepsilon^2)\log(1/\delta))$ and in time $O((n + 1/\alpha^{d-3/2}) \cdot (1/\varepsilon^2)\log(1/\delta))$.*

k-Dependent (ε, α)-Kernels. The definition of (ε, α)-quantizations can be extended to a *k-variate (ε, α)-quantizations* data structure with the following properties. A k-variate ε-quantization M is a set of points in $\mathbb{R}^k$ which induces a function $h_M : \mathbb{R}^k \to [0, 1]$ where a query $h_M(x) = |M_x|/|M|$ returns the fraction of points in M which are dominated by or equal to x. Let $x^{(i)}$ represent the ith coordinate of a point $x \in \mathbb{R}^k$. For a query $x \in \mathbb{R}^k$, there exists a point $x' \in \mathbb{R}^k$ such that (1) for all integers $i \in [1, k]$ $|x^{(i)} - (x')^{(i)}| \leq \alpha x^{(i)}$ and (2) $|M(x) - F_{\mu_P}(x')| \leq \varepsilon$.

In addition, $(\varepsilon, \delta, \alpha)$-kernels can be generalized to approximate cumulative density functions of other functions $f : \mathbb{R}^{dn} \to \mathbb{R}^k$, specified as follows. We say a point $p' \in \mathbb{R}^k$ is a *relative θ-approximation* of $p \in \mathbb{R}^k$ if for each coordinate i we have $|p^{(i)} - p'^{(i)}| \leq \theta p^{(i)}$. For a parameter $a \in [0, 1]$, we say that f is *relative $\theta(\alpha)$-approximable* if for all $Q \in \mathbb{R}^{dn}$ and for any α-kernel K of Q, $f(K)$ is a relative $\theta(\alpha)$-approximation of $f(Q)$.

By setting $m = O((k/\varepsilon^2)\log(1/\delta))$ in the above algorithm, we can build a *k-dependent $(\varepsilon, \delta, \alpha)$-kernel* data structure $\mathcal{K}$ with the following properties. It has size $O((1/\alpha^{(d-1)/2})(k/\varepsilon^2)\log(1/\delta))$ and can be built in time $O((n+1/\alpha^{d-3/2})(k/\varepsilon^2) \cdot \log(1/\delta))$. To create a k-variate (ε, α)-quantization for a function f (with probability at least $1 - \delta$), create a k-dimensional point $p_j = f(K_j)$ for each α-kernel K_j in $\mathcal{K}$. The set of m k-dimensional points forms the k-variate (ε, α)-quantization M.

Theorem 3.4. *Given a distribution μ_P of n points in $\mathbb{R}^d$, for $m = O((k/\varepsilon^2)\log(1/\delta))$, we can create a k-dependent $(\varepsilon, \delta, \alpha)$-kernel $\mathcal{K}$ of size $O((1/\alpha^{(d-1)/2})m)$ and in time $O((n + 1/\alpha^{d-3/2})m)$. Let f be any relative $\theta(\alpha)$-approximable function that takes $T_f(N)$ time to evaluate on a set of N points. From $\mathcal{K}$, we can create a k-variate $(\varepsilon, \theta(\alpha))$-quantization of F_{μ_P}*

(a) of size $O((k/\varepsilon^2)\log(1/\delta))$ and in time $O(T_f(1/\alpha^{(d-1)/2})m)$,

(b) of size $O((k/\varepsilon)\log^{2k}(k/\varepsilon))$ and in time $O(T_f(1/\alpha^{(d-1)/2})m + (k/\varepsilon^3)\log^{6k}(1/\varepsilon)m)$, or

(c) of size $O((k/\varepsilon^2)\log(k/\varepsilon))$ and in time $O(T_f(1/\alpha^{(d-1)/2})m + (k^{3k}/\varepsilon^{2k})\log^k(k/\varepsilon)m)$

Proof. Let $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$ be the m points sets drawn randomly from μ_P and for the set $\mathcal{K} = \{K_1, \ldots, K_m\}$ let K_j be the α-kernel of Q_j. Consider the probability distribution g describing the values of $f(Q)$ where Q is drawn randomly from μ_P. The set of m k-dimensional points $\{w_1 = f(Q_1), \ldots, w_m = f(Q_m)\}$ describes an ε-sample of $(g, \mathcal{R}_+)$ and hence also an ε-quantization of F_{μ_P}. We claim the set $\{w'_1 = f(K_1), \ldots, w'_m = f(K_m)\}$ forms an (ε, α)-quantization of F_{μ_P}.

For a query point $w \in \mathbb{R}^k$, let γm point sets from $\mathcal{Q}$ produce a value $w_j = f(Q_j)$ such that $w_j \preceq w$, and let $\gamma' m$ point sets from $\mathcal{K}$ produce a value $w'_j = f(K_j)$ such that $w'_j \preceq w$. Note that $\gamma' > \gamma$. Let $\hat{w} = w - \theta(\alpha)w$; more specifically, for each coordinate $w^{(i)}$ of w, $\hat{w}^{(i)} = w^{(i)} - \theta(\alpha)w^{(i)}$. Because f is relative $\theta(\alpha)$-approximable, for each point set $Q_j \in \mathcal{Q}$ such that $w_j \not\preceq w$, then $w'_j \not\preceq \hat{w}$. Thus, the number of point sets such that $f(K_j) \preceq \hat{w}$ is at most γm, and hence there is a point w' between w and $\hat{w}$ such that the fraction of sampled point sets such that $f(K_j) \preceq w'$ is exactly γ, and hence is within ε of the true fraction of point sets sampled from μ_P with probability at least $1 - \delta$. □

To name a few examples, the width and diameter are relative 2α-approximable functions, thus the results apply directly with $k = 1$. The radius of the minimum enclosing ball is relative 4α-approximable with $k = 1$. The d directional widths of the minimum perimeter or minimum volume axis-aligned rectangle is relative 2α-approximable with $k = d$.

Remark. If an $(\varepsilon, \delta, \alpha)$-kernel is used for one query, it is correct with probability at least $1 - \delta$, and if it is used for another query, it is also correct with probability at least $1 - \delta$. Although there is probably some dependence between these two quantities, it is not easy to prove in general, hence we only claim the probability they are both correct is at least $(1 - \delta)^2$. We can increase this back to $1 - \delta$ for k queries by setting $m = O((k/\varepsilon^2)\log(1/\delta))$, but we need to specify k in advance. If we had a deterministic construction to create an $(\varepsilon, 0, \alpha)$-kernel this would not be a problem, and we could, say, guarantee an (ε, α)-quantization for width in all directions simultaneously. However, this appears to be a much more difficult problem.

Other coresets. In a similar fashion, coresets of a point set distribution μ_P can be formed using other coresets of discrete point sets. For instance, sample $m = O((1/\varepsilon^2)\log(1/\delta))$ points sets $\{P_1, \ldots, P_m\}$ each from μ_P and then store α-samples $\{Q_1 \subseteq P_1, \ldots, Q_m \subseteq P_m\}$ of each. (If we use random sampling in the second set, then not all distributions μ_{p_i} need to be sampled for each P_j in the first round.) This results in an $(\varepsilon, \delta, \alpha)$-sample of μ_P, and can, for example, be used to construct (with probability $1-\delta$) an (ε, α)-quantization for the fraction of points expected to fall in a query disk. Similar constructions can be done for other coresets, such as ε-nets [58], k-center [10, 54], or smallest enclosing ball [25].

3.3.1 Experiments with $(\varepsilon, \delta, \alpha)$-Kernels and ε-Quantizations

We implemented these randomized algorithms for $(\varepsilon, \delta, \alpha)$-kernels and ε-quantizations for diameter (diam), width in a fixed direction (dwid), and radius of the smallest enclosing ℓ_2 ball (seb_2). We used existing code from Hai Yu [119] for α-kernels and Bernd Gärtner [46] for seb_2. For the input set μ_P we generated 5000 points $P \subset \mathbb{R}^3$ on the surface of a cylinder piece with radius 1 and axis length 10. Each point $p \in P$ represented the center of a Gaussian with standard deviation 3. We set $\varepsilon = \delta = .2$ and generated α-kernels of size at most 40 (the existing code did not allow the user to specify a parameter α, only the maximum size). We generated a total of $m = 40$

point sets from μ_P. The $(\varepsilon, \delta, \alpha)$-kernel has a total of 1338 points. We calculated ε-quantizations and (ε, α)-quantizations for diam, dwid, and seb$_2$, each of size 10; see Figure 3.4 and Table 3.1.

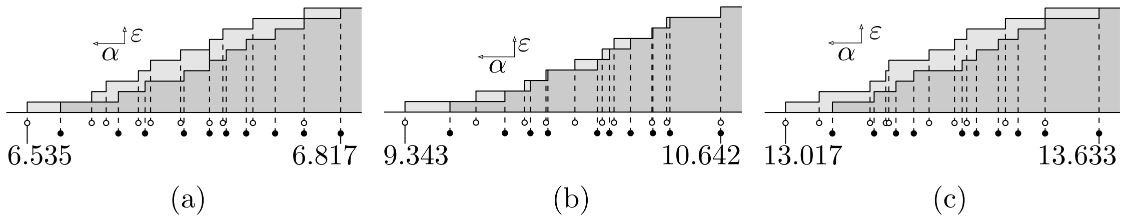


FIGURE 3.4: (ε, α)-quantization (white circles) and ε-quantization (black circles) for (a) seb$_2$, (b) dwid, and (c) diam.

Table 3.1: (ε, α)-quantizations versus ε-quantizations.

func	ap	.1	.2	.3	.4	.5	.6	.7	.8	.9	1.0
seb$_2$	ε	6.56	6.62	6.64	6.68	6.70	6.71	6.73	6.76	6.78	6.82
seb$_2$	$\varepsilon\alpha$	6.53	6.59	6.61	6.63	6.65	6.67	6.69	6.71	6.74	6.78
dwid	ε	9.53	9.75	9.86	9.93	10.13	10.18	10.27	10.36	10.43	10.64
dwid	$\varepsilon\alpha$	9.34	9.63	9.83	9.92	10.04	10.15	10.20	10.36	10.42	10.64
diam	ε	13.11	13.19	13.23	13.27	13.36	13.39	13.43	13.47	13.53	13.63
diam	$\varepsilon\alpha$	13.02	13.08	13.18	13.21	13.22	13.30	13.35	13.37	13.45	13.53

3.4 Shape Inclusion Probabilities

We can also use a variation of Algorithm 3.2 to construct ε-shape inclusion probability functions. For a point set $Q \subset \mathbb{R}^d$, let the summarizing shape $S_Q = \mathcal{S}(Q)$ be from some geometric family $\mathcal{S}$ so $(\mathbb{R}^d, \mathcal{S})$ has bounded VC-dimension ν. We randomly sample m point sets $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$ each from μ_P and then find the summarizing shape $S_{Q_j} = \mathcal{S}(Q_j)$ (e.g. minimum enclosing ball) of each Q_j. Let this set of shapes be $S^{\mathcal{Q}}$. If there are multiple shapes from $\mathcal{S}$ which are equally optimal (as can happen degenerately with, for example, minimum width slabs), choose one of these shapes at random. For a set of shapes $S' \subseteq \mathcal{S}$, let $S'_p \subseteq S'$ be the subset of shapes that contain $p \in \mathbb{R}^d$. We store $S^{\mathcal{Q}}$ and evaluate a query point $p \in \mathbb{R}^d$ by counting what fraction of the shapes the point is contained in, specifically returning $|S^{\mathcal{Q}}_p|/|S^{\mathcal{Q}}|$ in $O(\nu|S^{\mathcal{Q}}|)$ time. In some cases, this evaluation can be sped up with point location data structures.

Theorem 3.5. *Consider a family of summarizing shapes $\mathcal{S}$ where $(\mathbb{R}^d, \mathcal{S})$ has VC-dimension ν and where it takes $T_{\mathcal{S}}(n)$ time to determine the summarizing shape*

$\mathcal{S}(Q)$ for any point set $Q \subset \mathbb{R}^d$ of size n. For a distribution μ_P of a point set of size n, with probability at least $1-\delta$, we can construct an ε-sip function of size $O(2^{\nu+1}(\nu/\varepsilon^2)\log(1/\delta))$ and in time $O(T_{\mathcal{S}}(n)(1/\varepsilon^2)\log(1/\delta))$.

Proof. If $(\mathbb{R}^d, \mathcal{S})$ has VC-dimension ν, then the dual range space $(\mathcal{S}, P^*)$ has VC-dimension $\nu' \leq 2^{\nu+1}$, where P^* is all subsets $\mathcal{S}_p \subseteq \mathcal{S}$, for any $p \in \mathbb{R}^d$, such that $\mathcal{S}_p = \{S \in \mathcal{S} \mid p \in S\}$. Using the above algorithm, sample $m = O((\nu'/\varepsilon^2)\log(1/\delta))$ point sets Q from μ_P and generate the m summarizing shapes S_Q. Each shape is a random sample from $\mathcal{S}$ according to μ_P, and thus $S^{\mathcal{Q}}$ is an ε-sample of $(\mathcal{S}, P^*)$.

Let $w_{\mu_P}(S)$, for $S \in \mathcal{S}$, be the probability that S is the summarizing shape of a point set Q drawn randomly from μ_P. Let $W_{\mu_P}(\mathcal{S}') = \int_{S \in \mathcal{S}'} w_{\mu_P}(S)$, where $\mathcal{S}' \subseteq P^*$, be the probability that some shape from the subset $\mathcal{S}'$ is the summarizing shape of Q drawn from μ_P.

We approximate the sip function at $p \in \mathbb{R}^d$ by returning the fraction $|S_p^{\mathcal{Q}}|/m$. The true answer to the sip function at $p \in \mathbb{R}^d$ is $W_{\mu_P}(\mathcal{S}_p)$. Since $S^{\mathcal{Q}}$ is an ε-sample of $(\mathcal{S}, P^*)$, then with probability at least $1-\delta$

$$\left| \frac{|S_p^{\mathcal{Q}}|}{m} - \frac{W_{\mu_P}(\mathcal{S}_p)}{1} \right| = \left| \frac{|S_p^{\mathcal{Q}}|}{|S^{\mathcal{Q}}|} - \frac{W_{\mu_P}(\mathcal{S}_p)}{W_{\mu_P}(P^*)} \right| \leq \varepsilon.$$

Since for the family of summarizing shapes $\mathcal{S}$ the range space $(\mathbb{R}^d, \mathcal{S})$ has VC-dimension ν, each can be stored using that much space. □

Using deterministic techniques the size can then be reduced to $O(2^{\nu+1}(\nu/\varepsilon^2) \cdot \log(1/\varepsilon))$ in time $O((2^{3(\nu+1)}(\nu/\varepsilon^2)\log(1/\varepsilon))^{2^{\nu+1}} \cdot 2^{3(\nu+1)}(\nu/\varepsilon^2)\log(1/\delta))$.

3.4.1 Representing ε-sip Functions by Isolines.

Shape inclusion probability functions are density functions. One convenient way of visually representing a density function in $\mathbb{R}^2$ is by drawing the isolines. A *γ-isoline* is a closed curve such that on the inside the density function is $> \gamma$ and on the outside is $< \gamma$.

In each part of Figure 3.5 and Figure 3.6 a set of 5 circles correspond to points with a probability distribution. For part (a) of both figures, the probability distribution is uniform over the inside of the circles, in part (b) of both figures it is drawn from a multivariate Gaussian distribution with standard deviation as the radius. We generate ε-sip functions for smallest enclosing ball in Figure 3.5 and for smallest axis-aligned bounding box in Figure 3.6.

In all figures we draw approximations of $\{.9, .7, .5, .3, .1\}$-isolines. These drawing are generated by randomly selecting $m = 5000$ (Figure 3.5) or $m = 25000$ (Figure 3.6) shapes, counting the number of inclusions at different points in the plane and interpolating to get the isolines. The innermost and darkest region has probability $> 90\%$, the next one probability $> 70\%$, etc., the outermost region has probability $< 10\%$.

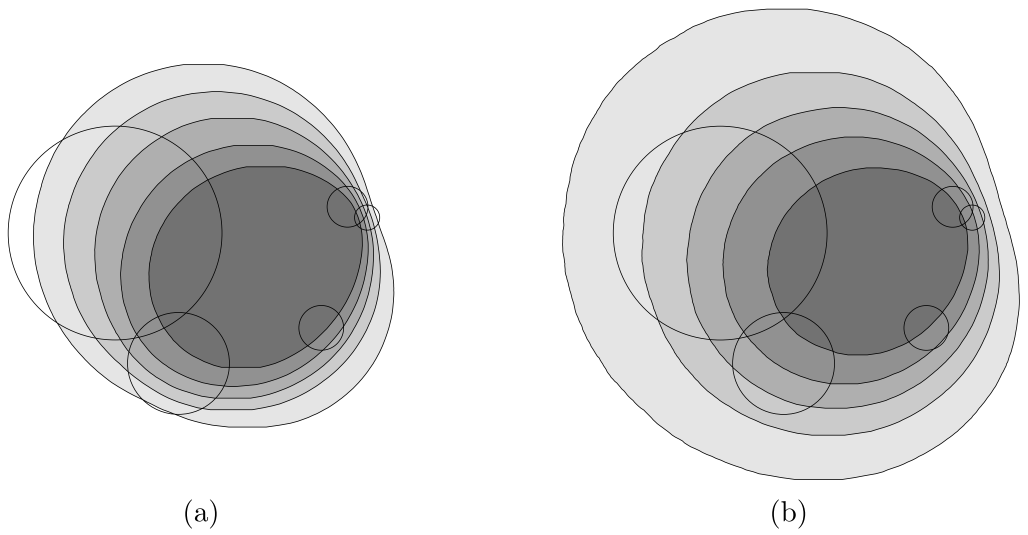


FIGURE 3.5: (a) The shape inclusion probability for the smallest enclosing ball, for points uniformly distributed inside the circles. (b) The same, but for normally distributed points around the circle centers, with standard deviations given by the radii.

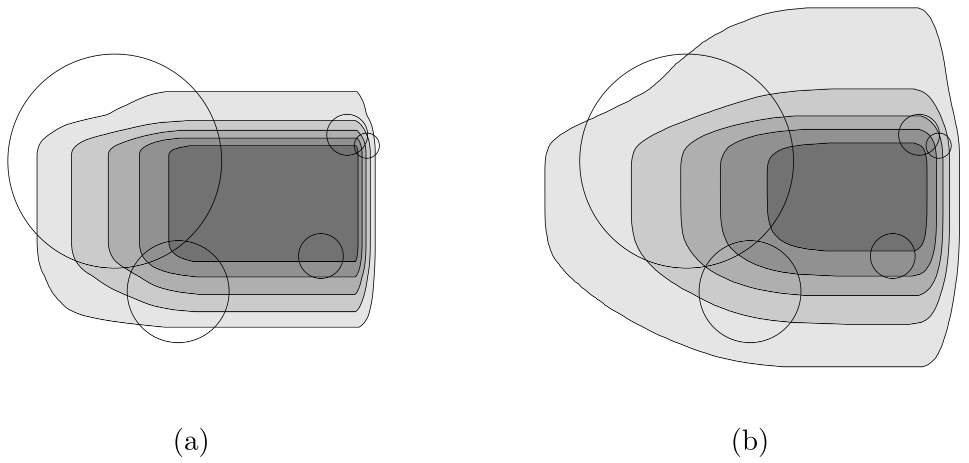


FIGURE 3.6: (a) The shape inclusion probability for the smallest enclosing axis-aligned rectangle, for points uniformly distributed inside the circles. (b) The same, but for normally distributed points.

A center point for μ_P. We can create a point $\bar{q} \in \mathbb{R}^d$ that is in the convex hull of a sampled point set Q from μ_P with high probability. This implies that for any summarizing shape that contains the convex hull, $\bar{q}$ is also contained in that summarizing shape. For a point set $P \subset \mathbb{R}^d$, a *β-center point* is a point $q \in \mathbb{R}^d$, such that any closed halfspace that contains q also contains at least $1/\beta$ fraction points of all points in P. It is known that for any discrete point set a $(d+1)$-center point always exists [105]. Let $\mathcal{H}$ be the family of subsets defined by halfspaces. For a point set P of size n, a $(2d+2)$-center point can be created in $O(d^{5d+3} \log^d d)$ time [34] by first creating an $(1/(2d+2))$-sample of $(P, \mathcal{H})$, and then running a brute force algorithm. Because the first step is creating an ε-sample, this can be extended to Lebesgue-measureable sets such as probability distributions as well.

We use the following algorithm:

1. Create a $(2d+2)$-center point $\bar{p}_i$ for each μ_{p_i}. Let the set be $\bar{P}$.

2. Create $(2d+2)$-center point $\bar{q}$ of $\bar{P}$.

For d constant, the algorithm runs in $O(n)$ time because we can create $(2d+2)$-center points a total of $n+1$ times, and each takes $O(1)$ time.

Lemma 3.2. *Given a distribution of a point set μ_P (such that each point distribution is polygonally approximable) of n points in $\mathbb{R}^d$, there is an $O(n)$ time algorithm to create a point $\bar{q}$ that will be in the convex hull of a point set drawn from μ_P with probability at least $1 - (e^{1/(2d+2)^2})^n$.*

Proof. Because $\bar{p}_i$ is a $(2d+2)$-center point of μ_{p_i}, any halfspace that contains $\bar{p}_i$ on its boundary (and does not contain $\bar{q}$) has probability at least $1/(2d+2)$ of containing a point randomly drawn from μ_{p_i}. Also, because $\bar{q}$ is a $(2d+2)$-center point of $\bar{P}$, for any direction $u \in \mathbb{S}^{d-1}$ there are at least $n/(2d+2)$ points $\bar{p}_i$ from $\bar{P}$ for which $\langle q, u \rangle \le \langle \bar{p}_i, u \rangle$. Thus, if a point q_i is drawn from μ_{p_i} such that $\langle q, u \rangle \le \langle \bar{p}_i, u \rangle$ then the probability that $\langle \bar{q}, u \rangle \le \langle q_i, u \rangle$ is at least $1/(2d+2)$. Hence, the probability that there is a separating halfspace between $\bar{q}$ and the convex hull of Q (where the halfspace is orthogonal to some direction u) is at most

$$(1 - 1/(2d+2))^{n/(2d+2)} = ((1 - 1/(2d+2))^{1/(2d+2)})^n \le (e^{1/(2d+2)^2})^n.$$

□

Theorem 3.6. *For a set of $m < n$ point sets drawn i.i.d. from μ_P, it follows that $\bar{q}$ is in each of the m convex hulls for each point sets with high probability (specifically with probability $\ge 1 - m(e^{1/(2d+2)^2})^n$).*

Proof. Let $\beta = e^{1/(2d+2)^2}$. For any one point set the probability that $\bar{q}$ is contained in the convex hull is at least $1 - \beta^n$. By the union bound, the probability that it is contained in all m convex hulls is at least $(1-\beta^n)^m = 1 - m\beta^n + \binom{m}{2}\beta^{2n} - \binom{m}{3}\beta^{3n} + \ldots$. Since $n > m$, the sum of all terms after the first two in the expansion increase the probability. □

We say a family of shapes $\mathcal{S}$ is *convex* if $S(P) \in \mathcal{S}$ contains the convex hull of P and $S(P)$ is always a convex set. When $\mathcal{S}$ is convex, then for any point q, the line segment $\overline{q\bar{q}}$ is completely contained in $S(P)$ if and only if $q \in S(P)$. Thus, given a set of m summarizing shapes, for every boundary of a summarizing shape $\overline{q\bar{q}}$ crosses, q is outside that summarizing shape. This implies the following corollary.

Corollary 3.1. *Consider a distribution μ_P of point sets of size n, a convex family of shapes $\mathcal{S}$ inducing a* **sip** *function $s_{\mathcal{S}}$ on μ_P, and a positive integer $m < n$. For $\gamma \leq 1-1/m$ the subset of $\mathbb{R}^d$ inside of the γ-isoline of $s_{\mathcal{S}}$, exists, is connected, and is star-shaped with high probability, specifically with probability at least $1 - m(e^{1/(2d+2)^2})^n$.*

3.5 Deterministic Constructions of ε-Quantizations

In this section we consider a family of shapes $\mathcal{S}$ which describe Lebesgue-measureable subset of $\mathbb{R}^d$, so that $(\mathbb{R}^d, \mathcal{S})$ has bounded VC-dimension, and for a point set $P \subset \mathbb{R}^d$, the summarizing shape of $\mathcal{S}(P)$ minimizes some quantity $f(P)$ of the point set P. In particular, we will focus on two examples. First, the seb_2 case, let $\mathcal{S}$ describe the set of discs in $\mathbb{R}^2$, $\mathcal{S}(P)$ is the smallest enclosing disc of P, and $f(P)$ is the radius of the smallest enclosing disc of P. Second, the aabbv case, let $\mathcal{S}$ describe the set of axis-aligned bounding boxes in $\mathbb{R}^d$, $\mathcal{S}(P)$ is the minimum volume axis-aligned bounding box of P, and $f(P)$ is its volume. We are concerned with $\mu_P = \{\mu_{p_1} \times \ldots \times \mu_{p_n}\}$, a distribution on point sets of size n, where for each μ_{p_i} we can deterministically construct an ε-sample of $(\mu_{p_i}, \mathcal{A})$ using the results of Chapter 1.

The overall strategy will be to deterministically create a specific ε-sample Q_{p_i} to represent each μ_{p_i}. Let $Q_P = \{Q_{p_1}, \ldots, Q_{p_n}\}$ describe this set of point sets. Then let the function $\tilde{f}(Q_P, r)$ describe the fraction of point sets $Q' = (q_1 \in Q_{p_1}, q_2 \in Q_{p_2}, \ldots, q_n \in Q_{p_n})$ such that $f(Q') \leq r$. We prove two results: we show how to generate a set of point sets Q_P such that

$$\left| \tilde{f}(Q_P, r) - F_{\mu_P}(r) \right| \leq \varepsilon$$

for all $r \in \mathbb{R}^+$ and we show how to efficiently evaluate $\tilde{f}(Q_P, r)$.

3.5.1 Approximating μ_p

Let $\mathcal{A}_{f,n}$ describe a family of Lebesgue-measureable sets defined by $n-1$ points $T \subset \mathbb{R}^d$ and a value w. Specifically, $A(T, w) \in \mathcal{A}_{f,n}$ is the set of points $\{p \in \mathbb{R}^d \mid f(T \cup p) \leq w\}$. Figure 3.7(a) shows an example element of $\mathcal{A}_{f,8}$ for the seb_2 case. Figure 3.8(b) shows an example element of $\mathcal{A}_{f,8}$ for the aabbv case.

Theorem 3.7. *Let $\mu_{p_1} \times \ldots \times \mu_{p_n} = \mu_P$ describe the distribution of a point set of size n. For $1 \leq i \leq n$, let Q_{p_i} be an ε'-sample of $(\mu_{p_i}, \mathcal{A}_{f,n})$, then for any r*

$$\left| F_{\mu_P}(r) - \tilde{f}(\{Q_{p_1}, Q_{p_2}, \ldots, Q_{p_n}\}, r) \right| \leq \varepsilon' n.$$

Proof. When P is drawn from a distribution μ_P, then we can write $F_{\mu_P}(r)$ as the probability that $f(P) \leq r$ as follows. Let $1(\cdot)$ be the indicator function, i.e., it is 1 when the condition is true and 0 otherwise.

$$F_{\mu_P}(r) = \int_{q_1} \mu_{p_1}(q_1) \dots \int_{q_n} \mu_{p_n}(q_n) 1(f(\{q_1, q_2, \dots, q_n\}) \leq r) \; dq_n dq_{n-1} \dots dq_1$$

Consider the inner most integral

$$\int_{q_n} \mu_{p_n}(q_n) 1(f(\{q_1, q_2, \dots, q_n\}) \leq r) \; dq_n,$$

where $\{q_1, q_2 \dots, q_{n-1}\}$ are fixed. The indicator function is true when for q_n

$$f(\{q_1, q_2, \dots, q_{n-1}, q_n\}) \leq r$$

and hence q_n is contained in a shape $A(\{q_1, \dots, q_{n-1}\}, r) \in \mathcal{A}_{f,n}$. Thus if we have an ε'-sample Q_{p_n} for $(\mu_{p_n}, \mathcal{A}_{f,n})$, then we can guarantee that

$$\begin{aligned} &\int_{q_n} \mu_{p_n}(q_n) 1(f(\{q_1, q_2, \dots, q_n\}) \leq r) \; dq_n \\ &\leq \frac{1}{|Q_{p_n}|} \sum_{q_n \in Q_{p_n}} 1(f(\{q_1, q_2, \dots, q_{n-1}, q_n\}) \leq r) + \varepsilon'. \end{aligned}$$

We can then move the ε' to the outside, and we can change the order of the integrals to write:

$$\begin{aligned} F_{\mu_P}(r) \; &\leq \frac{1}{|Q_{p_n}|} \sum_{q_n \in Q_{p_n}} \\ &\left(\int_{q_1} \mu_{p_1}(q_1) \dots \int_{q_{n-1}} \mu_{p_{n-1}}(q_{n-1}) 1(f(\{q_1, q_2, \dots, q_n\}) \leq r) \; dq_{n-1} \dots dq_1 \right) + \varepsilon'. \end{aligned}$$

Repeating this procedure n times we get:

$$\begin{aligned} F_{\mu_P}(r) \; &\leq \; \left(\prod_{i=1}^{n} \frac{1}{|Q_{p_i}|} \right) \sum_{q_1 \in Q_{p_1}} \cdots \sum_{q_n \in Q_{p_n}} 1(f(\{q_1, q_2, \dots, q_n\}) \leq r) + \varepsilon' n. \\ &= \; \tilde{f}(Q_P, r) + \varepsilon' n. \end{aligned}$$

Using the same technique we can achieve a symmetric lower bound for $F_{\mu_P}(r)$. □

By setting $\varepsilon' = \varepsilon/n$ we can achieve an additive ε-approximation by using an ε'-sample for each $(\mu_{p_i}, \mathcal{A}_{f,n})$.

In the aabbv case, $(\mu_{p_i}, \mathcal{A}_{f,n})$ has constant VC-dimension. Shapes from $\mathcal{A}_{f,n}$ are determined by the placement of $2d$ points, the most extreme in each axis direction,

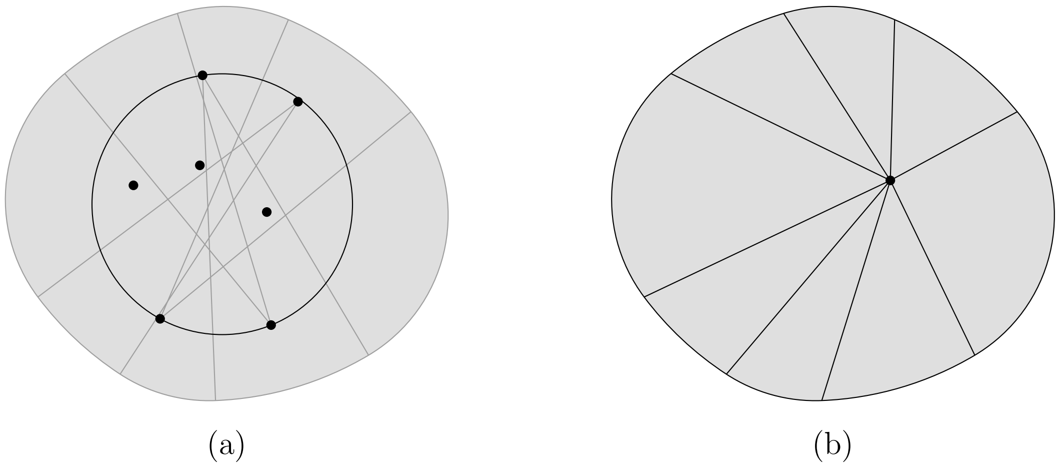


FIGURE 3.7: (a) A shape from $\mathcal{A}_{f,n}$ for smallest enclosing ball using the L_2 metric in $\mathbb{R}^2$. The curves are circular arcs of two different radii. (b) The same shape divided into wedges from $\mathcal{W}_{f,n}$.

thus its shatter dimension is $\sigma_f = 2d$. Hence an ε-sample for $(\mu_{p_i}, \mathcal{A}_{f,n})$ of size $O((1/\varepsilon^2)\log(1/\varepsilon))$ can be calculated in time $O((1/\varepsilon^2)\log^2(1/\varepsilon))$ for each μ_{p_i}.

In the seb_2 case, $(\mathbb{R}^2, \mathcal{A}_{f,n})$ has VC-dimension $\Omega(n)$; for each $A(T,w) \in \mathcal{A}_{f,n}$, the boundary may be described by up to $2n-2$ circular arcs of radius w or $2w$. Naive techniques would take time exponential in n to deterministically create an ε-sample, but we can do better by decomposing shapes from $\mathcal{A}_{f,n}$ into $O(n)$ disjoint shapes; See Figure 3.7(b). Let $\mathcal{W}_f$ be the family of shapes, *wedges*, formed by the intersection of a disc and two halfspaces. We can decompose a shape $A(T,w) \in \mathcal{A}_{f,n}$ into wedges by choosing a point q in the convex hull of T, and then for each circular arc defining a boundary piece of $A(T,w)$ define a wedge using the disc which has that circular arc on its boundary and the halfspaces with boundaries that go through the boundary of the circular arc and q. The VC-dimension of $(\mathbb{R}^2, \mathcal{W}_f)$ is at most 9 because it is described by the intersection of three shapes from families that would each have VC-dimension 3 in a range space with the same ground set. Thus an $(\varepsilon/2n)$-sample of $(\mu_{p_i}, \mathcal{W}_f)$ is an ε-sample of $(\mu_{p_i}, \mathcal{A}_{f,n})$, and it can be constructed of size $O((n^2/\varepsilon^2)\log(n/\varepsilon))$ in time $O((n^2/\varepsilon^2)\log^2(n/\varepsilon))$ for each μ_{p_i}.

We generalize this machinery to other summarizing shapes and in higher dimensions in Appendix 3.A. There are illustrations akin to Figure 3.7(a) for each family of shapes.

3.5.2 Evaluating $\tilde{f}(Q_P, r)$.

Evaluating $\tilde{f}(Q_P, r)$ in time polynomial in n and $|Q_{p_i}|$, for any i, is not completely trivial since there are $n^{|Q_{p_i}|}$ possible sets in Q_P. Let a *panchromatic set* be a set of n points, $G \in Q_1 \times \ldots \times Q_n$. For each panchromatic set G there exists a unique basis $B_G \subseteq G$ of at most $\sigma_{\mathcal{S}}$ points[3] which define the summarizing shape of G, i.e., $\mathcal{S}(G) = \mathcal{S}(B_G)$ (remember $\sigma_{\mathcal{S}}$ is the shatter dimension of $(\mathbb{R}^d, \mathcal{S})$ and $\sigma_{\mathcal{S}} < \nu_{\mathcal{S}}$, the VC-dimension). Define a *valid basis* to be a set of at most $\sigma_{\mathcal{S}}$ points in Q_P such that each point is from a different Q_{p_i} and if any point is removed the summarizing shape changes. Each valid basis may form a basis for several panchromatic sets.

We now construct (R, w), a weighted ε-quantization of F_{μ_P}. This approximation is created by calculating the summarizing shape for all panchromatic sets. Even though there are an exponential number of panchromatic sets, there are only a polynomial number of valid bases. Thus for each valid basis, we count the number of panchromatic sets it represents. And we let each valid basis B contribute to the ε-quantization; its position is determined by $f(B)$ and its weight by the number of panchromatic sets it represents. We initially store the ε-quantization as a sorted list of tuples (r, ξ) where $r = f(\{q_1, q_2, \ldots, q_{\sigma_{\mathcal{S}}}\})$ for some valid basis $\{q_1, q_2, \ldots, q_{\sigma_{\mathcal{S}}}\}$, and ξ is the fraction of the panchromatic sets which are represented by this valid basis. The details are outlined in Algorithm 3.5.1.

Algorithm 3.5.1 Construct ε-Quantization from Q_P

```
1: for all valid bases $q_1, q_2, \ldots, q_{\sigma_{\mathcal{S}}} \in Q_P$ do
2:   for $i = 1$ to $n$ do
3:     if $q_1 \in Q_i$ or $q_2 \in Q_i$ or ... or $q_{\sigma_{\mathcal{S}}} \in Q_i$ then
4:       Set $w_i = 1/|Q_i|$.
5:     else
6:       Set $w_i = (1/|Q_i|) \sum_{q_j \in Q_i} 1(q_j \in \mathcal{S}(\{q_1, q_2, \ldots, q_{\sigma_{\mathcal{S}}}\}))$
7:   Insert $(f(q_1, q_2, \ldots, q_{\sigma_{\mathcal{S}}}), \prod_i w_i)$ into $R$.
```

We now summarize the full deterministic algorithm. For $1 \le i \le n$ for $(\mu_{p_i}, \mathcal{A}_{f.n})$ we create an (ε/n)-sample Q_{p_i} of size $\lambda_f(n, \varepsilon)$. This makes the set Q_P have $\eta = \sum_{i=1}^n |Q_{p_i}| = n\lambda_f(n, \varepsilon)$ points in its sets. We examine $O((\lambda_f(n, \varepsilon))^{\sigma_{\mathcal{S}}})$ valid bases. For each valid basis we evaluate $f(G)$ and then we compute ξ in $\mathsf{RS}_f(n, \varepsilon)$ time using a range searching data structure, after preprocessing or with a naive search. Thus the deterministic running time for constructing an ε-quantization is $O((\lambda_f(n, \varepsilon))^{\nu_{\mathcal{S}}} \cdot \mathsf{RS}_f(n, \varepsilon))$ which is presented for various summarizing shapes in Table 3.4. For instance, for the aabbv case this takes $O(n^{4d+1}/\varepsilon^{4d} \log^{3d-1}(n/\varepsilon))$ time and for the seb_2 case this takes $O(n^{15}/\varepsilon^7 \log^{3.5}(n/\varepsilon))$ time. The total construction time for the ε-quantizations is the sum of this time and the time to construct n (ε/n)-samples of $(\mathbb{R}^d, \mathcal{A}_{f,n})$; for both the aabbv case and the seb_2 case it is the former.

A univariate ε-quantization can be reduced to size $O(1/\varepsilon)$. We can create k-variate ε-quantizations using the same procedure (such as the width in the k dimensions of an axis-aligned bounding box). Thus the same argument applies when $f : \mathbb{R}^{dn} \to \mathbb{R}^k$, and we can create k-variate ε-quantizations of size $(k^2/\varepsilon) \log^{O(k)}(k/\varepsilon)$ in the same deterministic times as long as $\nu_{\mathcal{S}} = O(k)$.

[3] This uniqueness requires careful construction of the ε-samples Q_{p_i}, as described in Section 1.3.1.

Theorem 3.8. *Consider a family of shapes $\mathcal{S}$, where for a point set P the statistic $f(P)$ optimizes a quantity of $\mathcal{S}(P)$. Let $(\mathbb{R}^d, \mathcal{S})$ have VC-dimension $\nu_{\mathcal{S}}$. Let $\mu_P = \mu_{p_1} \times \ldots \times \mu_{p_n}$ describe the distribution of n points. For $1 \leq i \leq n$, let Q_{p_i} be an (ε/n)-sample for $(\mu_{p_i}, \mathcal{A}_{f,n})$ of size $\lambda_f(n, \varepsilon)$. Given a set of m points, let $\mathsf{RS}(m, \mathcal{S})$ describe the time required to count the number of points in a shape from the family $\mathcal{S}$ using near-linear in m space and preprocessing time. We can construct a k-variate ε-quantization of F_{μ_P} of size $(k^2/\varepsilon)\log^{O(k)}(k/\varepsilon)$ in time $O((\lambda_f(n,\varepsilon))^{\nu_{\mathcal{S}}} \cdot n\mathsf{RS}(\lambda_f(n,\varepsilon), \mathcal{S}))$.*

Appendix to Chapter 3

3.A Shapes of $\mathcal{A}_{f,n}$ for Various Summarizing Shapes

Recall that we are considering a family of shapes $\mathcal{S}$ (Lebesgue-measureable subsets of $\mathbb{R}^d$) where for a point set P, $\mathcal{S}(P)$ is a summarizing shape of P and $f(P)$ is a function on $\mathcal{S}(P)$. We let $(\mathbb{R}^d, \mathcal{S})$ have VC-dimension $\nu_{\mathcal{S}}$. Recall that $\mathcal{A}_{f,n}$ is the family of shapes defined with respect to a point set T of size $n-1$ and a value w such that $A(T,w) \in \mathcal{A}_{f,n}$ is defined $\{p \in \mathbb{R}^d \mid f(T \cup p) \leq w\}$. Since $(\mathbb{R}^d, \mathcal{S})$ has VC-dimension $\nu_{\mathcal{S}}$, then any shape $\mathcal{S}(P)$ is completely determined by a subset $B_P \subset P$ of at most $\nu_{\mathcal{S}}$ points. In some cases (e.g., aabbv), we will be able to show $(\mathbb{R}^d, \mathcal{A}_{f,n})$ has constant VC-dimension. In other cases (e.g., seb_2) we will not, and then will need to show that we can decompose any shape $A \in \mathcal{A}_{f,n}$ into a disjoint set of wedges from some family of shapes $\mathcal{W}_f$.

Lemma 3.3. *If the disjoint union of m shapes from $\mathcal{W}_f$ can form any shape from $\mathcal{A}_{f,n}$, then an (ε/m)-sample of $(\mu_p, \mathcal{W}_f)$ is an ε-sample of $(\mu_p, \mathcal{A}_{f,n})$.*

Proof. For any shape $A \in \mathcal{A}_{f,n}$ we can create a set of m shapes $\{W_1, \ldots, W_n\} \subset \mathcal{W}_f$ whose disjoint union is A. Since each range of $\mathcal{W}_f$ may have error ε/m, their union has error at most ε. $\square$

The VC-dimension for $(\mathbb{R}^2, \mathcal{W}_f)$ is shown for several functions in Table 3.3.

Table 3.2: Runtimes for ε-Quantizations of Various Summarizing Shape Families.

case	randomized*	determ. $\mathbb{R}^2$	determ. $\mathbb{R}^d$
dwid	$O(n/\varepsilon^2)$	$\tilde{O}(n^3/\varepsilon^2)$	$\tilde{O}(n^3/\varepsilon^2)$
aabbp	$O(n/\varepsilon^2)$	$\tilde{O}(n^5/\varepsilon^4)$	$\tilde{O}(n^{4d+1}/\varepsilon^{4d})$
aabbv	$O(n/\varepsilon^2)$	$\tilde{O}(n^9/\varepsilon^8)$	$\tilde{O}(n^{4d+1}/\varepsilon^{4d})$
seb_∞	$O(n/\varepsilon^2)$	$\tilde{O}(n^4/\varepsilon^3)$	$\tilde{O}(n^{d+2}/\varepsilon^{d+1})$
seb_1	$O(n/\varepsilon^2)$	$\tilde{O}(n^4/\varepsilon^3)$	$\tilde{O}(n^{d+2}/\varepsilon^{d+1})$
seb_2	$O(n/\varepsilon^2)$	$\tilde{O}(n^{15}/\varepsilon^7)$	
diam	$O(n^2/\varepsilon^2)$	$\tilde{O}(n(n^4/\varepsilon^2)^{n+1})$	

* all randomized results are correct with constant probability.
$\tilde{O}(f(n,\varepsilon))$ ignores poly-logarithmic factors $(\log \frac{n}{\varepsilon})^{O(\mathrm{poly}(d))}$, for any $\tau > 0$.

3.A.1 Examples

We study several example cases for which we can deterministically compute ε-quantizations. For each case we show an example element of $\mathcal{A}_{f,n}$ on an example of 7 points.

Table 3.3: VC-dimension for Various Shape Families.

case	$(\mathbb{R}^2, \mathcal{A}_{f,n})$	$(\mathbb{R}^d, \mathcal{A}_{f,n})$	$(\mathbb{R}^2, \mathcal{W}_f)$
dwid	2	2	
aabbp	$O(1)$ $(\sigma = 4)$	$O(d \log d)$ $(\sigma = 2d)$	
aabbv	8	$O(d \log d)$ $(\sigma = 2d)$	
seb$_\infty$	4	$2d$	
seb$_1$	4	$2d$	
seb$_2$	∞	∞	9
diam	∞	∞	9

Directional Width. We first consider a family of shapes $\mathcal{S}$ which describe all slabs, the intersection of two parallel halfspaces, for a given normal direction u. Given a point set P, $S(P) \in \mathcal{S}$ is the minimum width slab containing P and $f(P)$ is the width of that slab (the dwid case). This can be thought of as a one-dimensional problem by projecting all points P using the operation $\langle \cdot, u \rangle$. The directional width is then just the difference between the largest point and the smallest point. As such, the VC-dimension of $(\mathbb{R}^d, \mathcal{S})$ is 2. Furthermore, $\mathcal{A}_{f,n} = \mathcal{S}$ in the dwid case, so $(\mathbb{R}^d, \mathcal{A}_{f,n})$ also has VC-dimension 2. For $1 \leq i \leq n$, we can then create an (ε/n)-sample Q_i of $(\mu_{p_i}, \mathcal{A}_{f,n})$ of size $\lambda_f(n, \varepsilon) = O(n/\varepsilon)$ in $O((n/\varepsilon)\log(n/\varepsilon))$ time given basic knowledge of the distribution μ_{p_i}. Range searching over each Q_i can be done in $O(\log(n/\varepsilon))$ time, so invoking Theorem 3.8, it takes $O((n^3/\varepsilon^2)\log(n/\varepsilon))$ time to construct an ε-sample of F_{μ_P} in the dwid case.

A separate manuscript addresses the specific dwid case in more detail [97]. The runtime can be improved to $O(n^{3+\tau}/\varepsilon)$ for any $\tau > 0$. Or in the case where each μ_{p_i} is either a Guassian, or they are all identical and single-peaked (with different means) then the runtime can be improved to $O((n^{1+\tau}/\varepsilon^2)\log(1/\varepsilon))$ for any $\tau > 0$.

Axis-aligned bounding box. We now consider the family of shapes $\mathcal{S}$ describing axis-aligned bounding boxes in $\mathbb{R}^d$. For a point set P, we let the summarizing shape $S(P) \in \mathcal{S}$ minimize $f(P)$, which either represents the d-dimensional volume of $S(P)$ (the aabbv case — minimizes the area in $\mathbb{R}^2$) or the $(d-1)$-dimensional volume of the boundary of $S(P)$ (the aabbp case — minimizes the perimeter in $\mathbb{R}^2$). Figure 3.8 shows two examples of elements of $\mathcal{A}_{f,n}$ for the aabbp case and the aabbv case in $\mathbb{R}^2$. For both $(\mathbb{R}^2, \mathcal{A}_{f,n})$ has a shatter dimension of 4 because the shape is determined by the x-coordinates of 2 points and the y-coordinates of 2 points. This generalizes to a shatter dimension of $2d$ for $(\mathbb{R}^d, \mathcal{A}_{f,n})$. We can also show the VC-dimension of $(\mathbb{R}^2, \mathcal{A}_{f,n})$ is 8 for aabbp because its shape is defined by the intersection of halfspaces with 4 predefined normal directions at 0°, 45°, 90°, and 135°. This can be generalized to higher dimensions.

Hence, for $1 \leq i \leq n$, for both cases we can create an (ε/n)-sample of $(\mu_{p_i}, \mathcal{A}_{f,n})$, each of size $\lambda_f(n, \varepsilon) = O((n^2/\varepsilon^2)\log(n/\varepsilon))$ in total time $O((n^3/\varepsilon^2)\log^2(n/\varepsilon))$. For the aabbp case in $\mathbb{R}^2$, an (ε/n)-sample of each $(\mu_{p_i}, \mathcal{A}_{f,n})$ of can be reduced further to

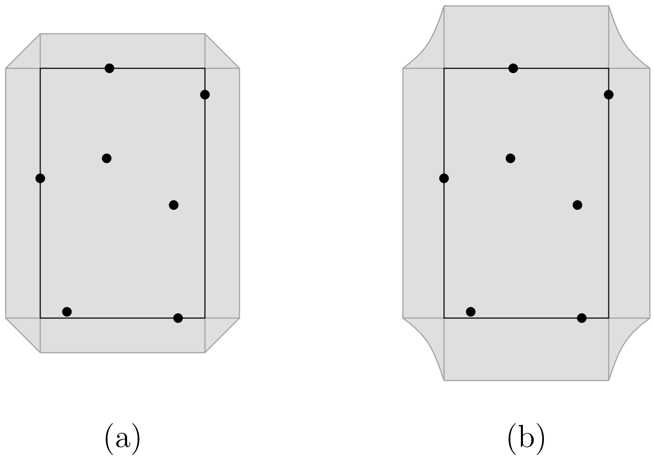


FIGURE 3.8: (a) Axis-aligned bounding box, measured by perimeter. (b) Axis-aligned bounding box, measured by area. The curves are hyperbola parts.

size $O((n/\varepsilon)\log^{16}(n/\varepsilon))$ in total time $O((n^5/\varepsilon^4)\log^{40}(n/\varepsilon))$. In $\mathbb{R}^d$, we can construct the ε-quantization in $(n^{6d}/\varepsilon^{4d})(\log(n/\varepsilon))^{O(d)}$ time, using orthogonal range searching. For the aabbp case in $\mathbb{R}^2$, the runtime improves to $O(n^8/\varepsilon^4\log^{65}(n/\varepsilon))$.

Smallest enclosing ball. Figure 3.9 shows example elements of $\mathcal{A}_{f,n}$ for smallest enclosing ball, for metrics L_∞ (the seb_∞ case) and L_1 (the seb_1 case) in $\mathbb{R}^2$. An example element of $\mathcal{A}_{f,n}$ for smallest enclosing ball for the L_2 metric (the seb_2 case) is shown in Figure 3.7. For seb_∞ and seb_1, $(\mathbb{R}^d, \mathcal{A}_{f,n})$ has VC-dimension $2d$ because the shapes are defined by the intersection of halfspaces from d predefined normal directions. For seb_1 and seb_∞, we can create n (ε/n)-samples of each $(\mu_{p_i}, \mathcal{A}_{f,n})$ of size $\lambda_f(n,\varepsilon) = O((n^2/\varepsilon^2)\log(n/\varepsilon))$ in total time $O((n^3/\varepsilon^2)\log^2(n/\varepsilon))$. The size for each can be reduced to $O((n/\varepsilon)\log^{2d}(n/\varepsilon))$ in $O((n^5/\varepsilon^4)\log^{8d}(n/\varepsilon))$ total time. Using an orthogonal range searching data structure we can calculate the ε-quantization in $O(n^{d+2}/\varepsilon^{d+1}\log^{7d-1}(n/\varepsilon))$ time.

For the seb_2 case in $\mathbb{R}^2$, $(\mathbb{R}^2, \mathcal{A}_{f,n})$ has infinite VC-dimension, but $(\mathbb{R}^2, \mathcal{W}_f)$ has VC-dimension at most 9 because it is the intersection of 2 halfspaces and one disc, as discussed in Section 3.5.1. Any shape $A(T,w) \in \mathcal{A}_{f,n}$ can be formed from the disjoint union of $2n$ wedges. Choosing a point in the convex hull of T as the vertex of the wedges will ensure that each wedge is completely inside the ball that defines part of its boundary. Thus, in $\mathbb{R}^2$ the n (ε/n)-samples of each $(\mu_{p_i}, \mathcal{A}_{f,n})$ are of size $\lambda_f(n,\varepsilon) = O(n^4/\varepsilon^2\log(n/\varepsilon))$ and can all be calculated in total time $O(n^5/\varepsilon^2\log^2(n/\varepsilon))$. And then the ε-quantization can be calculated in $O(n^{15}/\varepsilon^7\log^{3.5}(n/\varepsilon))$ time, using range searching data structures. We believe this technique can be extended to $\mathbb{R}^d$, but do not present a proof in this thesis.

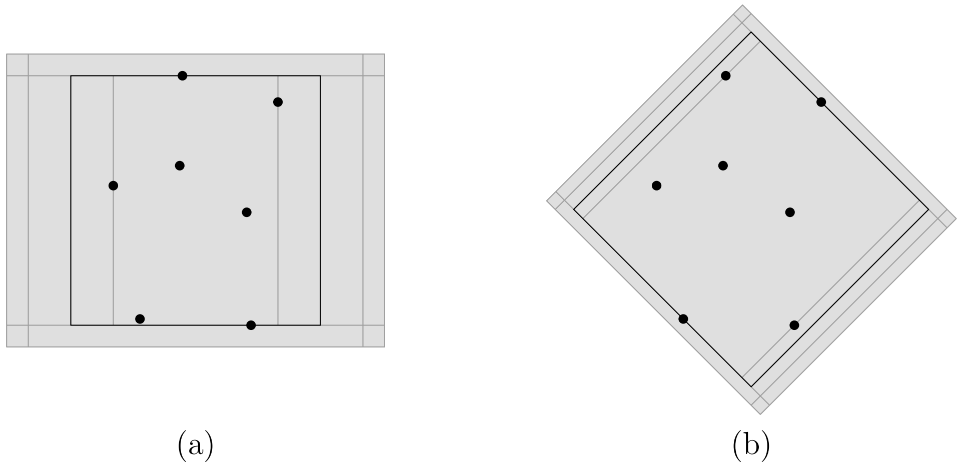


FIGURE 3.9: (a) Smallest enclosing ball, L_∞ metric. (b) Smallest enclosing ball, L_1 metric.

Table 3.4: ε-Samples for Summarizing Shape Family $\mathcal{A}_{f,n}$.

case	$\lambda_f(n,\varepsilon)$	$\nu_{\mathcal{S}}$	$(\lambda_f(n,\varepsilon))^{\nu_{\mathcal{S}}}$	$\mathsf{RS}(\lambda_f(n,\varepsilon),\mathcal{S})$	runtime
dwid	$O(n/\varepsilon)$	2	$O(n^2/\varepsilon^2)$	$\tilde{O}(1)$	$\tilde{O}(n^3/\varepsilon^2)$
aabbp	$\tilde{O}(n^2/\varepsilon^2)$	$2d$	$\tilde{O}(n^{4d}/\varepsilon^{4d})$	$\tilde{O}(1)$	$\tilde{O}(n^{4d+1}/\varepsilon^{4d})$
aabbv	$\tilde{O}(n^2/\varepsilon^2)$	$2d$	$\tilde{O}(n^{4d}/\varepsilon^{4d})$	$\tilde{O}(1)$	$\tilde{O}(n^{4d+1}/\varepsilon^{4d})$
seb$_\infty$	$\tilde{O}(n/\varepsilon)$	$d+1$	$\tilde{O}(n^{d+1}/\varepsilon^{d+1})$	$\tilde{O}(1)$	$\tilde{O}(n^{d+2}/\varepsilon^{d+1})$
seb$_1$	$\tilde{O}(n/\varepsilon)$	$d+1$	$\tilde{O}(n^{d+1}/\varepsilon^{d+1})$	$\tilde{O}(1)$	$\tilde{O}(n^{d+2}/\varepsilon^{d+1})$
seb$_2$ $\in \mathbb{R}^2$	$\tilde{O}(n^4/\varepsilon^2)$	3	$\tilde{O}(n^{12}/\varepsilon^6)$	$\tilde{O}(n^2/\varepsilon)$	$\tilde{O}(n^{15}/\varepsilon^7))$
diam $\in \mathbb{R}^2$	$\tilde{O}(n^4/\varepsilon^2)$	n	$\tilde{O}(n^4/\varepsilon^2)^n$	$\tilde{O}(n^4/\varepsilon^2)$	$\tilde{O}(n\cdot(n^4/\varepsilon^2)^{n+1})$

$\tilde{O}(f(n,\varepsilon))$ ignores poly-logarithmic factors $(\log(n/\varepsilon))^{O(\mathrm{poly}(d))}$.

Diameter. Given a distribution of an n point set μ_P, we consider computing an ε-quantization of F_{μ_P} where f measures the diameter of a point set (the diam case). Figure 3.10 shows an example element of $\mathcal{A}_{f,n}$ in $\mathbb{R}^2$. There is not a convenient family of shapes $\mathcal{S}$ so that for a point set $P \subset \mathbb{R}^d$, $\mathcal{S}(P)$ is the summarizing shape which optimizes the diameter, and where $(\mathbb{R}^d,\mathcal{S})$ has bounded VC-dimension. We can however define $\mathcal{S}$ as the intersections of n balls of a fixed common radius, so $(\mathbb{R}^d,\mathcal{S})$ has VC-dimension $\Omega(n)$. In this definition of $\mathcal{S}$, given a point set P, the summarizing shape $\mathcal{S}(P)$ is the intersection of balls centered at each point in P and the radius is the diameter of P—the smallest radius so $P \subset \mathcal{S}(P)$. An element $A(T,w) \in \mathcal{A}_{f,n}$ describes the intersection of $n-1$ balls of radius w, each centered at a point in T, such that $T \subset A(T,w)$; thus $A(T,w)$ must be convex. Hence $\mathcal{S} = \mathcal{A}_{f,n+1}$ and $(\mathbb{R}^d,\mathcal{A}_{f,n})$ does not have constant VC-dimension. We can, however, describe a family of wedges $\mathcal{W}_f$ such that $(\mathbb{R}^2,\mathcal{W}_f)$ has constant VC-dimension and for any $A \in \mathcal{A}_{f,n}$ we can describe A as the disjoint union of n shapes from the family $\mathcal{W}_f$. We use a similar construction as with the seb_2 case where an element $W \in \mathcal{W}_f$ is the intersection of a ball and two halfspaces. Thus, in $\mathbb{R}^2$ for $1 \leq i \leq n$ we can create an (ε/n)-sample for each $(\mu_{p_i}, \mathcal{A}_{f,n})$ of size $O(n^4/\varepsilon^2 \log(n/\varepsilon))$ in time $O(n^4/\varepsilon^2 \log^2(n/\varepsilon))$ by creating an (ε/n^2)-sample for $(\mu_{p_i}, \mathcal{W}_f)$. This takes total time $O(n^5/\varepsilon^2 \log^2(n/\varepsilon))$. We believe there is an analogous construction in $\mathbb{R}^d$, but do not present a proof in this thesis.

This allows us to create a set Q_P such that for all $r \in \mathbb{R}$ that $|\tilde{f}(Q_P, r) - F_{\mu_P}(r)| \leq \varepsilon$. However, because $(\mathbb{R}^d, \mathcal{S})$ has $\sigma_{\mathcal{S}} = n$ shatter dimension, Theorem 3.8 does not provide a method to create a point set to concisely describe an ε-quantization in time polynomial in n. The runtime is $O(n(n^4/\varepsilon^2 \log(n/\varepsilon))^{n+1})$. We have learned through personal communication [48] that a polynomial time algorithm for evaluating $\tilde{f}(Q_P, r)$ does exist.

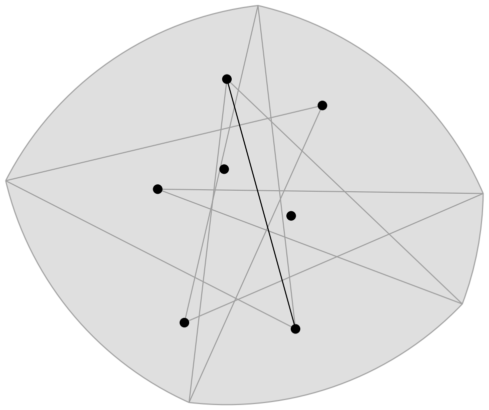


FIGURE 3.10: Diameter. The curves are circular arcs all of the same radius.

4
Computing Spatial Scan Statistics

4.1 Motivation and Problem Statement

Outlier detection is a common problem in data mining. Unlike in robust clustering settings, where the goal is to detect outliers in order to remove them, outliers are viewed as *anomalous events* to be studied further. In the area of biosurveillance for example, an outlier would consist of an area that had an unusually high disease rate (disease occurrence per unit population) of a particular ailment. In environmental monitoring scenarios, one might monitor the rainfall over an area and wish to determine whether any region had unusually high rainfall in a year, or over the past few years.

A formal statistical treatment of these problems allows us to abstract them into a common framework. Let P be a point set describing the geometric position of the data. It may be discrete, each point $p \in P$ may have a weight $\mu(p)$, or more generally, P may be Lebesgue measurable describing a distribution. Let b_0 and m_0 describe two functions on P. Specifically, let $b_0 : P \to \mathbb{R}^+$ represent the baseline data and let $m_0 : P \to \mathbb{R}^+$ represent the measured data. Examples of this sort of data include:

- Biosurveillance: P being the households of a set of people. The function b_0 could describe the number of people in each household. And the function m_0 could be the number of times someone in a household is treated for a disease.
- Enviorонmental Monitoring: P could represent a region of land. The function b_0 may describe the expected rainfall everywhere from some existing model. And the function m_0 may describe the rainfall recorded over one season, either interpolated over the entire region, or at a discrete set of measurements.
- National Security: P may represent a distribution of peoples' locations, and

they may not be known exactly because of collection errors or perturbation to preserve anonymity. The function b_0 could represent the number of search engine searches of each individual. This raw data may be augmented using spatial models of variation. Finally, m_0 may be the number of a specific type of search (i.e. those that may be the searches of terrorist).

Next, we may want to determine if there is a subset of P which somehow is anomalous. To prevent frivolous subsets which gerrymander just the data with large values of m_0 even if they have no spatial similarity, we need to consider a family $\mathcal{A}$ of subsets of P. Many families of $\mathcal{A}$ have been used, often those determined by inclusion in certain shapes: circles and ellipses [70, 72], axis-aligned rectangles [90, 40], connected components [94]. We call the subsets $R \subset P$ where $R \in \mathcal{A}$ a *range*. And the set $(P, \mathcal{A})$ is indeed a range space in the sense defined for ε-samples. For much of the computational aspects of this chapter we focus on the family $\mathcal{A} = \mathcal{R}_2$, defined by axis-aligned rectangles.

Finally, we need to consider a measure on a range R using P, m_0, and b_0, determining how anomalous R is. Specifically, for any range $R \subset P$ let $m(R) = \sum_{p \in R} m_0(p)$ and $b(R) = \sum_{p \in R} b_0(p)$ (sometimes this technically needs to be written with an integral instead of a sum), and let $M = m(P)$ and $B = b(P)$. Now we can introduce a *discrepancy function* $d'(m(R), b(R), M, B)$[1]. We will discuss several variants of discrepancy functions, with a particular interest in those defined to measure statistics-based properties. In general, the goal of a discrepancy function is to determine how different the functions b and m are in the range R versus their values on $P \setminus R$. For instance, the disease rate $m(R)/b(R)$ may be much larger in R than in $P \setminus R$. And the problem of *statistical discrepancy* is to determine the range which maximizes a discrepancy function (i.e. it is the most anomalous range with respect to m and b) and to estimate how unlikely the discrepancy of this range is. We derive several specific discrepancy functions based on the notion of a likelihood ratio test.

The statistical discrepancy value of the range that maximizes the chosen statistical discrepancy function is called a *spatial scan statistic* because it can be computed by "scanning" over all of the data with a shape defining the ranges and evaluating the discrepancy on each to determine the maximum. Let this value be v for discussion. Furthermore, to attempt to measure how unlikely this discrepancy measure v is, a *p-value* is often computed. This process involves, under a null model of what we expect the data to look like (i.e. we let $m_0'(p) = b_0(p)M/B$), we generate a large number N of random samples from the null model [41, 47] (i.e. according to m_0', fixing b_0 and P) and compute the maximum discrepancy range for each. The set of these values can be used to calculate an ε-quantization of the spatial scan statistic under the null model. So if $N = O((1/\varepsilon^2)\log(1/\delta))$, then, with probability $1-\delta$, we can predict the probability within ε that a value as large as v occurs under the null

[1] This is indeed related to the concepts of combinatorial and Lebesgue discrepancy discussed in Chapter 1, although we will not directly discuss these versions in this chapter.

model. This probability is known as the p-value[2]. Because N is often quite large (e.g. $N = 10,000$), evaluating the statistical discrepancy is reduced to computing the spatial scan statistic many times over, and the optimization of this core operation is amplified. Thus this chapter focuses on computing the spatial scan statistic quickly and accurately.

4.1.1 Problem Statement

We now narrow the problem to the case where $P \subset \mathbb{R}^2$ is a set of n points in the plane and the ranges $\mathcal{R}_2$ are the points defined by containment in an axis-aligned rectangles. In general all algorithms can be extended to $\mathbb{R}^d$ and axis-aligned boxes, $\mathcal{R}^d$, by multiplying the runtime by $O(n^{2(d-2)})$. Also, if P is Lebesgue-measurable, we can first take an ε'-sample using the results of Chapter 1, for an appropriate value of ε' as discussed in Section 4.5. Alternatively if P is piecewise-linear, the we could try computing directly on P, but this is difficult as described in Section 4.C.

For a range $R \in \mathcal{R}_2$, let $m_R = m(R)/M$ and $b_R = b(R)/B$. Most discrepancy functions we will consider can be written just in terms of m_R and b_R in which case we write

$$d'(m(R), b(R), M, B) = d(m_R, b_R)$$

for notational convenience. For instance we can write the Poisson discrepancy as

$$d_P(m_R, b_R) = m_R \log \frac{m_R}{b_R} + (1 - m_R) \log \frac{1 - m_R}{1 - b_R}$$

up to a fixed constant. For any parameters α, β, γ a *linear discrepancy function* is written the

$$d_l(m_R, b_R) = \alpha m_R + \beta b_R + \gamma.$$

d_P and d_l are graphed in Figure 4.1. Furthermore, *bichromatic discrepancy* is written

$$d'_\chi(m(R), b(R), M, B) = m(R) - b(R) = d_\chi(m_R, b_R) = M \cdot m_R - B \cdot b_R B$$

which counts the total difference between $m(R)$ and $b(R)$. Bichromatic discrepancy $d_\chi(m_R, b_R)$ is basically the same as the combinatorial discrepancy function $\overline{\text{disc}}_\chi(P \cap R)$ where $\chi(p) = +1$ if $p \in M$ and $\chi(p) = -1$ if $p \in B$. This is most interesting when $M = B$, and the name comes from when $m_0(p) = 1$ if p is red and $b_0(p) = 1$ if p is blue. Also note that d_χ is a variant of linear discrepancy. Every discrepancy function $d : [0,1]^2 \to \mathbb{R}^+$ we consider will be convex in m_R and b_R.

The key problem we study in this chapter is:

Problem 4.1 (Maximizing Discrepancy). *Given a point set P with measured and baseline functions m and b, a family of ranges $\mathcal{R}_2$, and a convex discrepancy function d, find the range $R \in \mathcal{R}_2$ that maximizes $d(m_R, b_R)$.*

[2] We acknowledge that the usefulness of the p-value is debatable; however, this chapter only focuses on the computational issues of this process, and leaves the debate of usefulness to another time.

An equivalent formulation, replacing the range R by the point $r = (m_R, b_R)$ is:

Problem 4.2. *Maximize convex discrepancy function d over all points $r = (m_R, b_R)$ such that $R \in \mathcal{R}_2$ for a range space $(P, \mathcal{R}_2)$.*

Assume that points $p \in P$ now arrive with a timestamp $t(p)$, along with the measurement $m(p)$ and baseline $b(p)$. In *prospective discrepancy* problems, the goal is to maximize discrepancy in a range $R_t \in \mathcal{T}_2$ where $\mathcal{T}_2$ is all subsets defined by containment in a rectangle and such that all time stamps $t(p) \in [t, \infty)$ for some $t \in \mathbb{R}$. In other words, the region includes all points with a timestamp between the present time and some time t in the past. Such regions are interesting when attempting to detect *recent* anomalous events.

Problem 4.3 (Prospective discrepancy)**.** *Given a point set P with measure and baseline functions m and b, timestamps t, a family of ranges $\mathcal{T}_2$, and a convex discrepancy function d, find the range $R_t \in \mathcal{T}_2$ that maximizes d.*

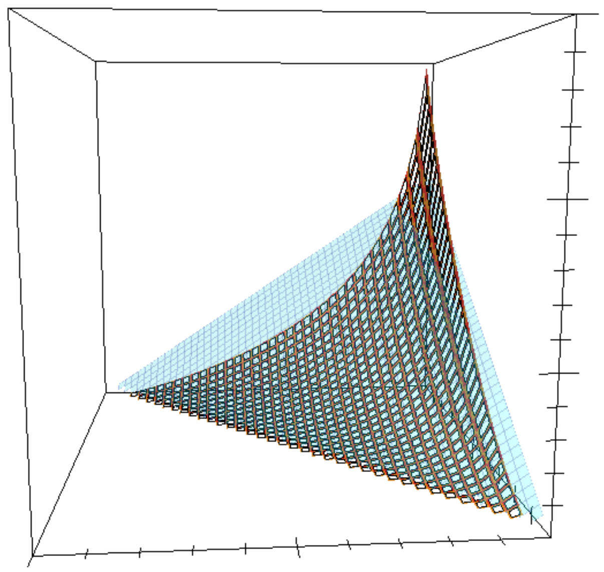


FIGURE 4.1: Graph of d_P (solid and gridded) and d_l (transparent) with $\alpha = 1$, $\beta = -1$, and $\gamma = 0$, plotted over $S_n \subset [0, 1]^2$.

Boundary conditions. The statistical discrepancy functions we consider can be expressed as log-likelihood ratios. As a consequence, they tend to ∞ when either argument tends to zero (while the other remains fixed). Another way of looking at this issue is that regions with very low support often correspond to overfitting and thus are not interesting. Therefore, this problem is typically addressed by requiring a *minimum level of support* in each argument. Specifically, we will only consider maximizing discrepancy over ranges $R \in \mathcal{R}_2$ such that $m_R > C/n, b_R > C/n$, for some constant C (for cleaner exposition, we usually set C to 1). In mapping shapes to points as described above, this means that the maximization is restricted to points in the square $[C/n, 1-C/n]^2$. Furthermore, we generally are looking for regions with more measured points than baseline points, otherwise, we would switch the roles of the measured and baseline points. Thus we restrict that $m_R > b_R$ as well. We refer to this remaining domain as $S_n = [C/n, 1 - C/n]^2 \setminus \{m_R \leq b_R\}$. For technical reasons, we will also assume that for all p, $m(p), b(p) = \Theta(1)$. Intuitively this reflects the fact that measurement values are independent of the number of observations made.

Grid algorithms For some algorithms, the data is assumed to lie on a grid, or is accumulated onto a set of grid cells. For such algorithms, we will assume a grid of size $g \times g$, with measurement and baseline values associated with each grid point as before. Note that in such a grid, the effective number of points is g^2, and the number of distinct axis-parallel rectangles is $O((g^2)^2) = O(g^4)$, which differs from the corresponding numbers n and $O(n^4)$ for points and axis-parallel rectangles in general position. It will be important to keep this distinction in mind when comparing grid algorithms with those taking inputs in general position.

We could also treat this data as each data point at a grid point actually has a probability distribution of being anywhere in that grid cell with equal probability. We can then use the techniques of Chapter 3 to create an ε-quantization over the values of statistical discrepancy.

4.1.2 Contributions

We consider four statistical discrepancy functions d_P, d_B, d_G, d_γ based on when the data is modeled as coming from a Poisson, Bernoulli, Gaussian, or gamma distribution, respectively. We derive these functions and properties of them in Section 4.B. We also describe how to derive the associated discrepancy function for data modeled by any one-parameter exponential family in Section 4.A. These two sections are put in the appendix of this chapter so as to not interrupt the important algorithmic ideas.

In Section 4.2 we develop a technique to approximate a convex function $d : S_n \to \mathbb{R}^+$ with a family of linear functions. This has application in approximating d_P, d_B, d_Q, and d_γ with families of simpler linear discrepancy functions. For instance we can create an additive ε-approximation of d_P by the upper envelope of $O((1/\varepsilon)\log^2 n)$ linear discrepancy functions.

We then describe several algorithms for computing the maximum discrepancy range $R \in \mathcal{R}_2$ for a set P of n points.

- The first, described in Section 4.3, is exact and runs in $O(n^4)$ time, and works for any discrepancy function $d(m_R, b_R)$.

- Next, in Section 4.4 we describe an $O(n^2 \log n)$ time algorithm for any linear discrepancy function, which can then be combined with the function approximation techniques to solve the statistical discrepancy functions approximately.

- In Section 4.5 we consider algorithms that first create an ε-sample of the data, and then perform the above algorithms on the ε-sample. It turns out this is efficient for many linear discrepancy functions and d_B, but not for the other statistical discrepancy functions. A variant of this approach can be used as a streaming algorithm. We also prove streaming lower bounds in this section.

- Finally, in Section 4.6 we develop algorithms for data that lie on an axis-aligned grid. We present exact and approximation grid algorithms based on the linearization theorem from Section 4.2.

In published work on this topic [2], we showed that these algorithms are comparable or better than existing algorithms in terms of the running time versus the discrepancy value returned and its variance. However, since we have made numerous subtle improvements to the algorithms since those simulations, we have not added the outdated experiments in this thesis. Further experiments are planned.

Furthermore, these results generalize to higher dimensional data points, as well as prospective discrepancy where we detect the maximum discrepancy region over all intervals starting from the present and going backwards in time.

4.1.3 Related Work

Detecting clustering effects in spatial data is a well-studied problem in statistics[3]. Much of the early focus has been on devising efficient statistical tests to detect presence of clustering at a global level without emphasis on identifying the actual clusters (see [37, Chapter 8]). The spatial scan statistic, introduced by Kulldorff [70] provides an elegant solution for detection and evaluation of spatial clusters. The technique has found wide applicability in areas like public health, biosurveillance, environmental monitoring. See the webpage `http://www.SatScan.org` [72] for his software, SatScan, and links to many variants and references. Generalization of the spatial scan statistic to a space-time scan statistic for the purpose of prospective surveillance has been proposed by Kulldorff [71], and Iyengar [64] suggested the use of "expanding-in-time" regions to detect space-time clusters. The regions Kulldorff considers are circular but only centered at a limited set of fixed points, or cylindrical in the case of prospective surveillance. Wang *et al.* [117] has generalized this framework for graphs where the subsets with large discrepancy are considered clusters.

The spatial scan statistics work is based on the simpler ideas of scan statistics which are strictly one-dimensional problems, and thus the ranges are always data

[3] It goes without saying that there is a huge literature on clustering spatial data. Since our focus is primarily on statistically sound measures, a survey of general clustering methods is beyond the scope of this thesis.

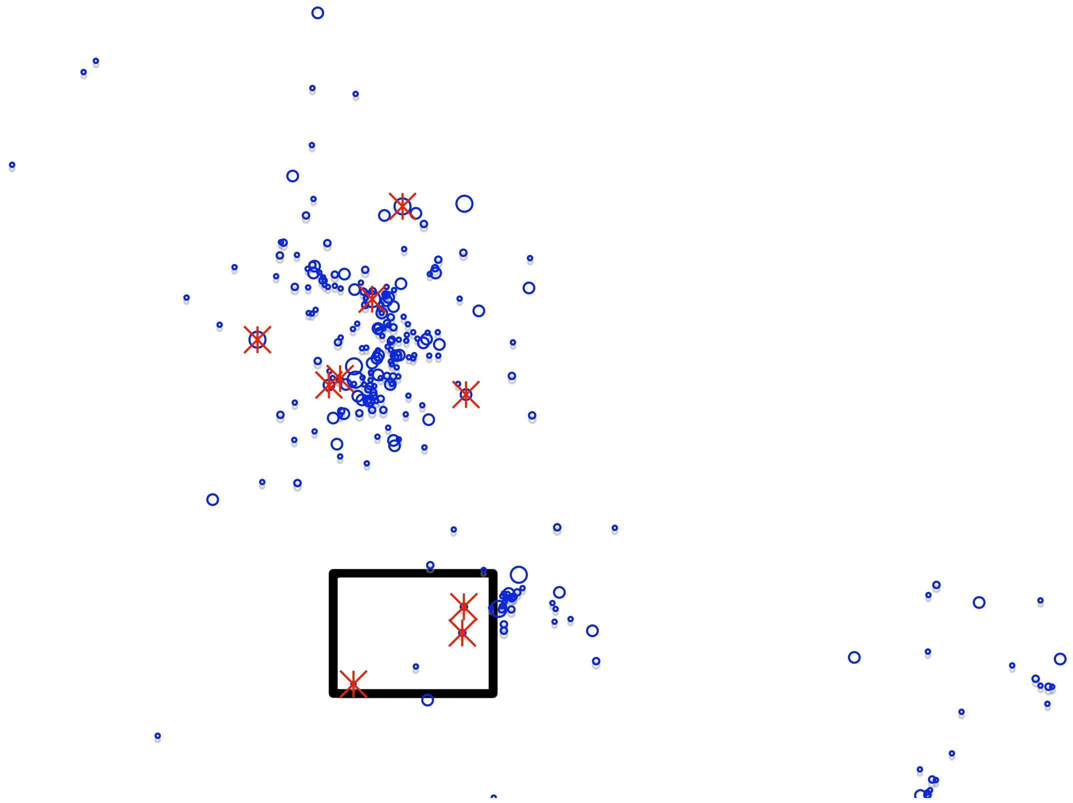


FIGURE 4.2: Example of maximal discrepancy range on a data set. Xs are measured data and Os are baseline data.

contained in intervals, often of a fixed width. For interesting applications and detailed description of scan statistics, we refer the reader to [65, 66]. Recently, the scan statistic has also been used in other areas like bioinformatics [63, 77] and for detecting chatter in massive social networks [104].

Dobkin and Eppstein [39] were the first to study efficient algorithms to compute maximum discrepancy over a range space. Their algorithms compute Lebesgue discrepancy in a region R as a difference between the fraction of points in R and the fraction of the total area represented by R. This measure stems from evaluating fundamental operations for computer graphics. Their ranges were defined by containment in half spaces and axis-oriented orthants centered at the origin, limited to the unit cube. Their results extended to d-dimensional spaces. Subsequently Dobkin, Gunopulous, and Maass [40] developed algorithms for computing maximum bichromatic discrepancy over axis-alligned rectangular regions. This solves the *minimum disagreement problem* from machine learning, where an algorithm finds the region with the most *good* points and the fewest *bad* points, a key subroutine in agnostic PAC learning.

Recently, Neill and Moore have developed a series of algorithms [90, 89, 88] to maximize discrepancy for measures such as d_P. Their approach works for axis parallel squares [89] and rectangles [90]. Their algorithms are conservative, in that they always find the region of maximum discrepancy. The worst-case running time of their algorithms is $O(g^4)$ for rectangles and $O(g^2)$ for fixed-size squares since the algorithms enumerate over all valid regions on a $g \times g$ grid. However, they use efficient pruning heuristics that allow for significant speedup over the worst case on most data sets. An alternative approach by Friedman and Fisher [44] greedily computes a high discrepancy rectangle, but has no guarantees as to how it compares to the optimal. Their approach is quite general, and works in arbitrary dimensional spaces, but is not conservative: many regions will remain unexplored.

A related problem that has a similar flavor is the so-called *Heavy Hitters* problem [36, 35]. In this problem, one is given a multiset of elements from a universe, and the goal is to find elements whose frequencies in the multiset are unusually high (i.e much more than the average). In a certain sense, the heavy hitter problem fits in our framework if we think of the baseline data as the uniform distribution, and the counts as the measurements. However, the ranges are the trivial subsets formed by one data point[4] and the heavy hitter problem itself is interesting in a streaming setting, where memory is limited; if linear memory is permitted, the problem is trivial to solve, in contrast to the problems we consider.

4.2 A Convex Approximation Theorem

This section describes a general approximation theorem for maximizing a convex discrepancy function d. Let $\ell(x, y) = ax + by + c$ denote a linear function in x and y, where a, b, c are constant. Define an *ε-approximate family* of d to be a collection of linear functions $\mathcal{L} = \{\ell_1, \ell_2, \ldots, \ell_t\}$ such that $l^U(x, y) = \max_{i \leq t} \ell_i(x, y)$, the *upper envelope* of the ℓ_i, has the property that $l^U(x, y) \leq d(x, y) \leq l^U(x, y) + \varepsilon$. Say a set of points Q in S_n, describes an ε-approximation family $\mathcal{L}_Q$ of linear functions for a smooth convex function f, where each $\ell_q \in \mathcal{L}_Q$ for $q \in Q$ is defined as the linear function equal and tangent to f at $f(q)$. Let

$$H(f) = \begin{pmatrix} \frac{d^2 f}{dx^2} & \frac{d^2 f}{dx dy} \\ \frac{d^2 f}{dy dx} & \frac{d^2 f}{dy^2} \end{pmatrix}$$

be the Hessian of f. Define $\lambda_{pq} = \max_{r \in \overline{pq}} u_{pq}^\top H(f(r)) u_{pq}$ where $u_{pq} = (p - q)/(||p - q||)$ and where $\overline{pq}$ is the segment from p to q. In other words, λ_{pq} is the largest eigenvalue of the Hessian $H(f)$ along the segment $\overline{pq}$. We simplify some notation by writing

$$f_{xx} = \frac{d^2 f}{dx^2} \quad \text{and} \quad f_{yy} = \frac{d^2 f}{dy^2}.$$

[4] Hierarchical heavy hitters defines a more interesting range space.

Lemma 4.1. *For a point set Q to describe an ε-approximation family of f it is sufficient that for any point $p \in S_n$ there exists a point $q \in Q$ such that $||p - q|| \leq \sqrt{\varepsilon / \lambda_{pq}}$.*

Proof. Let $\tilde{f}_q(p) = f(q) + (p - q)^\top \nabla f(q)$. The inequality $\tilde{f}_q(p) \leq f(p)$ follows from the convexity of f. By Taylor's theorem for multivariate functions, the error $f(p) - \tilde{f}_q(p) = \int_{r \in \overline{pq}} (q - p)^\top H(f(r))(q - p)$, where $H(f)$ is the Hessian of f, and $\overline{pq}$ is the segment joining p and q. Taking the maximum value r upper bounds the integral and implies $\varepsilon \geq \lambda_{pq} ||p - q||^2$. □

Lemma 4.2. *For a convex function f, then for any vector $u = (u_x, u_y)$, $u^\top H(f) u < 2(f_{xx} u_x^2 + f_{yy} u_y^2)$.*

Proof. $H(f)$ is a symmetric positive definite matrix. Any symmetric positive definite matrix

$$M = \begin{pmatrix} a & b \\ b & c \end{pmatrix} \text{ satisfies } a u_x^2 - 2b u_x u_y + c u_y^2 > 0$$

for any vector (u_x, u_y). Restating this as $a u_x^2 + c u_y^2 > 2b u_x u_y$ implies the upper bound for the cross term. The stated inequality follows. □

Using Lemma 4.2 we can upper bound the value λ_{pq} by $\lambda_{pq} \leq 2(\max_{r \in \overline{pq}} f_{xx}(r) + \max_{r \in \overline{pq}} f_{yy}(r))$, thus allowing us to greedily place points of Q in the x- and y-directions independently with respect to f_{xx} and f_{yy}, respectively.

Lemma 4.3. *Consider the region $H_{x,y} = [x - dx, x] \times [y, y + dy] \in S_n$ where $\lambda_{x,y}^x = \max_{r \in H_{x,y}} f_{xx}(r)$ and $\lambda_{x,y}^y = \max_{r \in H_{x,y}} f_{yy}(r)$ and where $dx \leq \sqrt{\varepsilon / \lambda_{x,y}^x}$ and $dy \leq \sqrt{\varepsilon / \lambda_{x,y}^y}$. Then a single point at $(x - dx/2, y + dy/2)$ describes an ε-approximation family for the underlying convex function f in the domain $H_{x,y}$.*

Proof. Any points $p \in H_{x,y}$ has distance less than $(1/2)dx$ in the x-direction and $(1/2)dy$ in the y-direction from the point $q = (x - dx/2, y + dy/2)$. By Lemma 4.1 for all points $p \in H_{x,y}$, as long as $f(p) - \bar{f}_q(p) \leq (p - q)^T \lambda_{pq} (p - q)$ then q describes an ε-approximation. Thus we can also enforce

$$\begin{aligned}
(p - q)^T \lambda_{pq} (p - q) &\leq \left(\frac{dx}{2}, \frac{dy}{2}\right)^T \lambda_{pq} \left(\frac{dx}{2}, \frac{dy}{2}\right) \\
&< 2\left(\lambda_{x,y}^x \left(\frac{dx}{2}\right)^2 + \lambda_{x,y}^y \left(\frac{dy}{2}\right)^2\right) \\
&= \frac{1}{2}\left(\lambda_{x,y}^x (dx)^2 + \lambda_{x,y}^y (dy)^2\right)
\end{aligned}$$

using Lemma 4.2. Therefore, as long as we bound $\lambda_{x,y}^x (dx)^2 \leq \varepsilon$ and $\lambda_{x,y}^y (dy)^2 \leq \varepsilon$, q describes and ε-approximation for $H_{x,y}$. □

This implies that if S_n is covered by regions $H_{x,y}$ as described in Lemma 4.3, then if a point $q \in Q$ is at the center of each region, then Q describes an ε-approximation for S_n.

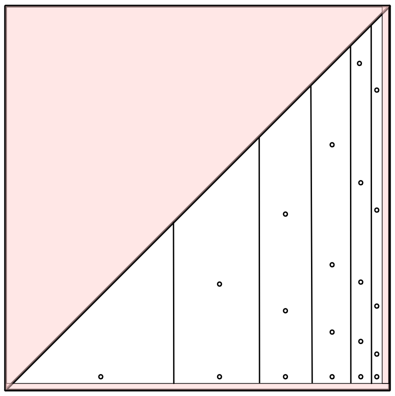


FIGURE 4.3: Region S_n in white. Points in point set Q represent the point of tangency of linear functions forming ε-approximation family with a function $f : S_n \to \mathbb{R}$.

We determine this covering with a greedy approach. Let $x_{\max} = \max_x\{(x, y) \in S_n\}$ and let $y_{\min} = \min_y\{(x_{\max}, y) \in S_n\}$. We start with $H_{x_{\max},y_{\min}}$ and then place another box at $H_{x_{\max},y_{\min}+dy}$ where dy is chosen to satisfy Lemma 4.3. We repeat this process until the y-coordinate is outside of S_n. Then we take the smallest dx for all boxes in that column and reset $x_{\max}$ to $x_{\max} - dx$ and repeat until the x-coordinate is outside of S_n.

Choosing the largest possible dx and dy at each step is easy if f_{xx} is increasing in x and f_{yy} is decreasing in y. Then $\lambda^x_{x,y} = f_{xx}(x, y)$ and $\lambda^y_{x,y} = f_{yy}(x, y)$. If this is not the case, then $\lambda^x_{x,y}$ and $\lambda^y_{x,y}$ can be estimated to within a factor of 2 quite easily by using $f_{xx}(x, y)$ and $f_{yy}(x, y)$ as estimates and bounding the estimates using third derivatives of f.

Remark. In the Binomial case the lower boundary of S_n is $y = Gx + 1/n$, not $y = 1/n$. The above argument can be modified to handle this case by placing each $y_{\min}$ on $y = Gx + 1/n$ and using this to determine dx.

Theorem 4.1. *The algorithm presented above constructs a point set that describes an ε-approximation family for a convex function f of size $O((1/\varepsilon)k_x(f) \cdot k_y(f))$, where for $i = \{x, y\}$ and for any constant $\eta = \{x, y\} \setminus i$ which is ignored in $f_{ii}(\cdot)$.*

$$k_i(f) = \begin{cases} O(\log n) & \min_{S_n} f_{ii}(j) = \Omega(1/j^2) \\ O(\sqrt{n} \log\log\log n) & \min_{S_n} f_{ii}(j) = \Omega(1/j^3) \\ O(\sqrt{n}) & \min_{S_n} f_{ii}(j) = \Omega(1/n) \end{cases}$$

Proof. The size of Q can be written as a recurrence function $C_\varepsilon : S_n \to \mathbb{R}$, which represents the number of points needed to describe an ε-approximation for the part of S_n with smaller x-coordinates and larger y-coordinates than the point given (i.e., $|Q| = C_\varepsilon(1-1/n, 1/n)$). We can write

$$C_\varepsilon(x,y) = Y_\varepsilon(x,y) + C_\varepsilon\left(x - \sqrt{\varepsilon/f_{xx}(x,y)}, y\right),$$

where $Y_\varepsilon(x,y)$ is the number of points needed to describe an ε-approximation in S_n with an x-value x and a y-value greater than y. We can write

$$Y_\varepsilon(x,y) = 1 + Y_\varepsilon\left(x, y + \sqrt{\varepsilon/f_{yy}(x,y)}\right).$$

We solve the recurrences for $Y_\varepsilon(x, 1/n)$ setting $f_{yy}(x, 1/j) = \{\Omega(1/j^2), \Omega(1/j^3), \Omega(1/n)\}$. The same analysis will apply for $C_\varepsilon(1/j, y)$ to yield the desired bound. Let y_i be the value of y in the ith recursive call of $Y_\varepsilon(x,y)$, and for $Y_\varepsilon(x,1/n)$, let $y_0 = 1/n$.

When $f_{yy}(j) = \Omega(1/j^2)$, we can state $y_i = y_0(1+\sqrt{\varepsilon})^i$. Thus $y_i \geq 1$ if $i \geq \log_{1+\sqrt{\varepsilon}} n = O((1/\sqrt{\varepsilon})\log n)$.

When $f_{yy}(j) = \Omega(1/n)$, then $y_{i+1} \geq y_i + \sqrt{\varepsilon}/\sqrt{n}$, then $y_i \geq 1$ when $i = O(\sqrt{n/\varepsilon})$.

When $f_{yy}(j) = \Omega(1/j^3)$ we first argue that if $y_i = t/n$ then $y_{i+s} \geq t/n + (\sqrt{\varepsilon}t^{3/2}/n^{3/2})s$. Thus if we set $s = O((1/\sqrt{\varepsilon})n^{1/2})$ then $y_{i+s} \geq 2y_i$ or even better if $y_i = t/n$, then $y_{i+s} = (t + t^{3/2})/n$. By repeating this $\log_{3/2}\log_2 n$ times we can show $Y_\varepsilon(x, 1/n) = O((1/\sqrt{\varepsilon})n^{1/2}\log\log n)$. We now improve this.

If $y_i = t/n$ and $t \geq 4$ (i.e., $t^{3/2}/2 \geq t$) and $t/2 \geq \log n$, then after $s = O((1/\sqrt{\varepsilon})n^{1/2}/\log n)$ steps, $y_{i+s} = 2t/n$. Thus after $t = 2\log n$, then it takes only $O((1/\sqrt{\varepsilon})n^{1/2})$ more steps to get $y_i \geq 1$. Applying the first argument lets us reach $y_i = (2\log n)/n$ in $O((1/\sqrt{\varepsilon})n^{1/2}\log\log\log n)$ steps. □

4.2.1 Bounds for Specific Distributions

For Poisson, Bernoulli, and gamma we have $Y_\varepsilon(x, 1/n) = O((1/\sqrt{\varepsilon})\log n)$, for all $x \in S_n$. For Gaussian we have $Y_\varepsilon(x, 1/n) = O(\sqrt{n/\varepsilon})$, for all $x \in S_n$. For Poisson and gamma we have $C_\varepsilon(1-1/n, y)$ requiring $O((1/\sqrt{\varepsilon})\log n)$ steps, for any value of y. For Bernoulli we have $C_\varepsilon(1-1/n, y)$ requiring $O(\sqrt{n/\varepsilon})$ steps, for $y = 1/n$. (This approach is not as straightforward because of the strange shape of its boundary — the lower y boundary follows $y = Gx + 1/n$, and each point along here has $f_{xx}(x, Gx+1/n) = O(1/n)$.) For Gaussian we have $C_\varepsilon(1-1/n, y)$ requiring $O(\sqrt{n/\varepsilon})$ steps, for $y = 1/n$.

Theorem 4.2. *We can construct a point set describing an ε-approximation for f_P and f_γ of size $O((1/\varepsilon)\log^2 n)$. We can construct a point set describing an ε-approximation for f_B of size $O((1/\varepsilon)n^{1/2}\log n\log\log\log n)$. We can construct a point set describing an ε-approximation for f_G of size $O((1/\varepsilon)n)$.*

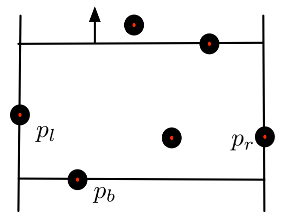


FIGURE 4.4: Sweep Line in Algorithm EXACT.

4.3 A Simple Exact Algorithm

In this section we present a simple algorithm running in time $O(n^4)$ that computes the maximum discrepancy rectangle exactly. Even though there are $O(n^4)$ rectangles to be considered, a naive search strategy might end up taking linear time for each rectangle (to estimate m_R, b_R) yielding a net running time of $O(n^5)$. We use a simple sweep line technique and incremental updates to avoid this problem.

Any set of four points defines a unique bounding rectangle, with one point defining each side. See Figure 4.4. Fix a pair of points $p_r, p_l \in P$, and consider the set of all rectangles whose left and right extremes are defined by this pair. Choose a third point $p_b \in P$ in between these two; this point defines the bottom edge of the rectangle if it is below otherwise one of p_r, p_l does. Now let a horizontal line segment spanning the rectangle sweep the plane upwards starting from p_b. Every time the sweep line encounters a point, we update m_R, b_R in constant time and recompute the discrepancy, maintaining the largest value. Each sweep takes linear time, and there are $O(n^3)$ choices of triples (p_l, p_r, p_b). Thus, the algorithm runs in time $O(n^4)$. We can make a qualified improvement of this runtime with the following lemma. Let $G = M/B$.

Lemma 4.4. *Each point $p \in R = r \cap P$ in the maximum discrepancy rectangle r with the maximum or minimum x- or y-coordinate must satisfy $m(p)/b(p) \geq G$.*

Proof. If a single point p is inserted or deleted into R, then either the boundary must be extended so one of these points is no longer extremal, or the boundary must be retracted so one of these points is excluded, and another point is extremal. Then if

$$\frac{m(p)/M}{b(p)/B} < \frac{m_R}{b_R} = \frac{m(R)/M}{b(R)/B},$$

then removing p from R shifts (m_R, b_R) towards the line $x = y$ in S_n where the discrepancy functions is 0, and because the discrepancy function is convex the new value must not increase. Likewise, because adding that point would not increase the discrepancy value, removing it would also not decrease the discrepancy. Finally we note that for the maximal discrepancy rectangular range R, $m(R)/b(R) > G$, implying that $m(p)/b(p) > G$ for any p defining a boundary of r. □

Let P_G be the set of points such that $p \in P_G$ satisfies $m(p)/b(p) \geq G$. Let $n_G = |P_G|$. Using Lemma 4.4 we can now restrict out search to $O(n_G^4)$ rectangles. Thus we have only $O(n_G^3)$ choices of (p_l, p_r, p_b) such that $p_l, p_r, p_b \in P_G$. The inner sweep line still requires $O(n)$ time to count the number the baseline and measured values of all points in R, so the algorithm runs in $O(n_G^3 n)$ time. The details are presented in Algorithm 4.3.

Algorithm 4.3.1 Algorithm Exact

```
maxd = -1
Sort all points by y-coordinate.
for all pairs of points $(p_l, p_r)$ in $P_G$ do
  for $i = 1$ to $n_G$ do
    Let $p_b \in P_G$ be the point with $i^{\text{th}}$ smallest y-coordinate and the point in $P$
    with the $k^{\text{th}}$ smallest $y$-coordinate.
    $m = 0, b = 0$
    for $j = k + 1$ to $n$ do {This is the sweep line}
      Let $p$ be point with $j^{\text{th}}$ smallest y-coordinate
      $m = m + m(p), b = b + b(p)$
      $d = d(m/M, b/B)$.
      if ($d >$ maxd) then
        maxd = $d$
```

4.4 Algorithms for Linear Discrepancy

We begin by summarizing results from Dobkin *et al.* [40]. A discrepancy maximizing rectangle has minimal and maximal points in the x and y dimensions. These four points fully describe the rectangle. If we specify the minimizing and maximizing x coordinates, the problem is reduced to a one-dimensional problem of all points within the slab this defines, as in Exact. By maintaining the maximal discrepancy interval in the one-dimensional problem under point insertion, only $O(n^2)$ insertions are necessary to check the maximum discrepancy interval over all possible slabs, and thus over all possible rectangles.

The one-dimensional problem is solved by building a binary tree of intervals. A node corresponding to interval $I = (i, l, r)$ stores the subinterval i of maximal discrepancy, the interval l of maximal discrepancy that includes the left boundary of I, and the interval r of maximal discrepancy that includes the right boundary of I. Two adjacent nodes, $I_{\mathsf{left}} : (i_{\mathsf{left}}, l_{\mathsf{left}}, r_{\mathsf{left}})$ and $I_{\mathsf{right}} : (i_{\mathsf{right}}, l_{\mathsf{right}}, r_{\mathsf{right}})$, can be merged to form a single nodes $I : (i, l, r)$ in constant time. i is assigned the interval with the maximum discrepancy out of the set $\{i_{\mathsf{left}}, i_{\mathsf{right}}, l_{\mathsf{right}} \cup r_{\mathsf{left}}\}$. l is assigned the interval with the maximum discrepancy out of the set $\{l_{\mathsf{left}}, I_{\mathsf{left}} \cup l_{\mathsf{right}}\}$, and r is assigned symmetrically to l. The entire interval, $[0, 1] = I : (i, l, r)$, is at the root of the tree, and the interval which maximizes the discrepancy is i.

Lemma 4.5 (Dobkin *et al.* [40]). *Bichromatic discrepancy for a set of red and blue points in the plane can be computed in time $O(n^2 \log n)$.*

Proof. We refer the reader to [40] for full details, but summarize their proof below for completeness.

By maintaining the maximal discrepancy interval in the one-dimensional problem under point insertion, only $O(n^2)$ insertions are necessary to check the maximum discrepancy interval over all possible slabs, and thus over all possible rectangles. Fix the left boundary $O(n)$ times, and for each increase the right boundary by one insertion $O(n)$ times.

Adding a point to the binary tree of intervals requires traversing and then merging $O(\log n)$ intervals if the tree is balanced. The tree can be kept balanced through rotations which only require a constant number of merge operations each. □

Alternatively, we can use a heap instead of a binary tree if the number of points is known in advance. Now instead of inserting points and rebalancing the tree, the points can be presorted and just be marked as active or inactive in the intervals.

4.4.1 Extending to Linear Discrepancy over S_n

We now describe how to extend Dobkin *et al.*'s algorithm [40] to handle any linear discrepancy function and also to only consider regions with at least a constant C amount of data from functions m and b. We only need to modify the algorithm for handling intervals.

Define $\mathcal{I}_C$ to be the set of all intervals such that the sums $\sum_{p\in I} m_0(p) \geq C$ and $\sum_{p\in I} b_0(p) \geq C$. For each interval $I : (i, \bar{l}, \bar{r})$, i must be in $\mathcal{I}_C$ and $\bar{l}$ and $\bar{r}$ must represent sets of intervals $\{l_1 \ldots l_k\}$ and $\{r_1 \ldots r_h\}$, respectively. l_k (resp. r_h) is the interval in $\mathcal{I}_C$ which contains the left (resp. right) boundary that has the maximum discrepancy. l_1 (resp. r_1) is the interval which contains the left (resp. right) boundary that has the maximum discrepancy. For all γ l_γ includes the left boundary and $|l_\gamma| < |l_j|$ for all $\gamma < j$. Also $\sum_{p\in l_\gamma} m_0(p) < C$ or $\sum_{p\in l_\gamma} b_0(p) < C$ for all $\gamma < k$. Finally, the set $\bar{l}$ must contain all local maximum; if l_γ were to gain on lose one point, the discrepancy would decrease. The restrictions are the same for all r_γ, except these intervals must contain the right boundary.

With this more complicated data structure we can now perform a merge between two adjacent intervals $I_{\mathsf{left}} : (i_{\mathsf{left}}, \bar{l}_{\mathsf{left}}, \bar{r}_{\mathsf{left}})$ and $I_{\mathsf{right}} : (i_{\mathsf{right}}, \bar{l}_{\mathsf{right}}, \bar{r}_{\mathsf{right}})$. Computing the maximum discrepancy interval in $\mathcal{I}_C$ can be done by checking i_{left} and i_{right} versus all pairs from $\bar{l}_{\mathsf{right}} \times \bar{r}_{\mathsf{left}}$ such that $\sum m(p) \geq C$ and $\sum b(p) \geq C$. By the local optimality restriction, the optimal interval in $\mathcal{I}_C$ spanning the boundary must have one endpoint in each set. Updating $\bar{l}$ and $\bar{r}$ can be done by just pruning from $\bar{l}_{\mathsf{left}}$ and $\bar{I}_{\mathsf{left}} \times \bar{r}_{\mathsf{left}}$ according to the restrictions for $\bar{l}$, and similarly for $\bar{r}$.

Theorem 4.3. *Let $\mathcal{R}'$ be the set of all rectangular ranges $R \in \mathcal{R}'$ such that both $\sum_{p\in R} m(p)$ and $\sum_{p\in R} b(p)$ are greater than the constant C. Then any linear discrepancy function of the form $a_1 \sum m(p) + a_2 \sum b(p) + a_3$ can be maximized over this set in $O(n^2 \log n)$ time.*

Proof. Because a_3 is constant for all intervals, it can be ignored. Thus the linear function has the form of $d(m_R, b_R) = \sum_{p\in R} \chi(p)$. The algorithm of [40] relies only on the fact that the discrepancy function is additive, and hence can be extended to the above discrepancy function by only modifying the intervals and merge operation in the one-dimensional case.

The local optimality restriction on the sets of intervals ensures $\sum_{p\in l_i} m_0(p) < \sum_{p\in l_{i+1}} m_0(p)$ and $\sum_{p\in l_i} b_0(p) < \sum_{p\in l_{i+1}} b_0(p)$. If either sum ($\sum m_0(p)$ or $\sum b_0(p)$) increases then the other must also increase or this would violate the local optimality condition. An increase of just a measure that increases discrepancy will cause the previous interval not to be optimal and an increase in just a measure that causes the discrepancy to decrease will cause the latter interval not to be optimal. Thus k and h are constants bounded by the number of p required for $\sum m_0(p) \geq C$ and $\sum b_0(p) \geq C$. Thus each interval in the tree structure stores a constant amount of information.

Finally, a merge between two adjacent intervals $I_{\mathsf{left}} : (i_{\mathsf{left}}, l_{\mathsf{left}}, r_{\mathsf{left}})$ and $I_{\mathsf{right}} : (i_{\mathsf{right}}, l_{\mathsf{right}}, r_{\mathsf{right}})$ also can be done in constant time. There are a constant number of intervals to check in computing the maximum discrepancy interval and the new interval is also a constant size after the pruning step. Thus a point can be added to the balanced tree in $O(\log n)$ time. Hence, the maximum discrepancy rectangular range R such that $m(R) > C$ and $b(R) > C$ for any linear discrepancy function can be found in $O(n^2 \log n)$ time. □

A similar argument applies if we consider prospective discrepancy, or higher dimensions.

Lemma 4.6. *A linear discrepancy function can be maximized over prospective rectangles in $O(n^3 \log n)$ time, or it can be maximized over axis-parallel hyper-rectangles in d-dimensions in time $O(n^{2d-2} \log n)$.*

Using Lemma 4.4 (since linear functions are convex) these bounds can be improved in practice. We only need to consider slabs which are bounded with points from P_G; however, we still need to run the final sweep line in full to maintain the interval data structure. Thus linear functions can be maximized over axis-parallel rectanges in d-dimensions in time $O(n_G^{2d-3} n \log n)$, and with an extra $O(n_G)$ factor for prospective rectangles.

4.4.2 Bounds for Specific Discrepancy Functions

Invoking Theorem 4.1 and Theorem 4.3 and we can state the following theorems. We refer to the associated algorithm as Approx-Linear.

Theorem 4.4. *An additive ε-approximation to the maximum discrepancy d over all axis-aligned rectangles containing at least a constant measure can be computed in time $O((1/\varepsilon)k_x(d)k_y(d)n^2 \log n)$. With respect to prospective time windows, the corresponding maximization takes time $O((1/\varepsilon)k_x(d)k_y(d)n^3 \log n)$.*

We get the following corollaries:

Corollary 4.1. *An additive ε-approximation to the maximum discrepancy d_P or d_γ over all axis-aligned rectangles containing at least a constant measure can be computed in time $O((1/\varepsilon)n^2 \log^3 n)$. With respect to prospective time windows, the corresponding maximization takes time $O((1/\varepsilon)n^3 \log^3 n)$.*

Corollary 4.2. *An additive ε-approximation to the maximum discrepancy d_B over all axis-aligned rectangles containing at least a constant measure can be computed in time $O((1/\varepsilon)n^{2.5} \log^2 n \log\log\log n)$. With respect to prospective time windows, the corresponding maximization takes time $O((1/\varepsilon)n^{3.5} \log^2 n \log\log\log n)$.*

Corollary 4.3. *An additive ε-approximation to the maximum discrepancy d_G over all axis-aligned rectangles containing at least a constant measure can be computed in time $O((1/\varepsilon)n^3 \log n)$. With respect to prospective time windows, the corresponding maximization takes time $O((1/\varepsilon)n^4 \log n)$.*

4.5 Sampling Algorithms

We now present another algorithm that finds an additive ε-approximation to the maximum linear discrepancy. It is based upon first appropriately constructing ε-samples.

Theorem 4.5. *For a linear discrepancy function $d_l(m_R, b_R) = \alpha m_R + \beta b_R + \gamma$ let $\tau = \max\{\alpha, |\beta|\}$. There exists an algorithm running in time $O(n\tau^3/\varepsilon^3 \log^{12}(\tau/\varepsilon))$ that returns an estimate E such that $|E - \max_{R \in \mathfrak{R}} d_l(m_R, b_R)| \leq \varepsilon$.*

Proof. We construct a weighted $(\varepsilon/2\tau)$-sample $Q \subset P$ of $(P, \mathfrak{R}_2)$, using Corollary 1.3, of size $O((\tau/\varepsilon)\log^4(\tau/\varepsilon))$, such that for all $R \in \mathfrak{R}_2$

$$\left| m_R - \frac{|Q \cap R|}{|Q|} \right| \leq \frac{\varepsilon}{2\alpha} \text{ and } \left| b_R - \frac{|Q \cap R|}{|Q|} \right| \leq \frac{\varepsilon}{2|\beta|} \quad .$$

This takes $O(n\tau^3/\varepsilon^3 \log^{12}(\tau/\varepsilon))$ time. We can now estimate d_l for any α, β, γ with

$$\tilde{d}_l(m_R, b_R) = \alpha\frac{|Q \cap R|}{|Q|} + \beta\frac{|Q \cap R|}{|Q|} + \gamma$$

such that for any $R \in \mathfrak{R}_2$ we have $|d_l(m_R, b_R) - \tilde{d}_l(m_R, b_R)| \leq \varepsilon$. $\square$

A related algorithm can be used when the data is known to lie on a grid.

To approximate a function $d = \{d_P, d_G, d_B, d_\gamma\}$ using Lemma 4.3 we need to bound $\tau = \max(\alpha, |\beta|) = \max_{x,y \in S_n} \max\{f_x(x,y), |f_y(x,y)|\}$ where $f_x = df/dx$ and $f_y = d_f/dy$. These quantities are defined in Section 4.B. For d_P and d_γ $\tau = O(n)$ so the ε-approximation maximum discrepancy can be solved in $O(n^2/\varepsilon^3 \log^{11}(n/\varepsilon))$ time by Corollary 4.1 and using $O(n^4/\varepsilon^3 \log^{12}(n/\varepsilon))$ time for Theorem 4.5. This does not improve on the standard application of Corollary 4.1. For d_G $\tau = O(n^2)$ so the ε-approximation of maximum discrepancy can be solved in $O(n^6/\varepsilon^4 \log^{13}(n/\varepsilon))$ time by Corollary 4.3 and using $O(n^7/\varepsilon^3 \log^{12}(n/\varepsilon))$ time for Theorem 4.5. This also does not improve on the standard application of Corollary 4.3. For d_B $\tau = O(\log n)$ so the ε-approximation of maximum discrepancy can be solved in $O((1/\varepsilon^{3.5}) \log^{2.5} n \cdot \log^{3.5}((1/\varepsilon)\log n) \log\log\log((1/\varepsilon)\log n))$ time by Corollary 4.2 and using $O(n/\varepsilon^3 \cdot \log^3 n \log^{12}((1/\varepsilon)\log n))$ for Theorem 4.5. This is a large improvement over the standard application of Corollary 4.2.

Theorem 4.6. *An additive ε-approximation to the maximum discrepancy d_B over all axis-aligned rectangles containing at least a constant measure can be computed in time $O(n/\varepsilon^3 \log^3 n \log^{12}((1/\varepsilon)\log n))$.*

We can choose $O((\tau/\varepsilon^2)\log(1/\delta))$ points at random in $O(n)$ time using, for instance, an algorithm of Knuth [68]. Thus Theorem 4.6 can be improved to run in $O(n + \mathsf{poly}(1/\varepsilon, \log n, \log 1/\delta))$ if we accept δ probability of failure.

Theorem 4.7. *An additive ε-approximation to the maximum discrepancy d_B over all axis-aligned rectangles containing at least a constant measure can be computed in $O(n + (1/\varepsilon^6)\log^{2.5} n \log^{2.5}(1/\delta) \cdot \log^2((1/\varepsilon)\log(n/\delta)) \log\log\log((1/\varepsilon)\log(n/\delta)))$ time, with probability $1 - \delta$.*

4.5.1 Streaming Algorithms

In this section we consider algorithms for the data stream model [60, 43, 14]. Here the data points arrive in some, possibly adversarial, order. An algorithm in the streaming model has limited space, S, to catalog all the points in the stream.

We can use Suri *et al.* [112]'s streaming algorithm for ε-samples, which if the total size of the stream is n and known in advance, are of size $O((1/\varepsilon)\log^2(\varepsilon n) \cdot \log^{2d}((1/\varepsilon)\log(\varepsilon n))$. If the size of the stream is now known in advance, then a randomized algorithm exists that requires $O((1/\varepsilon)\log^5(1/\varepsilon))$ space and is correct with probability 1/2. We summarize the application of these results for d_B.

Theorem 4.8. *For a stream with n elements, there exists a deterministic streaming algorithm for maintaining an ε-additive approximation to d_B using $O((1/\varepsilon)\log n \cdot \log^2(\varepsilon n/\log n)\log^4((1/\varepsilon)\log n \log(\varepsilon n/\log n))$ space when the size of the stream is known in advance, and a randomized streaming algorithm that requires $O((1/\varepsilon)\log n \cdot \log^5((1/\varepsilon)\log n))$ space and is correct with probability 1/2.*

Similar results for d_l require $O((\tau/\varepsilon)\log^2(\varepsilon n/\tau)\log^4((\tau/\varepsilon)\log(\varepsilon n/\tau))$ space for a deterministic algorithm and $O((\tau/\varepsilon)\log^5(\tau/\varepsilon))$ space for a randomized algorithm.

Lower bounds. As is typical for lower bounds in the stream model, our lower bounds are proved using reductions from communication complexity problems [73]. We denote $C_\delta(f)$ as the δ-error randomized communication complexity of function f. Also, let $C_\delta^{\text{1-way}}(f)$ be the one-way δ-error randomized communication complexity of function f.

Definition 1 (Indexing). *There are two player Y_1 and Y_2. Y_1 has an n bit string x and Y_2 has an index $j \in [n]$. The* indexing *function returns* $\text{INDEX}(x, j) = x_j$.

Definition 2 (Multi-Party Set Disjointness). *There are t players $Y_1, \ldots, Y_t$. Y_i has an n bit string x^i. The t*-party set disjointness *[17] function returns* $\text{DISJ}_{n,t}(x^1, \ldots, x^t) = \bigvee_{j=1}^{n} \bigwedge_{i=1}^{t} x_j^i$.

Lemma 4.7. *For any $0 < \delta < 1/4$, $C_\delta^{\text{1-way}}(\text{INDEX}_n) = \Omega(n)$. The result remains true for instances (x, j) where x has exactly $n/2$ entries which are 1.*

Lemma 4.8 (Chakrabarti *et al.* [27]). *For any $0 < \delta < 1/4$,*

$$C_\delta(\text{DISJ}_{n,t}) = \Omega\left(\frac{n}{t \log t}\right) \quad .$$

This result remains true for the following family $\mathcal{F}$ of instances $(x^1, \ldots x^t)$ satisfying

$$|\{j : x_j^i = 1\}| = n/2t \qquad \forall i \in [t] \tag{4.1}$$
$$|\{i : x_j^i = 1\}| \in \{0, 1, t\} \qquad \forall j \in [n] \tag{4.2}$$
$$|\{j : |\{i : x_j^i = 1\}| = t\}| \leq 1 \qquad . \tag{4.3}$$

For a linear discrepancy function,

$$d_l(m_R, b_R) = \alpha \cdot m_R + \beta \cdot b_R + \gamma \quad (\alpha > 0, \beta < 0)$$

we make the assumptions that $m_0 : P \to \mathbb{N}$, $b_0 : P \to \mathbb{N}$, that $m^* = \max_{p \in P} m_0(p)$ is constant and $\forall_{p \in P} b_0(p) = c$, for some constant c. As a preprocessing step to any algorithm, we construct two point sets P_m and P_b: for each $p \in P$ place $m_0(p)$ copies of the point in P_m and $b_0(p)$ copies of the point in P_b. For each $p \in P_m$ let $m_0(p) = 1$ and $b_0(p) = 0$. Similarly, for each $p \in P_b$ let $m_0(p) = 0$ and $b_0(p) = 1$. Henceforth we will refer to a point p being colored *red* if $p \in P_m$, or *blue* if $p \in P_b$. Note that $|P_m \cup P_b| = O(n)$ and that this construction can be achieved in $O(n)$ time. Finally note that discrepancy for any $R \in \mathcal{A}$ is the same with respect to P as it is to $P_m \cup P_b$.

We will also consider the problem of maximizing *numerical discrepancy* $d_\#$ defined for a range $R \subseteq P$ as

$$d_\#(R) = m_R - b_R \ ,$$

which basically measures the percentage of red points in a range versus the percentage of total points in the range. Note that $d_\#$ is a form of linear discrepancy.

Theorem 4.9. *Any* P *pass streaming algorithm returning a* t *relative approximation to the numerical discrepency with probability at least* $3/4$ *requires* $\Omega(n/(t^6 \mathsf{P} \log t))$ *space. Alternatively, any* P *pass streaming algorithm returning an* ε *additive approximation with probability at least* $3/4$ *requires* $\Omega(1/(\varepsilon \mathsf{P}))$ *space.*

Proof. Let $(x^1, \ldots, x^{t'}) \in \mathcal{F}$ be an instance of $\textsc{disj}_{n',t'}$ where $n' = n/(3t^2)$ and $t' = 3t^2$. We will show how to transform $(x^1, \ldots, x^{t'})$ into a size $n't' = n$ instance of the numerical discrepancy problem such that t-approximating the maximum numerical discrepancy problem determines the value of $\textsc{disj}_{n',t'}$.

We only need to consider a set of points $P \in \mathbb{R}^1$, thus the ranges are defined by intervals. Also, we can then order all the points we will see in P along $\mathbb{R}^1$ and for simplicity, place them on a grid so every pair of consecutive points in $\mathbb{R}^1$ are a unit distance apart. Assume that we begin the stream with all points in P_b, then since each $p \in P$ has $b_0(p) = 1$, all grid points have $b_0 = 1$. We then just need to find intervals where many points from P_m concentrate.

The rest of the stream we define consists of $n't'$ elements $E = P_m$ where elements will come from a universe $[n'(t'+1)] = P_b$. We partition the universe into intervals $I_1, \ldots, I_{n'}$ where $I_i = [(i-1)(t'+1)+1, i(t'+1)]$. Each player Y_i determines a size n' subset of the elements,

$$\begin{aligned} E_i &= \{(i-1)(t'+1)+j+1 : x^i_j = 0, j \in [n']\} \\ &\quad \cup \{(i-1)(t'+1)+1 : x^i_j = 1, j \in [n']\} \ . \end{aligned}$$

Note that every region contains t' elements from E. We next show how the maximum discrepancy of the set depends on the value of $\textsc{disj}_{n',t'}$.

1. If $\textsc{disj}_{n',t'} = 1$ then the maximum numerical discrepancy is at least

$$\frac{t'}{n't'} - \frac{1}{n'(t'+1)} = \frac{t'}{n'(t'+1))} \ ,$$

   since there exists an element with multiplicity t'.

2. If $\textsc{disj}_{n',t'} = 0$ then each element occurs at most once. Consider any interval $I \subseteq [n'(t'+1)]$. The numerical discrepancy in any I_i is exactly 0. Furthermore, the numerical discrepancy in any subinterval of I whose length $l \le t'$ is at most

$$\frac{l}{n't'} - \frac{l}{n'(t'+1)} = \frac{l}{n't'(t'+1)} \ .$$

   Hence the numerical discrepancy in interval I is at most $2/(n'(t'+1))$.

Hence, if an algorithm disambiguates between the maximum numerical discrepancy being greater than $t'/(n'(t'+1))$ or less than $2/(n'(t'+1))$ then the value of $\textsc{disj}_{n',t'}$ is also determined. Therefore, a relative approximation better than $\sqrt{t'/2} > t$ determines $\textsc{disj}_{n',t'}$.

Assume that there exists a P pass algorithm $\mathcal{M}$ that returns a t relative approximation to the maximum numerical discrepancy of n points (with probability at least $3/4$) and uses at most $S(n,t)$ bits of memory. This algorithm gives rise to a communication protocol for $\textsc{disj}_{n',t'}$ as follows. Let the stream be ordered as $E_1, E_2, \ldots, E_{t'}$. Let $m_{i,j}$ be the memory state of $\mathcal{M}$ after the last elements from E_i has gone past in the j pass. Each player Y_i constructs E_i from x^i. Y_1 runs $\mathcal{M}$ on E_1 and sends the memory state $m_{1,1}$ to Y_2. Y_2 initializes $\mathcal{M}$ with memory state $m_{1,1}$, runs $\mathcal{M}$ on E_2 and sends the memory state, $m_{1,2}$, to Y_3. They continue in this way where $m_{i,j}$ is the $(i + t'(j-1))$th message sent. The memory state $m_{t',\mathsf{P}}$ determines a t approximation to the maximum discrepancy and, therefore, the value of $\textsc{disj}_{n',t'}$. Each message is at most $S(n,t)$ bits long and there are at most $t'\mathsf{P}$ messages. Hence the total communication is $O(t'S(n,t)\mathsf{P})$ bits. By appealing to Theorem 4.8, we deduce that,

$$S(n,t) = \Omega\left(\frac{n'}{t'^2\mathsf{P}\log t'}\right) = \Omega\left(\frac{n}{t^6\mathsf{P}\log t}\right) \quad .$$

The second lower bound uses a similar reduction to the first except that $t' = 3, n' = 1/(8\varepsilon)$ and every point in the above construction is replaced by $8\varepsilon n/3$ identical points. □

Note that the above result also applies to approximating the maximum linear discrepancy where $\alpha = -\beta = 1$. Although the data in this lower bound lies on the grid, it applies when the data need not lie on a grid; shifting each point slightly gives the same discrepancy values.

Corollary 4.4. *Any P pass streaming algorithm returning a t relative approximation to the maximum linear discrepency with probability at least $3/4$ requires $\Omega(n/(t^6\mathsf{P}\log t))$ space. Alternatively, any P pass streaming algorithm returning an ε additive approximation with probability at least $3/4$ requires $\Omega(1/(\varepsilon\mathsf{P}))$ space.*

The next lower bound gives a dependence on β when approximating the maximum linear discrepancy.

Theorem 4.10. *Any one pass streaming algorithm that ε additively approximates the maximum linear discrepancy with probability at least $3/4$ requires $\Omega(|\beta|/\varepsilon)$ space.*

Proof. Consider an instance (x,j) of $\textsc{index}_{|\beta|/\varepsilon}$. Let $w = |\beta|/(2\varepsilon)$ be the number of 1's in x. We will show how to transform (x,j) into a size $n+1$ instance of the linear discrepancy problem such that an additive ε-approximation of the maximum linear discrepancy problem determines the value of $\textsc{index}_{|\beta|/\varepsilon}(x,j)$.

The stream starts with elements determined by Y_1: for each $i \in [|\beta|/\varepsilon]$ such that $x_i = 1$ there are two blue points with value i. The stream ends with one red point j. Note that the maximum value of $\alpha m_R + \beta b_R + \gamma$ is $\alpha + \gamma$ if $\textsc{index}_{|\beta|/\varepsilon}(x,j) = 0$ and is $\alpha - 2\varepsilon + \gamma$ if $\textsc{index}_{|\beta|/\varepsilon}(x,j) = 1$.

Then, by appealing to Theorem 4.7, we deduce that the space required is $\Omega(|\beta|/\varepsilon)$. □

Note that these lower bounds hold even if we restrict that every range contains at least a constant C number of points.

4.6 Grid Algorithms

As we mentioned earlier, algorithms like those presented in [90, 89] aggregate data to a regular $g \times g$ grid. Since such a grid contains g^2 points, one can run any of the above mentioned algorithms, setting $n = g^2$. However, this is very inefficient, and ignores the special structure of the grid. For example, algorithm Exact would then run in time $O((g^2)^4) = O(g^8)$. In this section, we present two algorithms that take advantage of grid structured data.

4.6.1 Exact-Grid Algorithm

The first algorithm returns the maximum discrepancy rectangle in time $O(g^4)$. It is quite similar to the algorithm of Section 4.3, using a sweep line to explore the space of rectangles. The basic idea is as follows. We maintain four sweep lines, two horizontal and two vertical. The two vertical sweep lines move from left to right. At any moment, one of them is at x position i, and the other at x position $j > i$. As the second sweep line moves from i to the right most position, we maintain a count, for each row, of the total measured and baseline mass in this row between i and j. This can be done in time $O(g)$ for each move of the second sweep line. Once the two vertical sweep lines are fixed, two horizontal sweep lines move from bottom to top.

Algorithm 4.6.1 Algorithm Exact-Grid: Input is $g \times g$ grid with values $m(i, j), b(i, j)$

```
for $i = 1$ to $g$ do {Left sweep line}
  Initialize $m[y] = m(i, y), b[y] = b(i, y)$ for all $y$
  for $y = 2$ to $g$ do
    $m[y] += m[y-1], b[y] += b[y-1]$
  for $j = i + 1$ to $g$ do {Right sweep line}
    $m = 0, b = 0$
    for $y = 1$ to $g$ do
      $m += m(j, y), b += b(j, y), m[y] += m, b[y] += b$
    for $k = 1$ to $g$ do {Bottom sweep line}
      for $l = k$ to $g$ do {Top sweep line}
        if $k = 1$ then
          $m = m[k], b = b[k]$
        else
          $m = m[l] - m[k-1], b = b[l] - b[k-1]$
        if $(d(m/M, b/B) > \max)$ then
          $\max = d(m/M, b/B)$
```

Since we maintain counts of the total mass in each row, the discrepancy function for the range bounded by the four sweep lines can be computed in constant time every time the higher horizontal sweep line is moved. A detailed description is presented in Algorithm 4.6.1.

Using Lemma 4.4, this algorithm can be sped up in practice by only running the top sweep line if the cells in the bottom one satisfy $m_k/b_k \geq G$ where m_k is the sum of the measured data in those cells and b_k is the sum of the baseline data in those cells.

4.6.2 Approx-Grid Algorithm

Our second algorithm is approximate, and builds upon the approximate schemes of Theorem 4.4. In all our approximate schemes, the main subroutine is an $O(n^2 \log n)$ time algorithm for maximizing a linear discrepancy function over the space of all axis-parallel rectangles. It is easy to extract from this algorithm an $O(n)$ algorithm Linear1D-Grid for finding the interval in one dimension that maximizes any linear discrepancy function. Naively transferring the algorithm over rectangles to the grid would yield an algorithm running in time $O(g^4 \log g)$. We improve this to $O(g^3)$. In brief, the $O(n^2 \log n)$ procedure in Section 4.4 uses two horizontal sweep lines going from bottom to top. For any position of the two sweep lines, the maximum discrepancy rectangle among rectangles bounded by these lines can be found by projecting all points onto the lower sweep line and solving a one-dimensional problem (the resulting interval defines the x-extents of the optimal rectangle). In the modified grid variant, we maintain two arrays $m[]$, $b[]$, each of size g, such that $m[i]$ stores the sum of all values $m(i, j)$ between the lower and upper sweep lines. Note that this can be maintained in constant time per entry as the upper sweep line moves. For each such movement, we run Linear1D-Grid on the values of $m[]$ and $b[]$. The total running time is therefore g positions of the bottom sweep line $\times$ g positions of the top sweep line $\times$ $O(g)$ for updating counts and running Linear1D-Grid, for a

Algorithm 4.6.2 Algorithm Linear-Grid: Input is $g \times g$ grid with values $m(i, j), b(i, j)$, and linear function ℓ

```
maxd = $-1$
**for** $i = 1$ to $g$ **do** {Left sweep line}
  Initialize $m[y] = m(i, y), b[y] = b(i, y)$ for all $y$
  **for** $j = i + 1$ to $g$ **do** {Right sweep line}
    **for** $y = 1$ to $g$ **do**
      $m[y] += m(j, y), b[y] += b(j, y)$
    $(d, y_b, y_t)$ = Linear1D($\ell, m[], b[]$).
    **if** ($d >$ maxd) **then**
      maxd = $d$; $r = [i, j] \times [y_b, y_t]$
**return** (maxd, $r$)
```

net running time of $O(g^3)$.

We describe the algorithm in detail in two parts. First we give the $O(g^3)$ gridded algorithm for linear discrepancy functions on a grid: Algorithm 4.6.2. This algorithm is then used as the core subroutine in Algorithm 4.6.3.

Algorithm 4.6.3 Algorithm APPROX-GRID

```
maxd = $-1$
Use Theorem 4.1 to construct a family $\mathcal{L} = \{\ell_1, \dots, \ell_t\}$ of $t$ linear functions.
for $i = 1$ to $t$ do
    $(d, r_i)$ = LINEAR-GRID $(m[], b[], \ell_i)$.
    $d_i = d(r_i)$
    if ($d_i >$ maxd) then
        maxd = $d_i$; $r' = r_i$
```

The runtime of APPROX-GRID is $O((1/\varepsilon)g^3 \log^2 g)$ for d_P and d_γ, since there are $t = O((1/\varepsilon)\log^2 g)$ calls of LINEAR-GRID which runs in $O(g^3)$.

Appendix to Chapter 4

4.A One-parameter Exponential Families

In this section we derive a general expression for a likelihood-based discrepancy measure for the one-parameter exponential family. Many common distributions like the Poisson, Bernoulli, Gaussian and gamma distributions are members of this family. Subsequently we will derive specific expressions for the above mentioned distribution families.

Definition 4.1 (One-parameter exponential family). *The distribution of a random variable y belongs to a one-parameter exponential family (which we denote by $y \sim \mathsf{1EXP}(\eta, \phi, T, B_e, a)$) if it has probability density given by*

$$f(y;\eta) = C(y,\phi)e^{\frac{\eta T(y) - B_e(\eta)}{a(\phi)}}$$

where $T(\cdot)$ is some measurable function, $a(\phi)$ is a function of some known scale parameter $\phi > 0$, η is an unknown parameter (called the natural parameter), and $B_e(\cdot)$ is a strictly convex function. The support $\{y : f(y;\eta) > 0\}$ is independent of η.

It can be shown that the expected value $E_\eta(T(Y)) = B_e^{'}(\eta)$ and the variance $\mathrm{Var}_\eta(T(Y)) = a(\phi)B_e^{''}(\eta)$. In general, $a(\phi) \propto \phi$.

Let $\bar{y} = \{y_i : i \in R\}$ denote a set of $|R|$ variables that are independently distributed with $y_i \sim \mathsf{1EXP}(\eta, \phi_i, T, b, a)$, $(i \in R)$. The joint distribution of $\bar{y}$ is given by

$$f(\bar{y};\eta) = \prod_{i\in R} C(y_i,\phi_i)e^{\frac{\eta T^*(\bar{y}) - B_e(\eta)}{\phi^*}}$$

where $1/\phi^* = \sum_{i\in R}(1/a(\phi_i))$, $v_i = \phi^*/a(\phi_i)$, and $T^*(\bar{y}) = \sum_{i\in R}(v_i T(y_i))$.

Given data $\bar{y}$, the *likelihood* of parameter η is the probability of seeing $\bar{y}$ if drawn from a distrbution with parameter η. This function is commonly expressed in terms of its logarithm, the *log-likelihood* $l(\eta;\bar{y})$, which is given by (ignoring constants that do not depend on η)

$$l(\eta;\bar{y}) = (\eta T^*(\bar{y}) - B_e(\eta))/\phi^* \tag{4.4}$$

and depends on data only through T^* and ϕ^*.

Lemma 4.9. *Let $\bar{y} = (y_i : i \in R)$ be independently distributed according to $y_i \sim$ 1EXP(η, ϕ_i, T, b, a). Then, the maximum likelihood estimate (MLE) of η is $\hat{\eta} = g_e(T^*(\bar{y}))$, where $g_e = (B_e^{'})^{-1}$. The maximized log-likelihood (ignoring additive constants) is $l(\hat{\eta};\bar{y}) = (T^*(\bar{y})g_e(T^*(\bar{y})) - B_e(g_e(T^*(\bar{y}))))/\phi^*$.*

Proof. The MLE is obtained by maximizing (4.4) as a function of η. Since B_e is strictly convex, $B_e^{'}$ is strictly monotone and hence invertible. Thus, the MLE obtained as a solution of $l(\eta;\bar{y})^{'} = 0$ is $\hat{\eta} = (B_e^{'})^{-1}(T^*(\bar{y})) = g_e(T^*(\bar{y}))$. The second part is obtained by substituting $\hat{\eta}$ in (4.4). $\square$

The likelihood ratio test for outlier detection is based on the following premise. Assume that data is drawn from a one-parameter exponential family. For a given region R_1 and its complement R_2, let η_{R_1} and η_{R_2} be the MLE parameters for the data in the regions. Consider the two hypothesis $H_0 : \eta_{R_1} = \eta_{R_2}$ and $H_1 : \eta_{R_1} \neq \eta_{R_2}$. The test then measures the ratio of the likelihood of H_1 versus the likelihood of H_0. The resulting quantity is a measure of the strength of H_1; the larger this number is, the more likely it is that H_1 is true and that the region represents a true outlier. The likelihood ratio test is *individually* the test with most statistical power to detect the region of maximum discrepancy and hence is optimal for the problems we consider. A proof of this fact for Poisson and Bernoulli distributions is provided by Kulldorff [70] and extends to 1EXP without modification.

We are now ready to state the main theorem of this section.

Theorem 4.11. *Let $\hat{y}_{R_j} = (y_{R_j i} : i \in R_j)$ be independently distributed with $y_{R_j i} \sim$ 1EXP(η_{R_j},$\phi_{R_j i}$,T, B_e, a), for $j = 1, 2$. The log-likelihood ratio test statistic for testing $H_0 : \eta_{R_1} = \eta_{R_2}$ versus $H_1 : \eta_{R_1} \neq \eta_{R_2}$ is given by*

$$\Delta = \kappa(G_{R_1}, \Phi_{R_1}) + \kappa(G_{R_2}, \Phi_{R_2}) - \kappa(G, \Phi) \tag{4.5}$$

where $\kappa(x, y) = (xg_e(x) - B_e(g_e(x)))/y$, $G_{R_j} = T^*(\hat{y}_{R_j}), 1/\Phi_{R_j} = \sum_{i \in R_j}(1/a(\phi_{R_j i}))$, $1/\Phi = 1/\Phi_{R_1} + 1/\Phi_{R_2}$, $b_{R_1} = \frac{1/\Phi_{R_1}}{(1/\Phi_{R_1}+1/\Phi_{R_2})}$ and $G = b_{R_1}G_{R_1} + (1 - b_{R_1})G_{R_2}$.

Proof. The likelihood ratio is given by

$$\frac{\sup_{\eta_{R_1} \neq \eta_{R_2}} L(\eta_{R_1}, \eta_{R_2}; \hat{y}_{R_1}, \hat{y}_{R_2})}{\sup_{\eta} L(\eta; \hat{y}_{R_1}, \hat{y}_{R_2})}.$$

Substituting the MLE expressions $\hat{\eta}_{R_1}$ and $\hat{\eta}_{R_2}$ from Lemma 4.9, and setting

$$\begin{aligned} G = T^*(\hat{y}_{R_1}, \hat{y}_{R_2}) &= \frac{\sum_{j=1,2}\sum_{i \in R_j} T(y_{R_j i})/a(\phi_{R_j i})}{\sum_{j=1,2}\sum_{i \in R_j} 1/a(\phi_{R_j i})} \\ &= \frac{1/\Phi_{R_1}}{(1/\Phi_{R_1} + 1/\Phi_{R_2})} G_{R_1} + \frac{1/\Phi_{R_2}}{(1/\Phi_{R_1} + 1/\Phi_{R_2})} G_{R_2} \\ &= b_{R_1}G_{R_1} + (1 - b_{R_1})G_{R_2}, \end{aligned}$$

the result follows by computing logarithms. □

Fact 4.1. *To test $H_0 : \eta_{R_1} = \eta_{R_2}$ versus $H_1 : \eta_{R_1} > \eta_{R_2}$, the log-likelihood ratio test statistic is given by*

$$\Delta = 1(\hat{\eta_{R_1}} > \hat{\eta_{R_2}})(\kappa(G_{R_1}, \Phi_{R_1}) + \kappa(G_{R_2}, \Phi_{R_2}) - \kappa(G, \Phi)) \tag{4.6}$$

Similar result holds for the alternative $H_1 : \eta_{R_1} < \eta_{R_2}$ with the inequalities reversed.

In the above expression for Δ (with $R_1 = R, R_2 = R \setminus P$), the key terms are the values b_R and G_R. $G_R = T^*(\hat{y}_R)$ is a function of the data (T^* is a *sufficient statistic* for the distribution), and thus is the equivalent of a measurement. In fact, G_R is a weighted average of $T(y_i)$s in R. Thus, $G_R/\Phi_R = \sum_{i \in R} T(y_i)/a(\phi_i)$ represents the *total* in R. Similarly, $G/\Phi = M$ gives the aggregate for the region and hence $m_R = (\Phi/\Phi_R)(G_R/G)$ is the fraction of total measurements contained in R. Also, $1/\Phi_R$ gives the total baseline value of R which is independent of the actual measurements and only depends on some baseline measure, i.e., $1/\Phi_R = B$. Hence, $b_R = \Phi/\Phi_R$ gives the fraction of total baseline measure in R. The next theorem provides an expression for Δ in terms of m_R and b_R.

Theorem 4.12. *Let $R_1 = R$ and $R_2 = R \setminus P$. To obtain $R \in \mathcal{A}$ that maximizes discrepancy, assume $G = M/B$ and $\Phi = B$ to be fixed and consider the parametrization of Δ in terms of b_R and $m_R = b_R G_R/G$.*

The discrepancy measure (ignoring additive constants) $d(.,.)$ is given by

$$\begin{aligned} d(m_R, b_R)\frac{\Phi}{G} &= m_R g_e(G\frac{m_R}{b_R}) - \frac{b_R}{G} B_e(g_e(G\frac{m_R}{b_R})) + \\ & (1 - m_R) g_e(G\frac{1 - m_R}{1 - b_R}) - \\ & \frac{(1 - b_R)}{G} B_e(g_e(G\frac{1 - m_R}{1 - b_R})) \end{aligned} \tag{4.7}$$

Proof. Follows by substituting $G_R = Gm_R/b_R$, $G_{R\setminus P} = G(1 - m_R)/(1 - b_R)$ in (4.5), simplifying and ignoring additive constants. □

4.B Discrepancy Measures For Specific Distributions

We can now derive discrepancy functions for specific distribution families and some properties that will be used to analyze algorithms for maximizing them.

4.B.1 Poisson Scan Statistic

The Poisson scan statistic was designed for measured data generated by an underlying Poisson distribution. We reproduce Kulldorff's derivation of the likelihood ratio test, starting from our general discrepancy function Δ.

In a Poisson distribution, underlying points are measured in the presence of some rare event (e.g. presence of some rare disease in an individual) and hence the measurement attached to each point is binary with a 1 indicating presence of the event. The number of points that get marked on a subset $R \subset P$ follows a Poisson process with base measure b and intensity λ if (i) $N(\emptyset) = 0$, (ii) $N(A) \sim \text{Poisson}(\lambda b(A)), A \subset R, b(\cdot)$ is a baseline measure defined on R and λ is a fixed intensity parameter (examples of $b(A)$ include the area of A, total number of points in A, *etc.*), and (iii) the number of marked points in disjoint subsets are independently distributed.

Derivation of the Discrepancy Function. A random variable $y \sim$ Poisson$(\lambda\xi)$ is a member of 1EXP with $T(y) = y/\xi, \phi = 1/\xi, a(\phi) = \phi, \eta = \log(\lambda), B_e(\eta) = \exp(\eta), g_e(x) = \log(x)$. For a set of n independent measurements with mean $\lambda\xi_i, i = 1, \cdots, n$, $T^*(\bar{y}) = \sum_{i=1}^n y_i / \sum_{i=1}^n \xi_i, \phi^* = (\sum_{i=1}^n \xi_i)^{-1}$. For a subset R, assume the number of marked points follows a Poisson process with base measure $b(\cdot)$ and log-intensity η_R while that in $R \setminus P$ has the same base measure but log-intensity $\eta_{R\setminus P}$. For any partition $\{A_i\}_i$ of R and $\{B_j\}_j$ of $R\setminus P$, $\{N(A_i)\}_i$ and $\{N(B_j)\}_j$ are independently distributed Poisson variables with mean $\{\exp(\eta_R)b(A_i)\}$ and $\{\exp(\eta_{R\setminus P})b(B_j)\}$ respectively. Then, $1/\Phi_R = \sum_{A_i} b(A_i)) = b(R)$, $1/\Phi_{R\setminus P} = b(R \setminus P)$,

$$G_R = \frac{\sum_{A_i} N(A_i)}{\sum_{A_i} b(A_i)} = \frac{N(R)}{b(R)}, \ \ G_{R\setminus P} = \frac{N(R \setminus P)}{b(R \setminus P)}, \ \ \text{and } G = \frac{N(R) + N(R \setminus P)}{b(R) + b(R \setminus P)}.$$

Hence,

$$b_R = \frac{b(R)}{b(R) + b(R \setminus P)} \ \ \text{and} \ \ m_R = \frac{N(R)}{N(R) + N(R \setminus P)}.$$

$$\begin{aligned}
\tilde{d}_P(m_R, b_R)\frac{\Phi}{G} &= m_R\left(\log(G) + \log\left(\frac{m_R}{b_R}\right)\right) - b_R\frac{m_R}{b_R} + \\
&\quad (1 - m_R)\left(\log(G) + \log\left(\frac{1 - m_R}{1 - b_R}\right)\right) - \frac{1 - m_R}{1 - b_R}(1 - b_R) \\
&= m_R \log\left(\frac{m_R}{b_R}\right) + (1 - m_R)\log\left(\frac{1 - m_R}{1 - b_R}\right) + c_1,
\end{aligned}$$

for a constant c_1, and hence

$$\tilde{d}_P(m_R, b_R) = M\left(m_R \log\left(\frac{m_R}{b_R}\right) + (1 - m_R)\log\left(\frac{1 - m_R}{1 - b_R}\right) + c_1\right).$$

Note that the discrepancy is independent of the partition used and hence is well defined. To normalize $\tilde{d}_P$ so that it is not dependent on the size of the data we set

$$d_P(m_R, b_R) = m_R \log\left(\frac{m_R}{b_R}\right) + (1 - m_R)\log\left(\frac{1 - m_R}{1 - b_R}\right).$$

Properties of the Poisson Scan Statistic. It is easy to see that d_P is a convex function of m_R and b_R, is always positive, and grows without bound as either of m_R and b_R tends to zero. It is zero when $m_R = b_R$. The Poisson scan statistic can also be viewed as the Kullback-Leibler distance between the two two-point distributions $[m_R, 1 - m_R]$ and $[b_R, 1 - b_R]$. We will consider maximizing the Poisson scan statistic over the region $S_n = [1/n, 1 - 1/n]^2$. To estimate the size of an ε-approximate family for d_P, we will compute λ^* over S_n.

Let

$$f_P(x, y) = x \log\frac{x}{y} + (1 - x)\log\frac{1 - x}{1 - y}.$$

$$\nabla f_P = \mathbf{i}\left(\log\frac{x}{1-x} - \log\frac{y}{1-y}\right) + \mathbf{j}\left(-\frac{x}{y} + \frac{1-x}{1-y}\right)$$

$$H(f_P) = \begin{pmatrix} \frac{\partial^2 f}{\partial x^2} & \frac{\partial^2 f}{\partial x \partial y} \\ \frac{\partial^2 f}{\partial y \partial x} & \frac{\partial^2 f}{\partial y^2} \end{pmatrix} = \begin{pmatrix} \frac{1}{x(1-x)} & \frac{-1}{y(1-y)} \\ \frac{-1}{y(1-y)} & \frac{x}{y^2} + \frac{1-x}{(1-y)^2} \end{pmatrix}$$

Note that $f_{xx}(1-1/i, y) = O(i)$ and $f_{xx}(1/i, y) = O(i)$ in S_n for all y and also that $f_{yy}(x, 1/i) = O(i^2)$ for $x > C/n$ and $f_{yy}(x, 1-\frac{1}{i}) = O(i^2)$ for x small enough. Also $f_x(1-1/i, y) = O(\log i)$ and $|f_y(x, 1/i)| = O(i)$.

The Jensen-Shannon divergence is a symmetrized variant of the Kullback-Leibler distance. We mentioned earlier that d_P can be expressed as a Kullback-Leibler distance. Replacing this by the Jensen-Shannon distance, we get a symmetric version of the Poisson scan statistic, for which all the bounds above hold.

4.B.2 Gaussian Scan Statistic

It is more natural to use an underlying Gaussian process when measurements are real numbers, instead of binary events. In this section, we derive a discrepancy function for an underlying Gaussian process. To the best of our knowledge, at the time of original publication this derivation is novel.

Derivation of the Discrepancy Function. A random variable y that follows a Gaussian distribution with mean ξ and variance $1/\tau^2$ (denoted as $y \sim N(\xi, 1/\tau^2)$ is a member of 1EXP with $T(y) = y, \eta = \xi, B_e(\eta) = \eta^2/2, \phi = 1/\tau^2, a(\phi) = \phi, g_e(x) = x$. For a set of n independent measurements with mean ξ and variances $1/\tau_i^2, i = 1, \cdots, n$(known), $\phi^* = (\sum_{i=1}^n \tau_i^2)^{-1}$ and $T^*(\bar{y}) = \sum_{i=1}^n y_i \tau_i^2 / \sum_{i=1}^n \tau_i^2$. Assume measurements in $i \in R$ are independent $N(\xi_R, 1/\tau_i^2)$ while those $i \in R \setminus P$ are independent $N(\xi_{R\setminus P}, 1/\tau_i^2)$. Then, $\Phi_R = (\sum_{i \in R} \tau_i^2)^{-1}$, $\Phi_{R\setminus P} = (\sum_{i \in R\setminus P} \tau_i^2)^{-1}$,

$$G_R = \frac{\sum_{i \in R} \tau_i^2 y_i}{\sum_{i \in R} \tau_i^2}, \quad G_{R\setminus P} = \frac{\sum_{i \in R\setminus P} \tau_i^2 y_i}{\sum_{i \in R\setminus P} \tau_i^2}, \quad \text{and} \quad G = \frac{\sum_{i \in R \cup R\setminus P} \tau_i^2 y_i}{\sum_{i \in R \cup R\setminus P} \tau_i^2}.$$

Hence,

$$b_R = \frac{1/\Phi_R}{(1/\Phi_R + 1/\Phi_{R\setminus P})} = \frac{\sum_{i \in R} \tau_i^2}{\sum_{i \in R \cup R\setminus P} \tau_i^2} \quad \text{and} \quad m_R = \frac{\sum_{i \in R} \tau_i^2 y_i}{\sum_{i \in R \cup R\setminus P} \tau_i^2}.$$

Thus,

$$\begin{aligned}
\tilde{d}_G(m_R, b_R)\frac{\Phi}{G} &= m_R G \frac{m_R}{b_R} - \frac{b_R}{G} G \frac{m_R}{b_R} + (1-m_R) G \frac{1-m_R}{1-b_R} - \frac{1-b_R}{G} G \frac{1-m_R}{1-b_R} \\
&= G\left(\frac{m_R^2}{b_R} + \frac{(1-m_R)^2}{1-b_R}\right) - 1 \\
&= G\frac{(m_R - b_R)^2}{b_R(1-b_R)}
\end{aligned}$$

and hence

$$\tilde{d}_G(m_R, b_R) = \frac{M^2}{B}\frac{(m_R - b_R)^2}{b_R(1-b_R)}.$$

An important special case occurs when the variance is constant (i.e. $\tau_i^2 = \tau^2$ for each i). Then,

$$b_R = \frac{|R|}{|R| + |R \setminus P|} \text{ and } m_R = \frac{\sum_{i \in R} y_i}{\sum_{i \in R \cup R \setminus P} y_i}$$

with the discrepancy being maximized if a small fraction of points account for a large fraction of the total or vice-versa. Note that the underlying baseline b is a weighted counting measure which aggregate weights τ_i^2 attached to points in a region.

To normalize $\tilde{d}_G$ so it is not dependent on the size of the data (note $M^2/B = O(M)$ because $M = O(B)$ in this setting) we set

$$d_G(m_R, b_R) = \frac{(m_R - b_R)^2}{b_R(1-b_R)}.$$

Properties of the Gaussian Scan Statistic. Again, it can be shown that d_G is a convex function of both parameters, and grows without bound as b_R tends to zero or one. Note that this expression can be viewed as the χ^2-distance between the two two-point distributions $[m_R, 1-m_R], [b_R, 1-b_R]$. Let

$$f_G(x,y) = \frac{(x-y)^2}{y(1-y)}.$$

$$\nabla f_G = \mathbf{i}\left(\frac{2x-2y}{y(1-y)}\right) + \mathbf{j}\left(\frac{2y-2x}{y(1-y)} - \frac{(x-y)^2(1-2y)}{y^2(1-y)^2}\right)$$

$$H(f_G) \;=\; \begin{pmatrix} \frac{2}{y(1-y)} & \frac{-2}{y(1-y)} - \frac{2(x-y)(1-2y)}{y^2(1-y)^2} \\ \frac{-2}{y(1-y)} - \frac{2(x-y)(1-2y)}{y^2(1-y)^2} & \frac{2}{y(1-y)} + \frac{4(x-y)(1-2y)+2(x-y)^2}{y^2(1-y)^2} + \frac{(x-y)^2(1-2y)^2}{y^3(1-y)^3} \end{pmatrix}$$

Note that $f_{xx}(x, 1/i) = O(i)$ and that $f_{yy}(x, 1/i) = O(i^3)$. Also not that $f_x(x, 1/i) = O(i)$ and $|f_y(x, 1/i)| = O(i^2)$.

4.B.3 Bernoulli Scan Statistic

Modeling a system with an underlying Bernoulli distribution is appropriate when the events are binary and each measured event corresponds to a particular baseline event. G is the percentage of baseline points associated with a measured point. For instance, a baseball player's batting average describes the fraction of *at bats* that the player gets a *hit*. We could say the location of the final pitch of an at bat is a baseline point and it is also a measured point if the batter gets a hit. This requires a further restriction on S_n that $b_R > G \cdot m_R$, more specifically $b_R \geq G \cdot m_R + 1/n$. When referring to S_n for the Bernoulii distribution we include this extra condition unless otherwise stated.

Derivation of the Discrepancy Function. A binary measurment y at a point has a Bernoulli distribution with parameter θ if $P(y=1)=\theta^y(1-\theta)^{1-y}$. This is a member of 1EXP with $T(y)=y, \eta=\log(\theta/(1-\theta)), B_e(\eta)=\log(1+\exp(\eta)), \phi=1, a(\phi)=1, g_e(x)=\log(x)-\log(1-x)$.

For a set of n independent measurements with parameter η, $\phi^*=1/n$, $T^*(\bar{y})=\sum_{i=1}^n y_i/n$. Assuming measurements in R and $R\setminus P$ are independent Bernoulli with parameters η_R and $\eta_{R\setminus P}$ respectively, $\Phi_R=1/|R|, \Phi_{R\setminus P}=1/|R\setminus P|$,

$$G_R=\frac{y(R)}{|R|},\ \ G_{R\setminus P}=\frac{y(R\setminus P)}{|R\setminus P|},\ \ G=\frac{y(R)+y(R\setminus P)}{|R|+|R\setminus P|},$$

$$b_R=\frac{|R|}{|R|+|R\setminus P|},\ \ \text{and}\ \ m_R=\frac{y(R)}{y(R)+y(R\setminus P)}.$$

Note that $y(A)$ denotes the number of 1's in a subset A. Thus,

$$\begin{aligned}\tilde{d}_B(m_R,b_R)\frac{\Phi}{G} &= m_R\log\left(\frac{m_R}{b_R}\right)+(1-m_R)\log\left(\frac{1-m_R}{1-b_R}\right)\\ &+\left(\frac{b_R}{G}-m_R\right)\log\left(1-G\frac{m_R}{b_R}\right)\\ &+\left(\frac{1-b_R}{G}-1+m_R\right)\log\left(1-G\frac{1-m_R}{1-b_R}\right).\end{aligned}$$

To normalize $\tilde{d}_B$ so it is not dependent on the size of the data set set

$$\begin{aligned}d_B(m_R,b_R) &= m_R\log\left(\frac{m_R}{b_R}\right)+(1-m_R)\log\left(\frac{1-m_R}{1-b_R}\right)\\ &+\left(\frac{b_R}{G}-m_R\right)\log\left(1-G\frac{m_R}{b_R}\right)\\ &+\left(\frac{1-b_R}{G}-1+m_R\right)\log\left(1-G\frac{1-m_R}{1-b_R}\right).\end{aligned}$$

Properties of the Bernoulli Scan Statistic. Much like d_P, it is easy to see that d_B is a convex function of m_R and b_R, is always positive, and grows without bound as either b_R or m_R tend to zero or one. The complexity of an ε-approximate family for d_B, the Bernoulli scan statistic, can be analyzed by letting

$$\begin{aligned}f_B(x,y) &= x\log\frac{x}{y}+(1-x)\log\frac{1-x}{1-y}+\left(\frac{y}{G}-x\right)\log\left(1-G\frac{x}{y}\right)+\\ &\left(\frac{1-y}{G}-1+x\right)\log\left(1-G\frac{1-x}{1-y}\right),\end{aligned}$$

where G is a constant.

$$\nabla f_B = \mathbf{i}\left(\log\frac{x}{y} - \log\frac{1-x}{1-y} + \log\left(1 - G\frac{1-x}{1-y}\right) - \log\left(1 - G\frac{x}{y}\right)\right) + \mathbf{j}\left(\frac{1}{G}\log\left(1 - G\frac{x}{y}\right) - \frac{1}{G}\log\left(1 - G\frac{1-x}{1-y}\right) + 1 + \frac{1}{G}\right)$$

$$H(f_B) = \begin{pmatrix} \frac{1}{x} + \frac{1}{1-x} + \frac{G}{(1-y)-G(1-x)} + \frac{G}{y-Gx} & \frac{-1}{y-Gx} - \frac{1}{(1-y)-G(1-x)} \\ \frac{-1}{y-Gx} - \frac{1}{(1-y)-G(1-x)} & \frac{x}{(y-Gx)y} + \frac{1-x}{((1-y)-G(1-x))(1-y)} \end{pmatrix}$$

Note that $f_{xx}((x, Gx + 1/n)) = O(n)$ for any x in S_n. Also, $f_{yy}((x, Gx + 1/i) = O(i)$ for x large enough in S_n. Also note that $f_x(x, Gx + 1/i) = O(\log i)$ and $|f_y(x, Gx + 1/i)| = O(\log i)$.

4.B.4 Gamma Scan Statistic

When events arrive one after another, where a Poisson variable describes the interval between events, then a gamma distribution describes the count of events after a set time.

Derivation of the Discrepancy Function. A positive measurement y has a gamma distribution with mean $\xi > 0$ and shape $\nu > 0$ if it has density

$$\frac{\nu^\nu}{\xi^\nu \Gamma(\nu)} \cdot e^{\left(-\frac{\nu}{\xi}y\right)} \cdot x^{\nu-1}$$

and is a member of 1EXP with $T(y) = y, \eta = -\frac{1}{\xi}(< 0), B_e(\eta) = -\log(-\eta), \phi = 1/\nu, a(\phi) = \phi, g_e(x) = -\frac{1}{x}$. Following arguments similar to the Gaussian case, $\Phi_R = (\sum_{i \in R} \nu_i)^{-1}, \Phi_{R \setminus P} = (\sum_{i \in R \setminus P} \nu_i)^{-1}$,

$$G_R = \frac{\sum_{i \in R} \nu_i y_i}{\sum_{i \in R} \nu_i}, \quad G_{R \setminus P} = \frac{\sum_{i \in R \setminus P} \nu_i y_i}{\sum_{i \in R \setminus P} \nu_i}, \quad \text{and} \quad G = \frac{\sum_{i \in R \cup R \setminus P} \nu_i y_i}{\sum_{i \in R \cup R \setminus P} \nu_i}.$$

Hence,

$$b_R = \frac{1/\Phi_R}{(1/\Phi_R + 1/\Phi_{R \setminus P})} = \frac{\sum_{i \in R} \nu_i}{\sum_{i \in R \cup R \setminus P} \nu_i} \quad \text{and} \quad m_R = \frac{\sum_{i \in R} \nu_i y_i}{\sum_{i \in R \cup R \setminus P} \nu_i y_i}.$$

Thus,

$$\begin{aligned}
\tilde{d}_\gamma(m_R, b_R)\frac{\Phi}{G} &= m_R\left(-\frac{b_R}{Gm_R}\right) - \frac{b_R}{G}\log\left(G\frac{m_R}{b_R}\right) + \\
&\quad (1-m_R)\left(-\frac{1-b_R}{G(1-m_R)}\right) - \frac{1-b_R}{G}\log\left(G\frac{1-m_R}{1-b_R}\right) \\
&= \frac{b_R}{G}\log\left(\frac{b_R(1-m_R)}{m_R(1-b_R)}\right) - \frac{1}{G}\log\left(\frac{1-m_R}{1-b_R}\right) + c_3 \\
&= \frac{b_R}{G}\log\left(\frac{b_R}{m_R}\right) + \frac{1-b_R}{G}\log\left(\frac{1-b_R}{1-m_R}\right) + c_3
\end{aligned}$$

for constant $c_3 > 0$ and hence ignoring additive constants,

$$\tilde{d}_\gamma(m_R, b_R) = B\left(b_R\log\left(\frac{b_R}{m_R}\right) + (1-b_R)\log\left(\frac{1-b_R}{1-m_R}\right)\right).$$

For a fixed shape parameter (i.e. $\nu_i = \nu$ for each i),

$$b_R = \frac{|R|}{|R| + |R \setminus P|} \text{ and } m_R = \frac{\sum_{i\in R} y_i}{\sum_{i\in R\cup R\setminus P} y_i}.$$

To normalize $\tilde{d}_\gamma$ so it is not dependent on the size of the data we set

$$d_\gamma(m_R, b_R) = b_R\log\left(\frac{b_R}{m_R}\right) + (1-b_R)\log\left(\frac{1-b_R}{1-m_R}\right) = d_P(b_R, m_R).$$

Properties of the Gamma Scan Statistic. Because $d_\gamma(m_R, b_R) = d_P(b_R, m_R)$ up to a relative and additive constant, $f_\gamma(x, y) = f_P(y, x)$ and properties of f_{xx} and f_{yy} of f_P apply.

4.C Combinatorial Algorithms on Terrains

4.C.1 Half spaces, intervals, and slabs

Let $h : \mathbb{R} \to \mathbb{R}$ be a piecewise-linear height function over a one-dimensional domain with a possibly negative range. Each range in $\mathcal{H}_{||}$ is defined by a single point in the domain.

Lemma 4.10. *For continuous $h : \mathbb{R} \to \mathbb{R}$ the range*

$$\arg\max_{R\in\mathcal{H}_{||}} \int_R h(p)\, dp$$

is defined by an endpoint r such that $h(r) = 0$.

Proof. If the end point r moved so the size of R is increased and $h(r) > 0$ then the integral would increase, so $h(r)$ must be non positive. If the end point r is moved so the size of R is decreased and $h(r) < 0$ then integral would also increase, so $h(r)$ must be non negative. $\square$

This proof extends trivially to axis-parallel slabs $\mathcal{S}_{||}$ (which can be thought of as intervals) as well.

Lemma 4.11. *For continuous $h : \mathbb{R} \to \mathbb{R}$, the range*

$$\arg\max_{R \in \mathcal{S}_{||}} \int_R h(p)\, dp$$

is defined by two endpoints r_l and r_r such that $h(r_l) = 0$ and $h(r_r) = 0$.

Let h have n vertices. For both $\mathcal{H}_{||}$ and $\mathcal{S}_{||}$, the optimal range can be found in $O(n)$ time. For $\mathcal{H}_{||}$, simply sweep the space from left to right keeping track of the integral of the height function. When the height function has a point r such that $h(r) = 0$, compare the integral versus the maximum so far.

For $\mathcal{S}_{||}$, we reduce this to a one-dimensional point set problem. First sweep the space and calculate the integral in between every consecutive pair of points r_1 and r_2 such that $h(r_1) = 0 = h(r_2)$ and there is no point r_3 such that $h(r_3) = 0$ and $r_1 < r_3 < r_2$. Treat each of these intervals as a point with weight set according to its value. Now run the one-dimensional algorithm from Section 4.4 for linear discrepancy of red and blue points where the positive intervals have a red weight equal to the integral and the negative intervals have a blue weight equal to the negative of the integral.

Theorem 4.13. *For continuous, piecewise-linear $h : \mathbb{R} \to \mathbb{R}$ with n vertices*

$$\arg\max_{R \in \mathcal{H}_{||}} \int_R h(p)\, dp \quad \textit{and} \quad \arg\max_{R \in \mathcal{S}_{||}} \int_R h(p)\, dp$$

can be calculated in $O(n)$ time.

This result extends trivially to a higher dimensional domains as long as the families of ranges are no more complicated.

Theorem 4.14. *For continuous, peicewise-linear $h : \mathbb{R}^d \to \mathbb{R}$ with n vertices,*

$$\arg\max_{R \in \mathcal{H}_{||}} \int_R h(p)\, dp \quad \textit{and} \quad \arg\max_{R \in \mathcal{S}_{||}} \int_R h(p)\, dp$$

can be calculated in $O(n)$ time.

Proof. The sweep step of the algorithms described above are performed in the same way, only now the integral of up to $O(n)$ cubic functions must be calculated. However, this integral can be stored implicitly as a single linear function and can be updated in constant time every time a new vertex is reached. $\square$

Finally, we now present a slightly surprising theorem about the difference between two terrains.

Theorem 4.15. *Let $M : \mathbb{R}^d \to \mathbb{R}$ and $B : \mathbb{R}^d \to \mathbb{R}$ be piecewise-linear functions with n and m vertices respectively.*

$$\arg\max_{R \in \mathcal{H}_{||}} \int_R M(p) - B(p)\ dp \quad \text{and} \quad \arg\max_{R \in \mathcal{S}_{||}} \int_R M(p) - B(p)\ dp$$

can be calculated in $O(n + m)$ time.

Naively, this could be calculated in $O(nm)$ time by counting the vertices on the terrain $h(p) = M(p) - B(p)$. But we can do better.

Proof. Although there are more than $n + m$ vertices in $h(p) = M(p) - B(p)$, the equations describing the height functions only change when a vertex of one of the original functions is encountered. Thus there are only $O(n + m)$ linear functions which might cross 0. These can be calculated by projecting M and B independently to the axis of $\mathcal{H}_{||}$ or $\mathcal{S}_{||}$ and then taking their sum between each consecutive vertex of either function. □

4.C.2 Rectangles

Although Theorem 4.3 extends the 1-dimensional case for point sets to a $O(n^2 \log n)$ algorithm for rectangles, when the data is given as picewise-linear terrains the direct extension does not go through. However, a simple $O(n^4)$ time algorithm, under a certain model, does work. Following algorithm Exact, we make four nested sweeps over the data. The first two bound the x coordinates and the second two bound the y coordinates. The inner most sweep keeps a running total of the integral in the range. However, unlike Exact each sweep does not give an exact bound for each coordinate, rather it just restricts its position between two vertices. The optimal position is dependent on all four positions, and needs to be determined by solving a system of four quadratic equations. This system seems to, in general, have no closed form solution (see the next subsection) and needs to be done via a numerical solver. However, these equations can be updated in constant time in between states of each sweep, so under the model that the numerical solver takes $O(1)$ time, this algorithm runs in $O(n^4)$ time.

For the full correctness of the algorithm, there is actually one more step required. Given that each side of the rectangle is bounded between two vertices, the set of four equations is dependent on which face of the terrain that the corner of the rectangle lies in. It turns out that each possible corner placement can be handled individually without affecting the asymptotics. The n vertices impose an $n \times n$ grid on the domain, yielding $O(n^2)$ grid cells in which a corner may lie. Because the terrain is a planar map, there are $O(n)$ edges as well, and each edge can cross at most $O(n)$ grid cells. Since no two edges can cross, this generates at most $O(n^2)$ new regions inside of all $O(n^2)$ grid cells. Since each rectangle is determined by the placement of two opposite corners the total complexity is not affected and is still $O(n^4)$. We summarize in the following lemma.

Lemma 4.12. *Consider a model where a system of* 4 *quadratic equations can be solved in* $O(1)$ *time. Then let* $h : \mathbb{R}^2 \to \mathbb{R}$ *be a piecewise-linear function with* n *vertices.*

$$\arg\max_{R \in \mathcal{R}_2} \int_R h(p)\, dp$$

can be solved in $O(n^4)$ *time.*

4.C.3 Algebra for Finding Optimal Rectangle on a P-L Terrain

For a piecewise-linear terrain $h : \mathbb{R}^2 \to \mathbb{R}$ we wish to evaluate $\bar{D}(h, R) = \int_R h(p)\, dp$ where R is some rectangle over the ground set of h. Within R, the value of h is described by a set of triangles $T_R = \{t_1, \ldots, t_k\}$. Let R be described by its four boundaries. Let x_1 and x_2 describe the left and right boundaries, respectively, and let y_1 and y_2 describe the top and bottom boundaries, respectively. Now

$$\bar{D}(h, R) = \int_{x_1}^{x_2} \int_{y_1}^{y_2} h(x, y)\, dydx.$$

Assume that we have computed the integral from x_1 up to $x(v)$ the x-coordinate of a vertex v in the piecewise-linear terrain. To extend the integral up to the $x(u)$ where u is the next vertex to the right. We need to consider all of the triangles that exist between $x(v)$ and $x(u)$. Label this set $\{t_1, \ldots, t_k\}$ where t_i is below t_j in the y-coordinate sense for $i < j$. Note that no triangle can begin or end in this range and this order must be preserved. We also consider the intersection between the edge of the triangulation and the rectangle with describes R a vertex. Let the slope within a triangle t_i be described

$$h_i(x, y) = \alpha_i x + \beta_i y + \gamma_i \tag{4.8}$$

and describe the edge of the triangulation that divides t_i and t_{i+1} as

$$l_i = \omega_i x + \kappa_i. \tag{4.9}$$

Now we want to solve the integral from $x(v)$ to $x(u)$

$$\int_{x(v)}^{x(u)} \int_{y_1}^{y_2} h(x, y)\, dydx.$$

But this is difficult, as hinted at by the following lemma.

Lemma 4.13. *The function*

$$\bar{D}(h, [x_1, x_2] \times [y_1, y_2]) = \int_{x_1}^{x_2} \int_{y_1}^{y_2} h(x, y)\, dydx$$

is a third order polynomial in x_2*, the* x*-position of the right endpoint.*

Proof. The integral from $x(v)$ to $x(u)$ is described

$$\int_{x(v)}^{x(u)}\int_{y_1}^{y_2} h(x,y)dydx$$

$$= \int_{x(v)}^{x(u)}\left[\int_{y_1}^{l_1(x)} h_1(x,y)dy + \sum_{i=2}^{k-1}\int_{l_{i-1}(x)}^{l_i(x)} h_i(x,y)dy + \int_{l_{k-1}(x)}^{y_2} h_k(x,y)dy\right]dx$$

$$= \int_{x(v)}^{x(u)}\left[\begin{array}{l} \int_{y_1}^{\omega_1 x+\kappa_1}(\alpha_1 x+\beta_1 y+\gamma_1)dy+ \\ \sum_{i=2}^{k-1}\int_{\omega_{i-1}x+\kappa_{i-1}}^{\omega_i x+\kappa_i}(\alpha_i x+\beta_i y+\gamma_i)dy+ \\ \int_{\omega_{k-1}x+\kappa_{k-1}}^{y_2}(\alpha_k x+\beta_k y+\gamma_k)dy \end{array}\right]dx$$

$$= \int_{x(v)}^{x(u)}\left[\begin{array}{l} \alpha_1 xy+\frac{1}{2}\beta_1 y^2+\gamma_1 y\,|_{y=y_1}^{\omega_1 x+\kappa_1} + \\ \sum_{i=2}^{k-1}\alpha_i xy+\frac{1}{2}\beta_1 y^2+\gamma_i y\,|_{y=\omega_{i-1}x+\kappa_{i-1}}^{\omega_1 x+\kappa_1} + \\ \alpha_{k-1}xy+\frac{1}{2}\beta_{k-1}y^2+\gamma_{k-1}y\,|_{y=\omega_{k-1}x+\kappa_{k-1}}^{y_2} \end{array}\right]dx$$

$$= \int_{x(v)}^{x(u)}\left[\begin{array}{l} (\alpha_1 xy_1+\frac{1}{2}\beta_1(y_1)^2+\gamma_1 y_1)-(\alpha_1 x(\omega_1 x+\kappa_1)- \\ \frac{1}{2}\beta_1(\omega_1 x+\kappa_1)^2-\gamma_1(\omega_1 x+\kappa_1))+ \\ \sum_{i=2}^{k-1}\left(\begin{array}{l} \alpha_i x(\omega_{i-1}x+\kappa_{i-1})+\frac{1}{2}\beta_i(\omega_{i-1}x+\kappa_{i-1})^2+ \\ \gamma_i(\omega_{i-1}x+\kappa_{i-1})-\alpha_i x(\omega_i x+\kappa_i)- \\ \frac{1}{2}\beta_i(\omega_i x+\kappa_i)^2-\gamma_i(\omega_i x+\kappa_i) \end{array}\right)+ \\ (\alpha_k x(\omega_{k-1}x+\kappa_{k-1})+\frac{1}{2}\beta_k(\omega_{k-1}x+\kappa_{k-1})^2+ \\ \gamma_k(\omega_{k-1}x+\kappa_{k-1}))-(\alpha_k xy_2+\frac{1}{2}\beta_k(y_2)^2\gamma_k y_2) \end{array}\right]dx$$

$$= \int_{x(v)}^{x(u)}\left[\begin{array}{l} (\alpha_1 xy_1+\frac{1}{2}\beta_1 y_1^2+\gamma_1 y_1)+ \\ \frac{1}{2}\sum_{i=1}^{k-1}\left(\begin{array}{l} (\alpha_{i+1}-\alpha_i)x+(\beta_{i+1}-\beta_i)(\omega_i^2 x^2+2\kappa_i\omega_i x+\kappa_i^2)+ \\ (\gamma_{i+1}-\gamma_i) \end{array}\right)- \\ (\alpha_k xy_2+\frac{1}{2}\beta_k y_2^2+\gamma_k y_2) \end{array}\right]dx$$

$$= \left[\begin{array}{l} \frac{1}{2}\alpha_1 y_1 x^2+(\frac{1}{2}\beta_1 y_1^2+\gamma_1 y_1)x+ \\ \frac{1}{2}\sum_{i=1}^{k-1}\left(\begin{array}{l} \frac{1}{2}(\alpha_{i-1}-\alpha_i+2\kappa_i\omega_i(\beta_{i+1}-\beta_i))x^2+ \\ \frac{1}{3}(\omega_i^2(\beta_{i+1}-\beta_i))x^3+ \\ (\gamma_{i+1}-\gamma_i+\kappa_i(\beta_{i+1}-\beta_i))x \end{array}\right)- \\ \frac{1}{2}\alpha_k y_2 x^2-(\frac{1}{2}\beta_k y_2^2+\gamma_k y_2)x \end{array}\right]_{x(v)}^{x(u)}$$

$$= \begin{array}{l} \frac{1}{2}(\alpha_1 y_1-\alpha_k y_2)(x(u)^2-x(v)^2)+ \\ (\frac{1}{2}\beta_1 y_1^2-\frac{1}{2}\beta_k y_2^2+\gamma_1 y_1-\gamma_k y_2)(x(u)-x(v))+ \\ \frac{1}{2}\sum_{i=1}^{k-1} \begin{array}{l} \frac{1}{2}(\alpha_{i-1}-\alpha_i+2\kappa_i\omega_i(\beta_{i+1}-\beta_i))(x(u)^2-x(v)^2)+ \\ \frac{1}{3}(\omega_i^2(\beta_{i+1}-\beta_i))(x(u)^3-x(v)^3)+ \\ (\gamma_{i+1}-\gamma_i+\kappa_i(\beta_{i+1}-\beta_i))(x(u)-x(v)) \end{array} \end{array}$$

□

Thus to solve

$$\arg\min_{x_2}\bar{D}(h,[x_1,x_2]\times[y_1,y_2])$$

requires finding where

$$\frac{\partial}{\partial x_2}\bar{D}(h,[x_1,x_2]\times[y_1,y_2])=0.$$

If it is outside the range $[x(v),x(u)]$, the two vertices bounding x_2, then it is minimized at one of the two end points. It should be noted that by symmetry, the minimum of x_1 and x_2 are independent, given y_1 and y_2, but that both are dependent on y_1 and y_2. Also, by symmetry, the previous statements can swap every x_1 and x_2 with y_1 and y_2, respectively.

Lemma 4.14. *Solving for*

$$\arg\min_{[x_1,x_2]\times[y_1,y_2]}\bar{D}(h,[x_1,x_2]\times[y_1,y_2])$$

requires solving 4 *quadratic equations of* 4 *variables, which in general has no closed-form solution.*

Proof. This system is of the form

$$\begin{bmatrix}0\\0\\0\\0\end{bmatrix}=\begin{bmatrix}0&0&\alpha_{y_1}&\alpha_{y_2}\\0&0&\beta_{y_1}&\beta_{y_2}\\\alpha_{x_1}&\alpha_{x_2}&0&0\\\beta_{x_1}&\beta_{x_2}&0&0\end{bmatrix}\begin{bmatrix}x_1\\x_2\\y_1\\y_2\end{bmatrix}+\begin{bmatrix}0&0&\omega_{y_1}&\omega_{y_2}\\0&0&\kappa_{y_1}&\kappa_{y_2}\\\omega_{x_1}&\omega_{x_2}&0&0\\\kappa_{x_1}&\kappa_{x_2}&0&0\end{bmatrix}\begin{bmatrix}x_1^2\\x_2^2\\y_1^2\\y_2^2\end{bmatrix}+\begin{bmatrix}\gamma_{x_1}\\\gamma_{x_2}\\\gamma_{y_1}\\\gamma_{y_2}\end{bmatrix}$$

which in general has no closed-form solution. □

Conclusion

As larger and larger data sets are being collected, the advancement of science not only becomes dependent on understanding these data sets, but also the process by which they are collected. The enormity of the data size precludes brute force approaches of analyzing it. Yet, the more we understand the collection process, the better we can create more data with less error, thus getting more accurate information. Both of these components of the modern scientific process indicates the need for efficient algorithms for processing this data. Or, to put it another way, algorithms are becoming the language to describe scientific hypothesis.

Statistics were created as a way of drawing conclusions about scientific hypothesis. That is, given a set of data collected in regards to a hypothesis, the statistical analysis of the data provides estimates of the validity of the hypothesis. Hence, statistics have formed the language used to understand science.

Combining these conclusions leads to the statement, *understanding algorithms for computing statistics is necessary for interpreting scientific hypothesis.* This thesis takes several important steps in this direction.

Coresets have shown to be a quite useful tool for understanding large data sets, particularly because they have guarantees in the error caused by approximation. Summarizing data sets to a manageable size is bound to introduce some error; hence methods which provide precise understanding of where this error occurs are inherently superior to those which do not. In the same vein, statistics is the process of reducing a data set to a small number of useful values that are hopefully invariant over the uncertainty in the data. Much of the field of statistics studies this relationship between data uncertainty and robustness of the statistics measured. But, when data sets are too large to directly compute statistics on the original data, the process is to first summarize the data (such as using a coreset) and then compute the statistic on the summary. In this process, it is important to analyze the final error of the statistics with respect to the original data, not just the error of the coreset with respect to the original data or the error of the statistic with respect to the coreset.

This thesis addresses this particular task for several problems. In particular, when the data set is not precise, but each point is modeled by a probability distribution, Chapter 3 creates coresets, both randomized and deterministic, for many shape fitting problems. Also, for the problem of computing spatial scan statistics we address in Chapter 4.5 the algorithm of computing an ε-sample of the data before running another approximation algorithm for a statistic. We distinguish between specific problem formulations where this is helpful and where it is not. The fact that it is not effective for certain problem formulations may indicate that these formulations are not robust, since the original data can be interpreted of an ε-sample of reality.

More work should be done on this problem in particular to investigate the power of our algorithms versus those proposed by others such as Kulldorff [72] or Neill and Moore [90], as opposed to just their efficiency and values returned on different problems. We should create spatial anomalies and determine which algorithms detect them with more accuracy, and possibly with respect to the efficiency. We suspect that there is a strong dependency on the shape of the anomaly and the family of shapes inducing the range space. Clearly, if they are from the same family, then we should be able to find a planted cluster, but if they are not, then the question becomes how closely can we approximate a shape from a different family of ranges.

There are numerous other challenges in the form of, given a large data set and a characteristic we want to measure, can we formulate a statistic that represents this characteristic and can be calculated efficiently, or an approximation of the statistic where the approximation does not distort the understanding of the characteristic. In particular, can this be done using coresets as an intermediary in some of these problems? For instance, can we capture the stability of persistence diagrams when the original data is uncertain by taking ε-samples of the available data? In robotic localization and learning using a process like particle filtering, what approximation guarantees can be make on the position of the robot or on its observations? When we model GIS data as having uncertainty described by probability distributions, what conclusions can be draw about the stability of predictions about the watershed hierarchies, or other complex geospatial properties?

Another challenge is in monitoring data from massive data sets. When we only care about extent measures, we can maintain stable ε-kernels as described in Chapter 2. But does this approach work in combination with the technique for handling outliers [6] in ε-kernels? Does it work for uncertain data where the uncertainty is described by a probability distribution? What other coresets can be maintained stably?

In summary, this thesis provides a contribution to the problem of creating algorithms for statistical problems with approximation guarantees. There is still plenty of work left to be done in this area in order to be able to create the most useful data given our resources and to be able to use that data at its potential. This is becoming a problem central to the dialogue of science, and the techniques presented herein will hopefully be used as building blocks for future progress in this direction.

Bibliography

[1] Charu C. Agarwal and Philip S. Yu, editors. *Privacy Preserving Data Mining: Models and Algorithms.* Springer, 2008.

[2] Deepak Agarwal, Andrew McGregor, Jeff M. Phillips, Suresh Venkatasubramanian, and Zhengyuan Zhu. Spatial scan statistics: Approximations and performance study. In *12 Annual ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 24–33, 2006.

[3] Deepak Agarwal, Jeff M. Phillips, and Suresh Venkatasubramanian. The hunting of the bump: On maximizing statistical discrepancy. In *SIAM-ACM Symposium on Discrete Algorithms*, pages 1137 – 1146, January 2006.

[4] Pankaj K. Agarwal, Sariel Har-Peled, and Kasturi Varadarajan. Approximating extent measure of points. *Journal of ACM*, 51(4):606–635, 2004.

[5] Pankaj K. Agarwal, Sariel Har-Peled, and Kasturi Varadarajan. Geometric approximations via coresets. *Current Trends in Combinatorial and Computational Geometry (E. Welzl, ed.)*, 2007.

[6] Pankaj K. Agarwal, Sariel Har-Peled, and Hai Yu. Robust shape fitting vai peeling and grating coresets. *Discrete & Computational Geometry*, 39:38–58, 2008.

[7] Pankaj K. Agarwal and Jeff M. Phillips. On bipartite matching under the rms distance. In *Proceedings of the 18th Canadian Conference on Computational Geometry (CCCG'06)*, pages 143–146, 2006.

[8] Pankaj K. Agarwal and Jeff M. Phillips. An efficient algorithm for euclidean 2-center with outliers. In *Proceedings of the 16th Annual European Symposium on Algorithms*, Lecture Notes in Computer Science. Springer Berlin / Heidelberg, 2008.

[9] Pankaj K. Agarwal, Jeff M. Phillips, and Hai Yu. Stability of ε-kernels. Submitted to SoCG '09, 2008.

[10] Pankaj K. Agarwal, Cecilia M. Procopiuc, and Kasturi R. Varadarajan. Approximation algorithms for k-line center. In *Proceedings 10th Annual European Symposium on Algorithms*, pages 54–63, 2002.

[11] Pankaj K. Agarwal, Bardia Sadri, and Jeff M. Phillips. Lipschitz unimodal and isotonic regression on paths and trees. Submitted to SoCG '09, 2008.

[12] Rakesh Agarwal and Ramakrishnan Srikant. Privacy-preserving data mining. *ACM SIGMOD Record*, 29:439–450, 2000.

[13] Ralph Alexander. Principles of a new method in the study of irregularities of distribution. *Inventiones Mathematicae*, 103:279–296, 1991.

[14] Noga Alon, Yossi Matias, and Mario Szegedy. The space complexity of approximating the frequency moments. *Journal of Computational Systems Science*, 58(1):137–147, 1999.

[15] A. Bagchi, A. Chaudhary, David Eppstein, and Michael Goodrich. Deterministic sampling and range counting on geometric data streams. In *Proceedings 20th Annual ACM Symposium on Computational Geometry*, pages 144–151, 2004.

[16] Deepak Bandyopadhyay and Jack Snoeyink. Almost-Delaunay simplices: Nearest neighbor relations for imprecise points. In *ACM-SIAM Symp on Discrete Algorithms*, pages 403–412, 2004.

[17] Ziv Bar-Yossef, T. S. Jayram, Ravi Kumar, and D. Sivakumar. An information statistics approach to data stream and communication complexity. *Journal of Computational Systems Science*, 68(4):702–732, 2004.

[18] Gil Barequet and Sariel Har-Peled. Efficiently approximating the mnimum-volume bounding box of a point set in three dimensions. *Journal of Algorithms*, 38:91–109, 2001.

[19] József Beck. Balanced two-coloring of finite sets in the square I. *Combinatorica*, 1:327–335, 1981.

[20] József Beck. Roth's estimate on the discrepancy of integer sequences is nearly sharp. *Combinatorica*, 1:319–325, 1981.

[21] József Beck. Irregularities of distribution I. *Acta Mathematics*, 159:1–49, 1987.

[22] József Beck. Probabilistic diophantine approximation, I Kronecker sequences. *Annals of Mathematics*, 140:451–502, 1994.

[23] József Beck and William Chen. *Irregularities of Distribution.* Cambridge University Press, 1987.

[24] József Beck and Tibor Fiala. ”Integer-Making” theorems. *Discrete Applied Mathematics*, 3:1–8, 1981.

[25] Mihai Bădoiu and Kenneth L. Clarkson. Smaller coresets for balls. In *Proceedings 14th ACM-SIAM Symposium on Discrete Algorithms*, pages 801–802, 2003.

[26] Sergio Cabello and Marc van Kreveld. Approximation algorithms for aligning points. *Algorithmica*, 37:211–232, 2003.

[27] Amit Chakrabarti, Subhash Khot, and Xiaodong Sun. Near-optimal lower bounds on the multi-party communication complexity of set disjointness. In *Proceedings IEEE Conference on Computational Complexity*, pages 107–117, 2003.

[28] Timothy Chan. Faster core-set constructions and data-stream algorithms in fixed dimensions. *Computational Geometry: Theory and Applications*, 35:20–35, 2006.

[29] Timothy Chan. Dynamic coresets. In *Proceedings of the 24th ACM Symposium on Computational Geometry*, pages 1–9, 2008.

[30] Badrish Chandramouli, Jeff M. Phillips, and Jun Yang. Value-based notification conditions in large-scale publish/subscribe systems. In *33rd International Conference on Very Large Data Bases (VLDB ’07)*, 2007.

[31] Bernard Chazelle. *The Discrepancy Method*. Cambridge University Press, 2000.

[32] Bernard Chazelle and Jiri Matousek. On linear-time deterministic algorithms for optimization problems in fixed dimensions. *Journal of Algorithms*, 21:579–597, 1996.

[33] Reynold Cheng, Dmitri V. Kalashnikov, and Sunil Prabhakar. Evaluating probabilitic queries over imprecise data. In *Proceedings 2003 ACM SIGMOD International Conference on Management of Data*, 2003.

[34] Kenneth L. Clarkson, David Eppstein, Gary L. Miller, Carl Sturtivant, and Shang-Hua Teng. Approximating center points with iterative Radon points. *International Journal of Computational Geometry and Applications*, 6:357–377, 1996.

[35] Graham Cormode, Flip Korn, S. Muthukrishnan, and Divesh Srivastava. Diamond in the rough: Finding hierarchical heavy hitters in multi-dimensional data. In *Proceedings ACM Special Interest Group on Management of Data*, pages 155–166, 2004.

[36] Graham Cormode and S. Muthukrishnan. What's hot and what's not: Tracking most frequent items dynamically. In *Proceedings ACM Symposium on Principles of Database Systems*, pages 296–306, 2003.

[37] Noel Cressie. *Statistics for Spatial Data.* John Wiley, 2nd edition, 1993.

[38] Amol Deshpande, Carlos Guestrin, Samuel R. Madden, Joseph M. Hellerstein, and Wei Hong. Model-driven data acquisition in sensor networks. In *Proceedings 13th International Conference on Very Large Data Bases*, 2004.

[39] David Dobkin and David Eppstein. Computing the discrepancy. In *Proceedings 9th Annual Symposium on Computational Geometry*, 1993.

[40] David P. Dobkin, Dimitrios Gunopulos, and Wolfgang Maass. Computing the maximum bichromatic discrepancy, with applications to computer graphics and machine learning. *NeuroCOLT Technical Report Series*, NC-TR-95-008, March 1995.

[41] M. Dwass. Modified randomization tests for nonparametric hypotheses. *Annals of Mathematical Statistics*, 28:181–187, 1957.

[42] Austin Eliazar and Ronald Parr. Dp-slam 2.0. In *Proceedings 2004 IEEE International Conference on Robotics and Automation*, 2004.

[43] J. Feigenbaum, S. Kannan, M. Strauss, and M. Viswanathan. An approximate L^1 difference algorithm for massive data streams. In *Proceedings IEEE Symposium on Foundations of Computer Science*, pages 501–511, 1999.

[44] Jerome H. Friedman and Nicholas I. Fisher. Bump hunting in high-dimensional data. *Statistics and Computing*, 9(2):123–143, April 1999.

[45] Sorabh Gandhi, Subhash Suri, and Emo Welzl. Catching elephants with mice: Sparse sampling for monitoring sensor networks. In *Proceedings 5th Embedded Networked Sensor Systems*, pages 261–274, 2007.

[46] Bernd Gärtner. Fast and robust smallest enclosing balls. In *Proceedings 7th Annual European Symposium on Algorithms*, volume LNCS 1643, pages 325–338, 1999.

[47] Phillip Good. *Permutation Tests - A Practical Guide to Resampling Methods for Testing Hypotheses.* Springer-Verlag, New York, 2nd edition, 2000.

[48] Joachim Gudmundsson and Allan Jorgensen. personal communcation. note.

[49] Leonidas J. Guibas, D. Salesin, and J. Stolfi. Epsilon geometry: building robust algorithms from imprecise computations. In *Proc. 5th Annu. ACM Sympos. Comput. Geom.*, pages 208–217, 1989.

[50] Leonidas J. Guibas, D. Salesin, and J. Stolfi. Constructing strongly convex approximate hulls with inaccurate primitives. *Algorithmica*, 9:534–560, 1993.

[51] R. H. Güting and M. Schneider. *Moving Object Databases.* Morgan Kaufmann, San Francisco, 2005.

[52] J. H. Halton. On the efficiency of certain quasi-random sequences of points in evaluating multidimensional integrals. *Numerical Mathematics*, 2:84–90, 1960.

[53] J. M. Hammersly. Monte Carlo methods for solving multivariable problems. *Annals of New York Acadamy of Science*, 86:844–874, 1960.

[54] Sariel Har-Peled. No coreset, no cry. In *Proceedings 24th Conference on Foundations of Software Technology and Theoretical Computer Science*, 2004.

[55] Sariel Har-Peled. *Approximation Algorithm in Geometry.* http://valis.cs.uiuc.edu/s̃ariel/teach/notes/aprx/, 2008.

[56] Sariel Har-Peled. Chapter 5: On complexity, sampling, and ε-nets and ε-samples. http://valis.cs.uiuc.edu/s̃ariel/teach/notes/aprx/lec/05_vc_dim.pdf, January 2008.

[57] David Haussler. Decision theoretic generalizations of PAC model of neurel net and other learning applications. *Information and Computation*, 100(78-150), 1992.

[58] David Haussler and Emo Welzl. epsilon-nets and simplex range queries. *Discrete and Computational Geometry*, 2:127–151, 1987.

[59] Martin Held and Joseph S. B. Mitchell. Triangulating input-constrained planar point sets. *Information Processing Letters*, page to appear, 2008.

[60] M. R. Henzinger, P. Raghavan, and S. Rajagopalan. Computing on data streams. Technical Report TR 1998-001, DEC Sys. Res. Center, 1998.

[61] John Hershberger and Subhash Suri. Adaptive sampling for geometric problems over data streams. *Computational Geometry: Theory and Applications*, 39:191–208, 2008.

[62] Christoph M. Hoffmann. The problems of accuracy and robustness in geometric computation. *IEEE Computer*, 22, 1989.

[63] J. Hoh and J. Ott. Scan statistics to scan markers for susceptibility genes. *Proceedings National Academy of Science USA*, 97(17):9615–9617, 2000.

[64] VS Iyengar. On detecting space-time clusters. In *Proceedings 10th Annual ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 587–592, 2004.

[65] J.Glaz and N. Balakrishnan. *Scan Statistics and Applications.* Birkhauser, Boston, 1999.

[66] J.Glaz, J.Naus, and S.Wallenstein. *Scan Statistics.* Springer-Verlag,New York, 2001.

[67] Leo Joskowicz, Elisah Sacks, and Vijay Srinivasan. Kinematic tolerance analysis. *Computer-Aided Design*, 29(2), 1997.

[68] Don E. Knuth. *Seminumerlcal Algorithms, The Art of Computer Programming*, volume 2. Addison-Wesley, 1973.

[69] Heinrich Kruger. Basic measures for imprecise point sets in $\mathbb{R}^d$. Master's thesis, Utrecht University, 2008.

[70] Martin Kulldorff. A spatial scan statistic. *Communications in Statistics: Theory and Methods*, 26:1481–1496, 1997.

[71] Martin Kulldorff. Prospective time-periodic geographical disease surveillance using a scan statistic. *Journal of the Royal Statistical Society, Series A*, 164:61–72, 2001.

[72] Martin Kulldorff. *SatScan User Guide.* http://www.satscan.org/, 7.0 edition, 2006.

[73] Eyal Kushilevitz and Noam Nisan. *Communication Complexity.* Cambridge University Press, 1997.

[74] Yi Li, Philip M. Long, and Aravind Srinivasan. Improved bounds on the samples complexity of learning. *Journal of Computer ans System Science*, 62:516–527, 2001.

[75] Zhenyu Li and Victor J. Milenkovic. Constructing strongly convex hulls using exact or rounded arithmetic. *Algorithmica*, 8:345–364, 1992.

[76] T. M. Lillesand, R. W. Kiefer, and J. W. Chipman. *Remote Sensing and Image Interpretaion.* John Wiley & Sons, 2004.

[77] Doron Lipson, Yonatan Aumann, Amir Ben-Dor, Nathan Linial, and Zohar Yakhini. Efficient calculation of interval scores for dna copy number data analysis. *9th Annual International Conference on Research in Computational Molecular Biology*, pages 83–100, 2005.

[78] Maarten Löffler and Jack Snoeyink. Delaunay triangulations of imprecise points in linear time after preprocessing. In *Proc. 24th Sympoium on Computational Geometry*, pages 298–304, 2008.

[79] Martin Löffler and Jeff M. Phillips. Shape fitting on point sets with probability distributions. Submitted to SoCG '09, 2008.

[80] Jiri Matoušek. Approximations and optimal geometric divide-and-conquer. In *Proceedings 23rd Symposium on Theory of Computing*, pages 505–511, 1991.

[81] Jiri Matoušek. Tight upper bounds for the discrepancy of halfspaces. *Discrete and Computational Geometry*, 13:593–601, 1995.

[82] Jiri Matoušek. *Geometric Discrepancy.* Springer, 1999.

[83] Jiri Matoušek. On the discrepancy for boxes and polytopes. *Monatsh. Math.*, 127:325–336, 1999.

[84] Jiri Matoušek, Emo Welzl, and Lorenz Wernisch. Discrepancy and approximations for bounded VC-dimension. *Combinatorica*, 13:455–466, 1993.

[85] P. McMullen. The maximum number of faces of a convex polytope. *Mathematika*, 17(179-184), 1971.

[86] Victor J. Milenkovic. Verifiable implementatios of geometric algorithms using finte precision arithmetic. *Artificial Intelligence*, 37:377–401, 1988.

[87] T. Nagai and N. Tokura. Tight error bounds of geometric problems on convex objects with imprecise coordinates. In *Jap. Conf. on Discrete and Comput. Geom.*, LNCS 2098, pages 252–263, 2000.

[88] Daniel Neill and Andrew Moore. Anomalous spatial cluster detection. In *Proceedings KDD 2005 Workshop on Data Mining Methods for Anomaly Detection*, 2005.

[89] Daniel B. Neill and Andrew W. Moore. A fast multi-resolution method for detection of significant spatial disease clusters. In *Proceedings Advances in Neural Information Processing Systems*, volume 10, pages 651–658, 2004.

[90] Daniel B. Neill and Andrew W. Moore. Rapid detection of significant spatial clusters. In *Proceedings 10th Annual ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2004.

[91] Harald Niederreiter. *Random Number Generation and Quasi-Monte Carlo Methods.* SIAM, 1992.

[92] Y. Ostrovsky-Berman and L. Joskowicz. Uncertainty envelopes. In *Abstracts 21st European Workshop on Comput. Geom.*, pages 175–178, 2005.

[93] Mark H. Overmars. The design of dynamic data structures. *Lecture Notes in Computer Science*, 156, 1983.

[94] G. P. Patil and C. Taillie. Upper level set scan statistic for detecting arbitrary shaped hotspots. *Environmentla and Ecological Statistics*, 11:183–197, 2004.

[95] Jeff Phillips, Andrew Ladd, and Lydia E. Kavraki. Simulated knot tying. In *IEEE International Conference on Robotics and Automation*, volume 1, pages 841–846, 2002.

[96] Jeff M. Phillips. Algorithms for ε-approximations of terrains. In *In Proceedings of 35th International Colloquim on Automata, Languages, and Programming*, volume 5125/2008, pages 447–458, 2008. arXiv:0801.2793.

[97] Jeff M. Phillips. Near-linear time, deterministic ε-quantizations in one dimension. 2008.

[98] Jeff M. Phillips, Nazareth Bedrossian, and Lydia E. Kavraki. Guided expansive trees: A search strategy for control and cost constrained systems. In *IEEE International Conference on Robotics and Automation*, volume 4, pages 3968–3973, May 2004.

[99] Jeff M. Phillips, Lydia E. Kavraki, and Nazareth Bedrossian. Probabilistic optimization applied to spacecraft rendezvous and docking. In *AAS/AIAA Space Flight Mechanics Meeting*, volume AAS 03-116, February 2003.

[100] Jeff M. Phillips, Lydia E. Kavraki, and Nazareth Bedrossian. Spacecraft rendezvous and docking with real-time, randomized optimization. In *AIAA Guidance, Navigation, and Control*, volume AIAA-2003-5511, August 2003.

[101] Jeff M. Phillips, Ran Liu, and Carlo Tomasi. Outlier robust ICP for minimizing fractional RMSD. In *6th International Conference on 3-D Digital Imaging and Modeling*, 2007.

[102] Jeff M. Phillips, Johannes Rudolph, and Pankaj K. Agarwal. Segmenting motifs in protein-protein interface surfaces. In *Proceedings of the 6th Workshop on Algorithms in Bioinformatics*, volume Volume 4175/2006 of *Lecture Notes in Computer Science*, pages 207–218. Springer Berlin / Heidelberg, September 2006.

[103] Shobha Potluri, Anthony K. Yan, James J. Chou, Bruce R. Donald, and Chris Baily-Kellogg. Structure determination of symmetric homo-oligomers by complete search of symmetry configuration space, using nmr restraints and van der Waals packing. *Proteins*, 65:203–219, 2006.

[104] C.E. Priebe, J.M. Conroy, D.J. Marchette, and Y. Park. Scan statistics on enron graphs. *Computational & Mathimatical Organization Theory*, 11(3):229–247, October 2005.

[105] R. Rado. A theorem on general measure. *Journal of the London Mathematical Society*, 21:291–300, 1947.

[106] S. Shekhar and S. Chawla. *Spatial Databases: A Tour.* Pearsons, 2001.

[107] Nisheeth Shrivastava, Subhash Suri, and Csaba D. Tóth. Detecting cuts in sensor networks. *ACM Transactions on Sensor Networks*, 4(10), 2008.

[108] Maxim Skriganov. Lattices in algebraic number fields and uniform distributions modulo 1. *Leningrad Mathematics Journal*, 1:535–558, 1990.

[109] Maxim Skriganov. Constructions of uniform distributions in terms of geometry of numbers. *St. Petersburg Mathematics Journal*, 6:635–664, 1995.

[110] Maxim Skriganov. Ergodic theory on $SL(n)$, diophantine approximations and anomalies in the lattice point problem. *Inventiones Mathematicae*, 132:1–72, 1998.

[111] Aravind Srinivasan. Improving the discrepancy bound for sparse matrices: Better approximations for sparse lattice approximation problems. In *Proceedings of the 8th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 692–701, 1997.

[112] Subhash Suri, Csaba D. Tóth, and Yunhong Zhou. Range counting over multidimensional data streams. In *Proceedings 20th Symposium on Computational Geometry*, pages 160–169, 2004.

[113] Sebastian Thrun. Robotic mapping: A survey. *Exploring Artificial Intelligence in the New Millenium*, 2002.

[114] J. G. van der Corput. Verteilungsfunktionen I. *Aka. Wet. Ams.*, 38:813–821, 1935.

[115] Marc van Kreveld and Maarten Löffler. Largest bounding box, smallest diameter, and related problems on imprecise points. In *Proc. 10th Workshop on Algorithms and Data Structures*, LNCS 4619, pages 447–458, 2007.

[116] Vladimir Vapnik and Alexey Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability and its Applications*, 16:264–280, 1971.

[117] Bei Wang, Jeff M. Phillips, Robert Schrieber, Dennis Wilkinson, Nina Mishra, and Robert Tarjan. Spatial scan statistics for graph clustering. In *8th SIAM Intenational Conference on Data Mining*, 2008.

[118] Hai Yu and Pankaj K. Agarwal. A space-optimal data-stream algorithm for coresets in the plane. In *Proceedings 23rd Symposium on Computational Geometry*, pages 1–10, 2007.

[119] Hai Yu, Pankaj K. Agarwal, Roditty Poreddy, and Kasturi Varadarajan. Practical methods for shape fitting and kinetic data structures using coresets. *Algorithmica*, to appear.

[120] Hamid Zarrabi-Zadeh. An almost space-optimal streaming algorithm for coresets in fixed dimensions. In *Proceedings of the 16th Annual Symposium on Algorithms*, volume 5193 LNCS, pages 817–829, 2008.

Biography

Jeffrey Michael Phillips was born September 18, 1980 in Milwaukee, WI. He earned a Bachelors of Science in Computer Science and a Bachelors of Arts in Mathematics from Rice University in May 2003.

Jeff was awarded a National Science Foundation Graduate Research Fellowship in 2004 and a James B. Duke Fellowship in 2003. Jeff was awarded "Best Student Paper" for his paper *Algorithms for ε-Approximations of Terrains* [96] at the International Conference on Automata, Programming, and Languages in 2008.

Jeff has also coauthored several papers [7, 8, 11, 30, 98, 99, 100, 95, 101, 102, 117] which did not directly contribute to this thesis.