Finding an Approximate Mode of a Kernel Density Estimate

Jasper C.H. Lee
Brown University, Providence, RI, USA

Jerry Li
Microsoft Research, Redmond, WA, USA

Christopher Musco
New York University, NY, USA

Jeff M. Phillips
University of Utah, Salt Lake City, UT, USA

Wai Ming Tai
University of Chicago, IL, USA

Abstract

Given points $P = \{p_1, ..., p_n\}$ subset of $\mathbb{R}^d$, how do we find a point x which approximately maximizes the function $\frac{1}{n}\sum_{p_i \in P} e^{-\|p_i - x\|^2}$? In other words, how do we find an approximate mode of a Gaussian kernel density estimate (KDE) of P? Given the power of KDEs in representing probability distributions and other continuous functions, the basic mode finding problem is widely applicable. However, it is poorly understood algorithmically. We provide fast and provably accurate approximation algorithms for mode finding in both the low and high dimensional settings. For low (constant) dimension, our main contribution is a reduction to solving systems of polynomial inequalities. For high dimension, we prove the first dimensionality reduction result for KDE mode finding. The latter result leverages Johnson-Lindenstrauss projection, Kirszbraun's classic extension theorem, and perhaps surprisingly, the mean-shift heuristic for mode finding. For constant approximation factor these algorithms run in $O(n(\log n)^{O(d)})$ and $O(nd + (\log n)^{O(\log^3 n)})$, respectively; these are proven more precisely as a $(1 + \epsilon)$-approximation guarantee. Furthermore, for the special case of $d = 2$, we give a combinatorial algorithm running in $O(n \log^2 n)$ time. We empirically demonstrate that the random projection approach and the 2-dimensional algorithm improves over the state-of-the-art mode-finding heuristics.

2012 ACM Subject Classification Theory of computation → Design and analysis of algorithms

Keywords and phrases Kernel density estimation, Dimensionality reduction, Coresets, Means-shift

Digital Object Identifier 10.4230/LIPIcs.ESA.2021.61

Funding *Jasper C.H. Lee*: Partially supported by NSF award IIS-1562657.
Christopher Musco: Partially supported by NSF CCF-2045590.
Jeff M. Phillips: Partially supported by NSF CCF-1350888, CNS-1514520, CNS-1564287, IIS-1619287, IIS-1816149, and CBET-1953350.

1 Introduction

Given a point set P in $\mathbb{R}^d$ and a kernel $K : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$, the kernel density estimate (KDE) is a function mapping from $\mathbb{R}^d$ to $\mathbb{R}$ and is defined as $\frac{1}{|P|}\sum_{p \in P} K(x, p)$ for any $x \in \mathbb{R}^d$. One common example of kernel K is the Gaussian kernel, $K(x, y) = e^{-\|x-y\|^2}$ for any $x, y \in \mathbb{R}^d$, which is the focus of this paper.

These kernel density estimates are a fundamental tool in statistics [48, 45, 18, 19] and machine learning [44, 23, 36]. For $d = 1$, KDEs with a triangular kernel ($K(x, p) = \max(0, 1 - |x - p|)$) can be seen as the average over all shifts of a fix-width histogram. And

29th Annual European Symposium on Algorithms (ESA 2021).
Editors: Petra Mutzel, Rasmus Pagh, and Grzegorz Herman; Article No. 61; pp. 61:1–61:19
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

unlike histograms these generalize naturally to a higher dimensions as a stable way to create a continuous function to represent the measure of a finite point set. Indeed, the KDEs constructed on an iid sample from any tame distribution will converge to that distribution in the limit as the sample size grows [48, 45]. Not surprisingly they are also central objects in Bayesian data analysis [27, 21]. Using Gaussian kernels (and other positive definite kernels), KDEs are members of a reproducing kernel Hilbert space [53, 50, 36] where for instance they induce a natural distance between distributions [49, 28]. Their other applications includes outlier detection [56], clustering [43], topological data analysis [40, 14], spatial anomaly detection [2, 24], and statistical hypothesis testing [23].

In this paper, we study how to find an *approximate mode* of a Gaussian KDE. An ϵ-approximate mode of a KDE is a point x' whose KDE value is at least $1-\epsilon$ times the maximum of the KDE. It is known that Gaussian KDEs can have complex structure of local maximum [20, 26], but other than some heuristic approaches [9, 10, 54, 22] there has been very little prior work [40, 2] (which we discuss shortly) in developing and formally analyzing algorithms to find this maximum. Beyond being a key descriptor (the mode) of one of the most common representations of a continuous distribution, finding the (global) maximum of a KDE has many other specific applications. It is a necessary step to create a simplicial complex to approximate superlevel sets of KDEs [40]; to localize and track objects [13, 46]; to quantify multi-modality of distributions [47]; to finding typical objects, including curves [22].

Problem Definition. For any $x, y \in \mathbb{R}^d$, we define the Gaussian kernel as $K(x,y) = e^{-\|x-y\|^2}$. The Gaussian kernel density estimate (KDE) $\overline{\mathcal{G}}_P(x)$ of a point set P is defined as $\overline{\mathcal{G}}_P(x) = \frac{1}{|P|}\sum_{p\in P} K(p,x)$, for $x \in \mathbb{R}^d$. We will sometimes use the notation $\mathcal{G}_P(x) = |P| \cdot \overline{\mathcal{G}}_P(x)$ to simplify calculations. In line with other works on optimization, we focus on the approximate version of the mode finding problem, defined as follows. Given a point set P of size n where $\max_{x\in\mathbb{R}^d} \overline{\mathcal{G}}_P(x) \geq \rho$ for some parameter ρ below which the maximum is uninteresting, and an error parameter $\epsilon > 0$, the goal is to find an ϵ-approximate mode x', such that $\overline{\mathcal{G}}_P(x') \geq (1-\epsilon)\max_{x\in\mathbb{R}^d} \overline{\mathcal{G}}_P(x)$. We assume the lower bound ρ is known to the algorithm; or we can set $\rho = 1/n$ since $\overline{\mathcal{G}}_P(p) \geq 1/n$ for any $p \in P$. In practice, one should expect that $\rho \ll \epsilon$, so we aim for algorithms with far smaller dependence on $1/\rho$ than on $1/\epsilon$.

Known Results. One trivial approach is exhaustive search. It is easy to see that the optimal point x^* cannot be too far away from the input data. More precisely, x^* should be within the radius of $\sqrt{\log\frac{1}{\rho}}$ of a point p for some $p \in P$. Given the above observation, one can construct a grid of width $\frac{1}{\epsilon\rho}$ around each point of input data and evaluate the value of $\mathcal{G}_P$ at each grid point. This approach will allow us to output a solution with additive error at most $\epsilon n\rho$. However, the size of the search space could be as large as $O\left(n\left(\sqrt{\log\frac{1}{\rho}}/\epsilon\rho\right)^d\right)$ which is infeasible in practice. A similar approach is suggested by [40].

Another approach, proposed by [2], is to compute the depth in an arrangement of a set of geometric objects. Namely, it is to find the point that maximizes the number of objects including that point. They consider a set $\mathcal{S}$ of segment in $\mathbb{R}^2$ and, for any $x \in \mathbb{R}^2$ and $s \in \mathcal{S}$, define $K(x,s) = K(x,y)$ where y is the closest point on s to x. In our setting, we treat the point set P as degenerate (length 0) segments $\mathcal{S}$. By discretizing the continuous function K into the level set of it, one can view the problem as computing the depth in an arrangement of a collection of level sets. This approach has a running time of $O(\frac{n}{\epsilon^4}\log^3 n)$ (this implicitly sets $\rho = 1/n$). One can generalizes their approach to the high dimensional case, but the running time would still be $O(n^{O(d)})$.

Related Work. As mentioned before, computing depth in an arrangement of a set of geometric object is highly related to our problem. Given a collection $\mathcal{C}$ of geometric object in $\mathbb{R}^d$, one can expressed the depth as $\sum_{c\in\mathcal{C}} \mathbf{1}_c(x)$ where $\mathbf{1}_c(x)$ is the indicator function of $x \in c$. It is easy to see that KDE is basically the same formula by replacing $\mathbf{1}_c(x)$ with $K(x,p)$. Namely, one can view finding a maximum point of KDE as computing the point of maximum "fractional" depth among kernels. Surprisingly, there are not many non-trivial algorithmic results on computing the depth of high-dimensional geometric objects. In general, whenever $\mathcal{C}$ is a collection of bounded complexity (e.g. VC dimension [52]) objects, the arrangement is always of complexity $O(n^{O(d)})$ and it can be constructed, with the depth encoded, in as much time. A celebrated case is when $\mathcal{C}$ is a collection of axis parallel box, the point of maximum depth can be found in $O(n^{d/2-o(1)})$ time [12]. For our task we can run such approaches on a sample of size $n_0 = O(\frac{d}{\epsilon^2\rho}\log\frac{1}{\rho})$ [32, 25], so the runtimes still have a $1/\rho^{O(d)}$ term.

Another line of work [11, 20, 5] attempts to bound the *number* of local maximum of a Gaussian KDE. Perhaps surprisingly, the number is greater than n for dimensions $d \geq 2$, and in fact can be at least $\binom{n}{d} + n$ for $n, d \geq 2$ [5]. It is currently unknown whether the number can be infinite, but the best upper bound *assuming finiteness* is $2^{d+\binom{n}{2}}(5+3d)^n$. So even if we could identify all of these, there still would be $\Omega(n^d)$ points to evaluate.

While not as explicit as the dimensionality reduction results we will present, other work based on LSH [6, 7] or related to geometric graphs [41] have shown properties of evaluating KDEs after what can be interpreted as forms of dimensionality reduction. For instance, Quanrud [41] shows that using dimensionality reduction, as well as other approximations and structures, one can evaluate KDEs within $1 \pm \epsilon$ error in roughly $1/\epsilon^2$ time, but also with logarithmic factors depending on, for example, spread parameters for the Gaussian kernel.

Our Approach and Result. We present an approximation scheme that reads the data (to sample it) in $O(nd)$ time, and then its runtime depends only on $1/\epsilon$ and $1/\rho$. At the heart of our algorithm are two techniques: dimensionality reduction and polynomial system solving. We also use standard coreset results for Gaussian kernel density estimates.

For dimensionality reduction (Section 3), we use Johnson-Lindenstrauss matrices to project the point set down to low dimensions, and solve the problem in low dimensions. The crucial issue is, if we solve the mode finding problem in the low dimensional space, it is not immediately clear that the original high dimensional space also has a point that gives a high KDE value. We resolve this with an application of Kirszbraun's extension theorem [30, 51], which shows the existence of such a high dimensional point. To find the actual point in the high dimensional space, we use one step of the *mean-shift* algorithm [9, 10], which is a known heuristic for the KDE maximum finding problem with provable monotonicity properties. We could alternatively combine a *terminal dimensionality reduction* result [37] with our mean-shift recovery strategy. Doing so would give the same level of dimensionality reduction, at the expense of reduced simplicity and runtime efficiency.

In low dimensions, we consider Taylor series truncations of the Gaussian kernel, and reduce the mode finding problem to solving systems of polynomial inequalities (Section 2). The result of [42] implies that one can find a solution to a system of λ polynomial inequalities with degree D and k variables in time $O((\lambda D)^{O(k)})$. Here, k will essentially be the dimensionality d of the problem, and λ will be a constant as shown in our constructions. We observe that since the optimal point must be close to one of the points in the input, we can consider a sufficiently fine grid in the vicinity of each input point, which totals to $O(n2^{O(d)})$ grid points. For each grid point, we formulate and solve a system of polynomial inequalities based on Taylor expansions, up to $O(\log\frac{1}{\rho})$ terms around that grid point. This gives a running time of $O(n(\log\frac{1}{\rho})^{O(d)})$, where n is the size of the input point set.

Combining the above ideas with standard coreset results (small subsets $Q \subset P$ so $\overline{\mathcal{G}}_Q$ approximates $\overline{\mathcal{G}}_P$) yields approximation schemes for the KDE mode finding problem. We present two such schemes, one with exponential runtime dependence on the dimensionality d which is more suitable for low dimensions, and another with only linear dependence in d (which is necessary for reading the input) and is designed for the high dimensional regime. Guarantees of these approximations are captured by Theorems 1 and 2; we provide algorithmic details and intuition, but proofs are in the appendices.

▶ **Theorem 1** (Low dimensional regime). *Given $\epsilon, \rho > 0$ and a point set $P \subset \mathbb{R}^d$ of size n with $\overline{\mathcal{G}}_P(x^*) \geq \rho$, where $x^* = \operatorname{argmax}_{x \in \mathbb{R}^d} \overline{\mathcal{G}}_P(x)$, we can find $x' \in \mathbb{R}^d$ so $\overline{\mathcal{G}}_P(x') \geq (1-\epsilon)\overline{\mathcal{G}}_P(x^*)$ in $O\left(nd + \frac{d}{\epsilon^2\rho} \cdot \log \frac{1}{\rho\delta} \cdot \left(\log \frac{d}{\epsilon\rho}\right)^{O(d)}\right)$ time with probability at least $1-\delta$.*

▶ **Theorem 2** (High dimensional regime). *Given $\epsilon, \rho > 0$ and a point set $P \subset \mathbb{R}^d$ of size n with $\overline{\mathcal{G}}_P(x^*) \geq \rho$, where $x^* = \operatorname{argmax}_{x \in \mathbb{R}^d} \overline{\mathcal{G}}_P(x)$, we can find $x' \in \mathbb{R}^d$ so $\overline{\mathcal{G}}_P(x') \geq (1-\epsilon)\overline{\mathcal{G}}_P(x^*)$ in $O\left(nd + \left(\log \frac{1}{\epsilon\rho}\right)^{O(\frac{1}{\epsilon^2}\log^3 \frac{1}{\epsilon\rho})} \cdot \log \frac{1}{\delta} + \min\{nd \log \frac{1}{\delta}, \frac{d}{\epsilon^2\rho^2} \log^2 \frac{1}{\delta}\}\right)$ time with probability at least $1-\delta$.*

One may set the relative error parameter ϵ, and failure probability δ to constants, and observe the mode of $\overline{\mathcal{G}}_P$ must be at least $1/n$ and set $\rho = 1/n$. Then the runtimes become $O(n(\log n)^{O(d)})$ for constant dimensions, and $O(nd + (\log n)^{O(\log^3 n)})$ in high dimensions.

In addition to our result in Theorems 1 and 2, we also consider the special case where $d = 2$. We present a combinatorial algorithm for the 2-dimensional regime which is easier to implement. Here, we borrow the idea from [2] which is to compute the depth. Instead of simply considering the level sets of the Gaussian kernel (which are circles in our setting), we consider a more involved decomposition. One important property of the Gaussian kernel is its multiplicatively separability – namely, the Gaussian kernel can be decomposed into factors, with one factor for each dimension. We now discretize each factor into level sets (which are simply intervals) and then consider their Cartesian products, generating a collection of axis-parallel rectangles. A similar idea was also suggested by [38]. Finally, if we compute the depth of this collection of axis-parallel rectangles, we can find out an approximate mode in time $O(\frac{1}{\epsilon^2\rho} \log^2 \frac{1}{\rho})$. This approach also works in higher dimensions, but it would yield a slower running time than our general approaches in Theorems 1 and 2. The formal guarantees of this 2-d algorithm are captured in Theorem 3, and proven in the appendices.

▶ **Theorem 3** (2-dimensional setting). *Given $\epsilon, \rho > 0$ and a point set $P \subset \mathbb{R}^2$ of size n such that $\overline{\mathcal{G}}_P(x^*) \geq \rho$, where $x^* = \operatorname{argmax}_{x \in \mathbb{R}^d} \overline{\mathcal{G}}_P(x)$, we can find $x' \in \mathbb{R}^2$ so $\overline{\mathcal{G}}_P(x') \geq (1-\epsilon)\overline{\mathcal{G}}_P(x^*)$ in $O\left(n + \frac{1}{\epsilon^2\rho}(\log \frac{1}{\rho} + \log \frac{1}{\delta}) \log(\frac{1}{\epsilon\rho} \log \frac{1}{\delta})\right)$ time with probability at least $1-\delta$.*

There are different extensions of our problem formulation. For example, one can define the weighted KDE of a point set P, $\sum_{p \in P} w_p K(x,p)$, and find its mode. Another common extension is to consider non-spherical Gaussians with different variances. We expect that our technique with some straightforward modifications work for these extensions and will omit the details.

2 KDE Mode Finding via System of Polynomials

In this section we provide algorithms that approximately find the maximum of the Gaussian KDE in $\mathbb{R}^d$. We first define the following notations. For a point $p \in \mathbb{R}^d$ and $r > 0$, we define $B_p(r)$ as $\left\{y \in \mathbb{R}^d \mid \|y - p\| \leq r\right\}$, namely the Euclidean ball around p. For a point set $P \subset \mathbb{R}^d$ and $r > 0$, we define $B_P(r)$ as $\cup_{p\in P} B_p(r)$, that is the union of Euclidean balls around all the points in P. For a point set $P \subset \mathbb{R}^d$, a point $q \in \mathbb{R}^d$ and $r > 0$, also define $Q_{P,q}(r) = P \cap B_q(r\sqrt{\log \frac{1}{\epsilon\rho}})$. Finally, let $\mathsf{Grid}(\gamma)$ be the infinite grid $\{x = (i_1\gamma, i_2\gamma, \ldots, i_d\gamma) \mid i_1, i_2, \ldots, i_d \text{ are integers}\}$, parametrized by a cell length $\gamma > 0$.

We first make an observation that the maximum point must be close to one of the data points, captured by Observation 4.

▶ **Observation 4.** $x^* \in B_P(\sqrt{\log \frac{1}{\rho}})$. *Recall that* $x^* = \operatorname{argmax}_{x\in\mathbb{R}^d} \mathcal{G}_P(x)$.

Proof. Suppose $x^* \notin B_P(\sqrt{\log \frac{1}{\rho}})$. Then, $\mathcal{G}_P(x^*) = \sum_{p\in P} e^{-\|p-x^*\|^2} < \sum_{p\in P} \rho = n\rho$. However, $\mathcal{G}_P(x^*) \geq n\rho$ by assumption. ◀

The algorithm presented in this section relies crucially on the result of Renegar for solving systems of polynomial inequalities, as stated in the following lemma.

▶ **Lemma 5** ([42]). *Consider λ polynomial inequalities with maximum degree D and k variables. There is an algorithm either finds a solution that satisfies all λ polynomial inequalities or returns NO SOLUTION in $O((\lambda D)^{O(k)})$ time.*

Before we give details of our algorithm, we present the family of systems of polynomial inequalities we formulate for mode finding. Let $\mathsf{SysPoly}(P, q, r, r', \beta)$ be the following system.

$$\sum_{p\in Q_{P,q}(r')} \prod_{i=1}^{d} \left(\sum_{j=0}^{s-1} \frac{1}{j!} \left(-(x_i - p_i)^2\right)^j \right) \geq \beta \quad \bigwedge \quad \|x - q\|^2 \leq r^2 \log \frac{1}{\epsilon\rho}$$

where $s = (r + r')^2 e^2 \log \frac{d}{\epsilon\rho}$. Intuitively, if a point $x \in \mathbb{R}^d$ satisfies the left inequality of $\mathsf{SysPoly}(P, q, r, r', \beta)$, then the value $\mathcal{G}_P(x)$ is larger than a threshold that is approximately β. It is because the LHS of the left inequality is the sum of the truncated Taylor expansion of the Gaussians centered at p that is around q. On the other hand, the truncated Taylor expansion only gives a good approximation locally. Hence, the right inequality of $\mathsf{SysPoly}(P, q, r, r', \beta)$ ensures that x is around q.

Also, let $\mathsf{SysPoly}(P, q, r, r')$ be the algorithm that performs binary search on β of the above system $\mathsf{SysPoly}(P, q, r, r', \beta)$ and terminates when the search gap is less than $\frac{1}{10}|P|\epsilon\rho$. Note that β lies between 0 and $O(|P|)$ which means we need $O\left(\log\left(|P|/\frac{1}{10}|P|\epsilon\rho\right)\right) = O\left(\log\frac{1}{\epsilon\rho}\right)$ iterations in binary search. The total running time of $\mathsf{SysPoly}(P, q, r, r')$ is $O\left((4s)^{O(d)} \log\frac{1}{\epsilon\rho}\right) = O(s^{O(d)})$ since $k = d$, $\lambda = 2$ and $D = 2s$ in Lemma 5.

The following lemma captures the approximation error from the Taylor series truncation.

▶ **Lemma 6.** *Suppose $r + r' > 1$ and $q \in \mathbb{R}^d$ such that $\|x^* - q\| \leq r\sqrt{\log \frac{1}{\epsilon\rho}}$. Then, the output $x^{(q)}$ of $\mathsf{SysPoly}(P, q, r, r')$ satisfies $\mathcal{G}_{Q_{P,q}(r')}(x^{(q)}) \geq \mathcal{G}_{Q_{P,q}(r')}(x^*) - |P|\frac{\epsilon\rho}{2}$.*

In short, it shows that the truncation of the above infinite summation of polynomial terms (wrapped in a sum over all points Q, and the product over d dimensions) induces an error terms $\mathcal{E}(x^{(q)})$ and $\mathcal{E}(x^*)$ at $x^{(q)}$ and x^*, respectively. We can show that the difference between these terms is at most $\epsilon\rho$ for our choice of s, as desired.

Proof. First, we write $\sum_{p\in Q_q(r')} e^{-\|p-x^{(q)}\|^2}$ into the following form.

$$\begin{aligned}\sum_{p\in Q_q(r')} e^{-\|p-x^{(q)}\|^2} &= \sum_{p\in Q_q(r')} \prod_{i=1}^{d} \left(\sum_{j=0}^{\infty} \frac{1}{j!}\left(-(x_i^{(q)}-p_i)^2\right)^j\right)\\ &= \sum_{p\in Q_q(r')} \prod_{i=1}^{d} \left(\sum_{j=0}^{s} \frac{1}{j!}\left(-(x_i^{(q)}-p_i)^2\right)^j\right) + \mathcal{E}(x^{(q)})\end{aligned}$$

where $\mathcal{E}(x) = \sum_{p\in Q_q(r')} \sum_{j_1,\ldots,j_d|\text{one of } j_i \,\geq\, s} \frac{1}{j_1!\cdots j_d!}\left(-(x_1-p_1)^2\right)^{j_1}\cdots\left(-(x_d-p_d)^2\right)^{j_d}$ for any $x\in\mathbb{R}^d$.

Now, we have

$$\begin{aligned}\sum_{p\in Q_q(r')} e^{-\|x^{(q)}-p\|^2} &= \sum_{p\in Q_q(r')} \prod_{i=1}^{d} \left(\sum_{j=0}^{s} \frac{1}{j!}\left(-(x_i^{(q)}-p_i)^2\right)^j\right) + \mathcal{E}(x^{(q)})\\ &\geq \sum_{p\in Q_q(r')} \prod_{i=1}^{d} \left(\sum_{j=0}^{s} \frac{1}{j!}\left(-(x_i^{*}-p_i)^2\right)^j\right) - |P|\,\frac{\epsilon\rho}{10} + \mathcal{E}(x^{(q)})\\ &\geq \sum_{p\in Q_q(r')} e^{-\|x^*-p\|^2} - |P|\,\frac{\epsilon\rho}{10} + \mathcal{E}(x^{(q)}) - \mathcal{E}(x^*)\end{aligned}$$

In order to analyze the term $\mathcal{E}(x^{(q)})$ and $\mathcal{E}(x^*)$, we can first analyze the term

$$\left|\sum_{j_1,\ldots,j_d|\text{one of } j_i \,\geq\, s} \frac{1}{j_1!\cdots j_d!}\alpha_1^{j_1}\cdots\alpha_d^{j_d}\right|$$

where $\alpha_i = -(y_i-p_i)^2$ where y is $x^{(q)}$ or x^*.

$$\sum_{j_1,\ldots,j_d|\text{one of } j_i \,\geq\, s} \left(\prod_{i=1}^{d}\frac{1}{j_i!}\alpha_i^{j_i}\right) = \sum_{i=1}^{d}\left(\prod_{k=1}^{i-1}\sum_{j=0}^{s-1}\frac{1}{j!}\alpha_k^j\right)\left(\sum_{j=s}^{\infty}\frac{1}{j!}\alpha_i^j\right)\left(\prod_{k=i+1}^{d}\sum_{j=0}^{\infty}\frac{1}{j!}\alpha_k^j\right)$$

For each $i=1,2,\ldots,d$, by taking $s=(r+r')^2e^2\log\frac{d}{\epsilon\rho}$,

$$\left|\sum_{j=s}^{\infty}\frac{1}{j!}\alpha_i^j\right| \leq \sum_{j=s}^{\infty}\frac{1}{j!}\left|\alpha_i\right|^j \leq \max_{\xi\in[-|\alpha_i|,|\alpha_i|]}\frac{e^{\xi}}{s!}\left|\alpha_i\right|^s$$

The last inequality is the error approximation of Taylor expansion of exponential function. Note that $|\alpha_i| = (y_i-p_i)^2 \leq \|y-p\|^2 \leq (\|y-q\|+\|p-q\|)^2 \leq \left(r\sqrt{\log\frac{1}{\epsilon\rho}}+r'\sqrt{\log\frac{1}{\epsilon\rho}}\right)^2 \leq (r+r')^2\log\frac{1}{\epsilon\rho}$. We have

$$\begin{aligned}\left|\sum_{j=s}^{\infty}\frac{1}{j!}\alpha_i^j\right| &\leq \frac{e^{(r+r')^2\log\frac{1}{\epsilon\rho}}}{s!}((r+r')^2\log\frac{1}{\epsilon\rho})^s\\ &\leq \frac{e^{(r+r')^2\log\frac{1}{\epsilon\rho}}}{s^s}((r+r')^2e\log\frac{1}{\epsilon\rho})^s &&\text{by } s!\geq(\frac{s}{e})^s\\ &\leq \frac{e^{(r+r')^2\log\frac{1}{\epsilon\rho}}}{e^s} \leq (\frac{\epsilon\rho}{d})^{(r+r')^2(e^2-1)} &&\text{recall that } s=(r+r')^2e^2\log\frac{d}{\epsilon\rho}\\ &\leq \frac{\epsilon\rho}{20d} &&\text{by } r+r'>1 \text{ and for sufficient small } \epsilon\rho\end{aligned}$$

Now, we can plug this into $\left|\sum_{j_1,\dots,j_d|\text{one of } j_i \ge s} \frac{1}{j_1!\cdots j_d!}\alpha_1^{j_1}\cdots\alpha_d^{j_d}\right|$.

$$\begin{aligned}
&\left|\sum_{j_1,\dots,j_d|\text{one of } j_i \ge s} \frac{1}{j_1!\cdots j_d!}\alpha_1^{j_1}\cdots\alpha_d^{j_d}\right| \\
&= \left|\sum_{i=1}^{d}\left(\prod_{k=1}^{i-1}\sum_{j=0}^{s-1}\frac{1}{j!}\alpha_k^j\right)\left(\sum_{j=s}^{\infty}\frac{1}{j!}\alpha_i^j\right)\left(\prod_{k=i+1}^{d}\sum_{j=0}^{\infty}\frac{1}{j!}\alpha_k^j\right)\right| \\
&\le \sum_{i=1}^{d}\left(\prod_{k=1}^{i-1}(1+\frac{\epsilon\rho}{10d})\right)\left(\frac{\epsilon\rho}{10d}\right)\left(\prod_{k=i+1}^{d}e^{\alpha_k}\right) \\
&\le \left(1+\frac{\epsilon\rho}{20d}\right)^d\frac{\epsilon\rho}{20} \le e^{\frac{\epsilon\rho}{20}}\frac{\epsilon\rho}{20} \le \frac{\epsilon\rho}{8} \qquad \text{for sufficient small } \epsilon\rho
\end{aligned}$$

That means

$$\begin{aligned}
\sum_{p\in Q_q(r')} e^{-\|x^{(q)}-p\|^2} &\ge \sum_{p\in Q_q(r')} e^{-\|x^*-p\|^2} - |P|\frac{\epsilon\rho}{10} + \mathcal{E}(x^{(q)}) - \mathcal{E}(x^*) \\
&\ge \sum_{p\in Q_q(r')} e^{-\|x^*-p\|^2} - |P|\frac{\epsilon\rho}{10} - |Q_{P,q}(r')|\frac{\epsilon\rho}{8} - |Q_{P,q}(r')|\frac{\epsilon\rho}{8} \\
&= \sum_{p\in Q_q(r')} e^{-\|x^*-p\|^2} - |P|\frac{\epsilon\rho}{2}.
\end{aligned}$$

◀

2.1 Algorithm for Searching Polynomial Systems in Neighborhoods

A first attempt invokes Lemma 6 in a ball $B_p\left(\sqrt{\log\frac{1}{\rho}}\right)$ around each $p \in P$. However, to find out the subset of points that lie inside the ball $B_p\left(\sqrt{\log\frac{1}{\rho}}\right)$ for each $p \in P$, one needs to search linearly over P naively. Therefore, it requires $\Omega(n^2)$ runtime.

Rather, following Algorithm 1, we create a set G_P of neighborhoods, defined by the subset of $\mathsf{Grid}\left(2\sqrt{\frac{\log\frac{1}{\epsilon\rho}}{d}}\right)$ which is within $4\sqrt{\log\frac{1}{\epsilon\rho}}$ of some point $p \in P$. For each $q \in \mathsf{G}_P$ we define a neighborhood set $Q_{P,q}(4)$, and run the algorithm in Lemma 5. Again we return the output with associated maximum $\mathcal{G}_{Q_{P,q}()}(\cdot)$ value, which satisfies Theorem 7.

Algorithm 1 Solving System of Polynomial using an Infinite Grid.

```
input: a point set $P \subset \mathbb{R}^d$, parameter $\epsilon, \rho > 0$
for each $p \in P$ do
    insert $p$ into $Q_{P,q}(4)$ for each $q \in B_p(4\sqrt{\log\frac{1}{\epsilon\rho}}) \cap \mathsf{Grid}(2\sqrt{\frac{\log\frac{1}{\epsilon\rho}}{d}})$
Let $\mathsf{G}_P$ be the $q \in \mathsf{Grid}(2\sqrt{\frac{\log\frac{1}{\epsilon\rho}}{d}})$ such that $Q_{P,q}(4)$ is non empty
for each $q \in \mathsf{G}_P$ do
    Let $x^{(q)}$ be the solution to $\mathsf{SysPoly}(P, q, 2, 4)$ by the algorithm in Lemma 5
return $x' = \mathrm{argmax}_{q\in\mathsf{G}_P}\,\mathcal{G}_{Q_{P,q}(4)}(x^{(q)})$
```

▶ **Theorem 7.** *Given $0 < \epsilon, \rho < 1/2$ and a point set $P \subset \mathbb{R}^d$ of size n, let $x^* = \mathrm{argmax}_{x\in\mathbb{R}^d}\,\overline{\mathcal{G}}_P(x)$. If $\overline{\mathcal{G}}_P(x^*) \ge \rho$, we find $x' \in \mathbb{R}^d$ with $\overline{\mathcal{G}}_P(x') \ge \overline{\mathcal{G}}_P(x^*) - \epsilon\rho$ in time*

$$O\left(n\cdot\log n\cdot(2\sqrt{2e\pi})^d + n\cdot\left(\log\frac{d}{\epsilon\rho}\right)^{O(d)}\right).$$

Proof. We need to argue that x^* must be contained in some neighborhood $B_q(2\sqrt{\log\frac{1}{\epsilon\rho}})$ for some $q \in \mathsf{G}_P$, and then apply Lemma 5 with $r = 2$ and $r' = 4$. This follows since, by Lemma 4, $x^* \in B_p(\sqrt{\log\frac{1}{\rho}}) \subset B_p(\sqrt{\log\frac{1}{\epsilon\rho}})$ for some $p \in P$. Then let $q \in \mathsf{G}_P$ be the closest grid point to that point p; the distance $\|p - q\| \leq \sqrt{d}\gamma/2 = \sqrt{\log\frac{1}{\epsilon\rho}}$ with $\gamma = 2\sqrt{\log(1/\epsilon\rho)/d}$. Then by triangle inequality $\|q - x^*\| \leq \|q - p\| + \|p - x^*\| \leq 2\sqrt{\log\frac{1}{\epsilon\rho}}$. Now, we conclude that the output of $\mathsf{SysPoly}(P, p, 2, 4)$ satisfies

$$\begin{aligned}\mathcal{G}_P(x') &\geq \mathcal{G}_{Q_{P,q}(4)}(x') = \mathcal{G}_{Q_{P,q}(4)}(x^{(q)}) \geq \mathcal{G}_{Q_{P,q}(4)}(x^*) - |P|\,\frac{\epsilon\rho}{2}\\ &= \sum_{p\in P} e^{-\|p-x^*\|^2} - \sum_{p\notin Q_{P,q}(4)} e^{-\|p-x^*\|^2} - |P|\,\frac{\epsilon\rho}{2}\end{aligned}$$

Note that $\|x^* - p\| \geq \|q - p\| - \|q - x^*\| \geq 4\sqrt{\log\frac{1}{\epsilon\rho}} - 2\sqrt{\log\frac{1}{\epsilon\rho}} = 2\sqrt{\log\frac{1}{\epsilon\rho}}$ since $x^* \in B_q\left(2\sqrt{\log\frac{1}{\epsilon\rho}}\right)$ and $p \notin B_q\left(4\sqrt{\log\frac{1}{\epsilon\rho}}\right)$.

$$\mathcal{G}_P(x') \geq \mathcal{G}_P(x^*) - |Q_{P,q}(4)|\,(\epsilon\rho)^4 - |P|\,\frac{\epsilon\rho}{2} \geq \mathcal{G}_P(x^*) - |P|\,\epsilon\rho \qquad \text{since } \epsilon, \rho < 1/2$$

We now compute the running time. First, to construct $Q_{P,q}(4)$ (for notation convenience, we use Q_q instead), for each $p \in P$, we enumerate all $q \in B_p\left(4\sqrt{\log\frac{1}{\epsilon\rho}}\right) \cap \mathsf{Grid}\left(2\sqrt{\frac{\log\frac{1}{\epsilon\rho}}{d}}\right)$ and insert p into Q_q. Since, by considering the volume of high dimensional sphere, there are $O\left(\frac{\pi^{d/2}}{\Gamma(\frac{d}{2}+1)}\left(4\sqrt{\log\frac{1}{\epsilon\rho}}\Big/2\sqrt{\frac{\log\frac{1}{\epsilon\rho}}{d}}\right)^d\right) = O\left((2\sqrt{2e\pi})^d\right)$ points in $B_p\left(4\sqrt{\log\frac{1}{\epsilon\rho}}\right) \cap \mathsf{Grid}\left(2\sqrt{\frac{\log\frac{1}{\epsilon\rho}}{d}}\right)$ for each $p \in P$, we have $\sum_{q\in\mathsf{G}_P} |Q_q| = O(n(2\sqrt{2e\pi})^d)$ and also there are only $O(n(4\sqrt{2e\pi})^d)$ non empty Q_q. Here, Γ is the gamma function and we use the fact of $\Gamma(x+1) \geq (\frac{x}{e})^x$. It is easy to construct a data structure to insert all p into all of the corresponding Q_q in $O\left(n(2\sqrt{2e\pi})^d \log\left(n(2\sqrt{2e\pi})^d\right)\right) = O\left(n(2\sqrt{2e\pi})^d(\log n + d)\right)$. Let $s = 36e^2 \log\frac{d}{\epsilon\rho}$. We now can precompute each polynomial $\prod_{i=1}^d \left(\sum_{j=0}^{s-1} \frac{1}{j!}\left(-(x_i - p_i)^2\right)^j\right)$ in $O(d(2s)^d)$ time for each $p \in P$ which takes $O(nd(2s)^d)$ total time to compute all of them. For each $q \in \mathsf{G}_P$, it takes $O(|Q_q|\,(2s)^d)$ to construct the polynomial and $O(s^{O(d)})$ time to solve the system of polynomial as suggested in Lemma 5. Therefore, the total running time is

$$\begin{aligned}&O\left(n(2\sqrt{2e\pi})^d(\log n + d) + nd(2s)^d + \sum_{q\in\mathsf{G}_P}(|Q_q|\,(2s)^d + s^{O(d)}\right)\\ &\quad = O\left(n \cdot \log n \cdot (2\sqrt{2e\pi})^d + n \cdot \left(\log\frac{d}{\epsilon\rho}\right)^{O(d)}\right).\end{aligned}$$ ◀

To achieve our final result for low dimensionality, we pre-process the input P by constructing, under the assumption that $\max_x \overline{\mathcal{G}}_P(x) \geq \rho$, a $(1 - \epsilon/3)$-approximation coreset from [55] of size $O(\frac{d}{\epsilon^2}\frac{1}{\rho}(\log\frac{1}{\rho} + \log\frac{1}{\delta}))$. Running Algorithm 1 on this coreset yields Theorem 1.

Proof of Theorem 1. Let $x^{**} = \operatorname{argmax}_{x\in\mathbb{R}^d} \overline{\mathcal{G}}_{P_1}(x)$. We first have $\overline{\mathcal{G}}_{P_1}(x^{**}) \geq \overline{\mathcal{G}}_{P_1}(x^*) \geq (1-\frac{1}{3}\epsilon)\overline{\mathcal{G}}_P(x^*) = \Omega(\rho)$ for small ϵ. By Theorem 7 and reparameterizing ϵ, we have

$$\begin{aligned}
\overline{\mathcal{G}}_P(x') &\geq \overline{\mathcal{G}}_{P_1}(x') - \frac{1}{3}\epsilon M_{x'} && \text{by the construction of } P_1\\
&\geq \overline{\mathcal{G}}_{P_1}(x^{**}) - \frac{1}{3}\epsilon\rho - \frac{1}{3}\epsilon M_{x'} && \text{since } \overline{\mathcal{G}}_{P_1}(x^{**}) = \Omega(\rho) \text{ and by Theorem 7}\\
&\geq \overline{\mathcal{G}}_{P_0}(x^{*}) - \frac{1}{3}\epsilon\rho - \frac{1}{3}\epsilon M_{x'} &&\\
&\geq (1-\frac{1}{3}\epsilon)\overline{\mathcal{G}}_P(x^*) - \frac{1}{3}\epsilon\rho - \frac{1}{3}\epsilon M_{x'} && \text{by } \overline{\mathcal{G}}_P(x^*) \geq \rho \text{ and construction of } P_1\\
&\geq (1-\epsilon)\overline{\mathcal{G}}_P(x^*) && \text{since } M_{x'} \leq \overline{\mathcal{G}}_P(x^*)
\end{aligned}$$

The final running time is $O(nd)$ to read data and construct P_1 plus

$$O\left(n_1 \cdot \log n_1 \cdot (2\sqrt{2e\pi})^d + n_1 \cdot \left(\log\frac{d}{\epsilon\rho}\right)^{O(d)}\right) = O\left(\frac{d}{\epsilon^2\rho}\cdot\log\frac{1}{\rho\delta}\cdot\left(\log\frac{d}{\epsilon\rho}\right)^{O(d)}\right). \quad ◀$$

3 Dimensionality Reduction for KDE Mode Finding

Leveraging Kirszbraun's extension theorem, we prove that compressing $P = \{p_1,\ldots,p_n\}$ using a Johnson-Lindenstrauss random projection to $O\left(\log n \log^2(1/\epsilon\rho)/\epsilon^2\right)$ dimensions preserves the mode of the KDE with centers in P, to a $(1-\epsilon)$ factor. Crucially, we then show that it is possible to recover an approximate mode for P from a solution to the low dimensional problem by applying a *single iteration* of the mean-shift algorithm.

In Section 3.1 we combine this result with our low dimensional algorithm from Section 2 and existing coreset results for KDEs (which allow us to eliminate the $\log n$ dependence) to give our final algorithm for high-dimensional mode finding. We first present the dimensionality reduction result in isolation as, like dimensionality reduction strategies for other computational hard problems [15, 35, 8], it could in principle be combined with any other heuristic or approximate mode finding method. For example, we show a practical strategy is to solve the low-dimensional problem using the mean-shift heuristic.

We need one basic definition before outlining our approach in Algorithm 2.

▶ **Definition 8** ((γ, k, δ)-Johnson-Lindenstrauss Guarantee). *A randomly selected matrix $\Pi \in \mathbb{R}^{m\times d}$ satisfies the (γ, k, δ)-JL Guarantee if, for any k data points $v_1,\ldots,v_k \in \mathbb{R}^d$,*

$$\|v_i - v_j\| \leq \|\Pi v_i - \Pi v_j\| \leq (1+\gamma)\|v_i - v_j\|,$$

for all pairs $i, j \in 1,\ldots,k$ simultaneously, with probability $(1-\delta)$.

Definition 8 is satisfied by many possible constructions. When Π is a properly scaled random Gaussian or sign matrix, it satisfies the (γ, k, δ)-JL guarantee as long as $m = O(\log(k/\delta)/\gamma^2)$ [17, 1]. In this case, Π can be multiplied by a d dimensional vector in $O(md)$ time. For simplicity, we assume such a construction is used in our algorithm. Other constructions, including fast Johnson-Lindenstrauss transforms [3, 4, 31] and sparse random projections [29, 16] satisfy the definition with slightly larger m, but faster multiplication time. Depending on problem parameters, using such constructions may lead to a slightly faster overall runtime.

Algorithm 2 Dimensionality Reduction for KDE mode finding.

```
input: a set of $n$ points $P \subset \mathbb{R}^d$, parameters $\epsilon, \delta > 0$, $\rho$ such that $\max_x G_P(x) \geq \rho n$
output: a point $x' \in \mathbb{R}^d$ satisfying $G_P(x') \geq (1-\epsilon)\max_x G_P(x)$ with prob. $1-\delta$
Set $\gamma = \frac{\epsilon}{4\log(4/\epsilon\rho)}$.
Choose a random matrix $\Pi \in \mathbb{R}^{m \times d}$ satisfying the $(\gamma, n+1, \delta)$-JL guarantee (Defn. 8).
For each $p_i \in P$, compute $\Pi p_i$ and let $\Pi P$ denote the data set $\{\Pi p_1, \ldots, \Pi p_n\}$
Using an algorithm for mode finding in low dimensions (e.g. Algorithm 1) find a point
    $x''$ satisfying $\mathcal{G}_{\Pi P}(x'') \geq (1-\epsilon/2)\max_{x\in\mathbb{R}^m}\mathcal{G}_{\Pi P}(x)$.
return $x' = \frac{\sum_{p\in P} p \cdot e^{-\|x''-\Pi p\|^2}}{\sum_{p\in P} e^{-\|x''-\Pi p\|^2}}$
```

▶ **Theorem 9.** *With probability $(1-\delta)$, Algorithm 2 returns an x' satisfying $\mathcal{G}_P(x') \geq (1-\epsilon)\max_x \mathcal{G}_P(x)$. When implemented with a random Rademacher or Gaussian Π, the algorithm runs in time $O(ndm) + T_{m,(1-\epsilon/2)}$, where $m = O\left(\frac{\log(n/\delta)\log^2(1/\epsilon\rho)}{\epsilon^2}\right)$ and $T_{m,(1-\epsilon)}$ is the time required to compute a $(1-\epsilon/2)$ approximate mode for an $O(m)$ dimension dataset.*

The runtime claim is immediate, so we focus on proving the correctness of Algorithm 9. The following key lemma is the main structural result, that the mode of our dimensionality reduced problem has approximately the same density as that of the original. Its proof, as we see below, crucially uses Kirszbraun's extension theorem.

▶ **Lemma 10.** *Suppose Π is a projection satisfying the $(\gamma, n+1, \delta)$-JL guarantee, then*

$$(1-\epsilon/2)\max_{x\in\mathbb{R}^d}\mathcal{G}_P(x) \leq \max_{x\in\mathbb{R}^m}\mathcal{G}_{\Pi P}(x) \leq \max_{x\in\mathbb{R}^d}\mathcal{G}_P(x) \tag{1}$$

Proof. Let $x^* = \operatorname{argmax}_x \mathcal{G}_P(x)$. Since Π was chosen to satisfy the $(\gamma, n+1, \delta)$-JL property with $\gamma = \frac{\epsilon}{4\log(4/\epsilon\rho)}$, we have that, with probability at least $1-\delta$, for all $y, z \in \{x^*\} \cup P$,

$$\|y-z\|^2 \leq \|\Pi y - \Pi z\|^2 \leq \left(1 + \frac{\epsilon}{4\log(4/\epsilon\rho)}\right)\|y-z\|^2. \tag{2}$$

The rest of our analysis conditions on this fact being true. We first prove the left side of (1). From (2), we have that $\|\Pi x^* - \Pi p\|^2 \leq (1 + \frac{\epsilon}{4\log(4/\epsilon\rho)})\|x^* - p\|^2$ for all $p \in P$. Accordingly,

$$\begin{aligned}\max_{x\in\mathbb{R}^m}\mathcal{G}_{\Pi P}(x) &\geq \mathcal{G}_{\Pi P}(\Pi x^*) = \sum_{p\in P} e^{-\|\Pi x^* - \Pi p\|^2} \geq \sum_{p\in P} e^{-(1+\frac{\epsilon}{4\log(4/\epsilon\rho)})\|x^*-p\|^2} \\ &\geq \sum_{\substack{p\in P \\ \|x^*-p\|^2 < \log(4/\epsilon\rho)}} e^{-\|x^*-p\|^2} e^{-\frac{\epsilon}{4\log(4/\epsilon\rho)}\|x^*-p\|^2} \geq (1-\epsilon/4) \sum_{\substack{p\in P \\ \|x^*-p\|^2 < \log(4/\epsilon\rho)}} e^{-\|x^*-p\|^2}.\end{aligned} \tag{3}$$

The last step uses that $e^{-\frac{\epsilon}{4\log(4/\epsilon\rho)}\|x-p\|^2} \geq 1 - \epsilon/4$ when $\|x-p\|^2 \leq \log(4/\epsilon\rho)$. Next we have

$$\sum_{\substack{p\in P \\ \|x^*-p\|^2 < \log\frac{4}{\epsilon\rho}}} e^{-\|x^*-p\|^2} \geq \sum_{p\in P} e^{-\|x^*-p\|^2} - \epsilon n \rho = \mathcal{G}_P(x^*) - \epsilon n\rho \geq (1-\epsilon/4)\mathcal{G}_P(x^*).$$

This statement follows from two facts: 1) If $\|x-p\|^2 \geq \log\frac{4}{\epsilon\rho}$ then $e^{-\|x-p\|^2} \leq \epsilon\rho/4$ and 2) we assume that $\mathcal{G}_P(x^*) \geq \rho n$. Combining with (3) we conclude that $\mathcal{G}_{\Pi P}(x) \geq (1-\epsilon/2)\mathcal{G}_P(x^*)$.

We are left to prove the right hand side of (1). To do so, we rely on the classic Kirszbraun extension theorem for Lipschitz functions, which is stated as follows:

▶ **Theorem 11** (Kirszbraun Theorem [30, 51]). *For any $\mathcal{S} \subset \mathbb{R}^z$, let $f : S \to \mathbb{R}^w$ be an L-Lipschitz function: for all $x, y \in \mathcal{S}$, $\|f(x) - f(y)\|_2 \leq L\|x - y\|_2$. Then there always exists some extension $\tilde{f} : \mathbb{R}^z \to \mathbb{R}^w$ of f to the entirety of $\mathbb{R}^z$ such that:*

1. *$\tilde{f}(x) = f(x)$ for all $x \in S$,*
2. *$\tilde{f}$ is also L-Lipschitz: for all $x, y \in \mathbb{R}^z$, $\|\tilde{f}(x) - \tilde{f}(y)\|_2 \leq L\|x - y\|_2$.*

We will apply this theorem to the function $g : \{\Pi x^*\} \cup \Pi P \to \{x^*\} \cup P$ with $g(\Pi y) = y$ for any $y \in \{x^*\} \cup P$. By (2), we have that g is 1-Lipschitz. It follows that there is some function $\tilde{g} : \mathbb{R}^m \to \mathbb{R}^d$ which agrees with g on inputs $\{\Pi x^*\} \cup P$ and satisfies $\|\tilde{g}(s) - \tilde{g}(t)\| \leq \|s - t\|$ for all $s, t \in \mathbb{R}^m$. This fact can be used to establish that, for any $x \in \mathbb{R}^m$, $\mathcal{G}_{\Pi P}(x) \leq \mathcal{G}_P(\tilde{g}(x))$:

$$\mathcal{G}_{\Pi P}(x) = \sum_{p \in P} e^{-\|x - \Pi p\|^2} \leq \sum_{p \in P} e^{-\|\tilde{g}(x) - \tilde{g}(\Pi p)\|^2} = \sum_{p \in P} e^{-\|\tilde{g}(x) - p\|^2} = \mathcal{G}_P(\tilde{g}(x)).$$

It thus follows that $\max_x \mathcal{G}_{\Pi P}(x) \leq \max_x \mathcal{G}_P(x)$, so the right side of (1) is proven. ◀

In proving Lemma 10 we have also proven the following statement.

▶ **Corollary 12.** *For any $x \in \mathbb{R}^m$, there exists some point $\tilde{g}(x) \in \mathbb{R}^d$ such that, for all $p \in P$, $\|\tilde{g}(x) - p\| \leq \|x - \Pi p\|$.*

We complete the proof of Theorem 9 by showing that, not only does the maximum of $\mathcal{G}_{\Pi P}$ approximate that of $\mathcal{G}_P$, but an approximate maximizer for $\mathcal{G}_{\Pi P}$ can be used to recover one for $\mathcal{G}_P$. Algorithm 2 does so on Line 5 by applying a single iteration of the *mean-shift* algorithm, a common heuristic KDE mode finding [9, 10], which repeatedly iterates the equation $x^{(i+1)} = \frac{\sum_{p \in P} p \cdot e^{-\|x^{(i)} - p\|^2}}{\sum_{p \in P} e^{-\|x^{(i)} - p\|^2}}$. While not guaranteed to converge to a point which maximizes $\mathcal{G}_P$, a useful property of the mean-shift algorithm is that its solution is guaranteed to never decrease in quality on each iteration:

▷ Claim 13. Given $y \in \mathbb{R}^d$, let $y' = \frac{\sum_{p \in P} p \cdot e^{-\|y - p\|^2}}{\sum_{p \in P} e^{-\|y - p\|^2}}$, then $\mathcal{G}_P(y') \geq \mathcal{G}_P(y)$.

Proof. We prove this well known fact for completeness. First, observe by rearrangement that $\mathcal{G}_P(y') - \mathcal{G}_P(y) = \sum_{p \in P} \left(e^{-\|y' - p\|^2 + \|y - p\|^2} - 1\right) e^{-\|y - p\|^2}$. Then, since $e^z \geq 1 + z$ for all z, we have $e^{-\|y' - p\|^2 + \|y - p\|^2} - 1 \geq -\|y' - p\|^2 + \|y - p\|^2 = -\left(\|y'\|^2 - \|y\|^2 - 2(y' - y)^T p\right)$.

$$\begin{aligned}\mathcal{G}_P(y') - \mathcal{G}_P(y) &\geq -\left(\|y'\|^2 - \|y\|^2\right) \sum_{p \in P} e^{-\|y - p\|^2} + 2(y' - y)^T \sum_{p \in P} p e^{-\|y - p\|^2} \\ &= \mathcal{G}_P(y)\left(-\|y'\|^2 + \|y\|^2 + 2(y' - y)^T y'\right) = \mathcal{G}_P(y)\|y' - y\|^2 \geq 0.\end{aligned}$$ ◀

Proof of Theorem 9. Recall from Corollary 12 that for any x, there is always a $\tilde{g}(x)$ with

$$\|\tilde{g}(x) - p\| \leq \|x - \Pi p\| \tag{4}$$

for all $p \in P$. Suppose this inequality was tight: i.e., suppose that for all $p \in P, x \in \mathbb{R}^m$, $\|\tilde{g}(x) - p\| = \|x - \Pi p\|$. Then letting x'' be as defined in Algorithm 2, we would have that Line 5 sets x' equal to a mean-shift update applied to $\tilde{g}(x'')$. From Claim 13 we would then immediately have that $\mathcal{G}_P(x') \geq \mathcal{G}_P(\tilde{g}(x'')) = \mathcal{G}_{\Pi P}(x'') \geq (1 - \epsilon/2) \max_x \mathcal{G}_{\Pi P}(x) \geq (1 - \epsilon) \max_x \mathcal{G}_P(x)$, which would prove the theorem.

However, since (4) is not tight, we need a more involved argument by lifting to a $d+1$-dimensional space. In particular, for each $p \in P$, let $\bar{p} \in \mathbb{R}^{d+1}$ be a vector with its first d entries equal to p and let the final entry be equal to $\sqrt{\|x - \Pi p\|^2 - \|\tilde{g}(x) - p\|^2}$. Additionally,

Algorithm 3 Full algorithm for high dimensional case.

```
input: a point set $P \in \mathbb{R}^d$, parameter $\epsilon, \rho, \delta > 0$
Generate $O(\log \frac{1}{\delta})$ random samples $P_0^j \subset P$ of size $n_0 = O(\frac{1}{\epsilon^2 \rho^2})$ (à la Lopaz-Paz et al.)
for $j \leftarrow 1$ to $O(\log \frac{1}{\delta})$ do
    Set $\gamma = \frac{\epsilon}{4 \log(4/\epsilon\rho)}$.
    Choose random matrix $\Pi \in \mathbb{R}^{m \times d}$ satisfying $(\gamma, n+1, 1/100)$-JL guarantee (Defn. 8)
    For each $p_i \in P_0^j$, compute $\Pi p_i$ and let $\Pi P_0^j$ denote the data set $\{\Pi p_1, \ldots, \Pi p_n\}$
    Run the algorithm in Phillips and Tai [39] to construct a subset $P_2^j \subset \Pi P_0^j$ of size
      $n_2 = O(\frac{\sqrt{m}}{\epsilon\rho} \sqrt{\log \frac{1}{\epsilon\rho}}) = O(\frac{1}{\epsilon^2 \rho} \log^2 \frac{1}{\epsilon\rho})$
    Set $x''$ as the output of Algorithm 1 (Section 2) on $P_2^j$ in dimension $m$
    Compute new $x' = \frac{\sum_{p \in P_0^j} p \cdot e^{-\|x'' - \Pi p\|^2}}{\sum_{p \in P_0^j} e^{-\|x'' - \Pi p\|^2}}$
Return the best solution from all iterations of Step 8, evaluated on $\bigcup_j P_0^j$
```

for every point $x \in \mathbb{R}^m$, let $\bar{\tilde{g}}(x) \in \mathbb{R}^{d+1}$ be a vector with its first d entries equal to $\tilde{g}(x) \in \mathbb{R}^d$ and final entry equal to 0. Clearly, for any $p \in P$,

$$\|\bar{\tilde{g}}(x) - \bar{p}\| = \|x - \Pi p\|. \tag{5}$$

For $z \in \mathbb{R}^{d+1}$, let $\overline{\mathcal{G}}_P(z) = \sum_{p \in P} e^{-\|z - \bar{p}\|^2}$ and let $\bar{x}' = \frac{\sum_{p \in P} \bar{p} e^{-\|x'' - \Pi p\|^2}}{\sum_{p \in P} e^{-\|x'' - \Pi p\|^2}}$. It follows from (5) and the argument above that $\overline{\mathcal{G}}_P(\bar{x}') \geq \overline{\mathcal{G}}_P(\bar{\tilde{g}}(x')) = \mathcal{G}_{\Pi P}(x'')$. But clearly it also holds that $\mathcal{G}_P(x') \geq \overline{\mathcal{G}}_P(\bar{x}')$ because, for any $p \in P$, $\|x' - p\| \leq \|\bar{x}' - \bar{p}\|$. So we conclude that $\mathcal{G}_P(x') \geq \mathcal{G}_{\Pi P}(x'')$ as desired. Furthermore, recall that x'' is an approximate mode in the projected setting. It satisfies $\mathcal{G}_{\Pi P}(x'') \geq \max_x (1 - \epsilon/2) \mathcal{G}_{\Pi P}(x)$, and from Lemma 10 we have that $\max_x \mathcal{G}_{\Pi P}(x) \geq (1 - \epsilon/2) \max_x \mathcal{G}_P(x)$. Chaining these inequalities gives the desired bound that $\mathcal{G}_P(x') \geq (1 - \epsilon/2)^2 \max_x \mathcal{G}_P(x) \geq (1 - \epsilon) \max_x \mathcal{G}_P(x)$. ◀

3.1 Final Result for High Dimensions

For the high dimensional case, we combine together the techniques of 1) dimensionality reduction, 2) polynomial system solving and 3) coresets by [34] and [39] to obtain an algorithm that is linear in the dimensionality d and exponential only in $\mathsf{poly}(1/\epsilon, \log 1/\rho)$, leading to Theorem 2.

In the regime where ϵ (the relative error) and δ (the probability of failure) are constant, the runtime simplifies to $O\left(\left(n + \frac{1}{\rho^2}\right) d + \left(\log \frac{1}{\rho}\right)^{O(\log^3 \frac{1}{\rho})}\right)$. Note however that if $1/\rho^2 \leq n_0$ dominates n, then we would not have constructed the coresets P_0^j in the first place but used the entire point set instead, and so we can treat the first term as just $O(nd)$. We also recall that $\rho = \overline{\mathcal{G}}_P(x^*) \geq 1/n$, which by substitution gives an upper bound of $O\left(nd + (\log n)^{O(\log^3 n)}\right)$.

Proof of Theorem 2. We first show the approximation guarantee. It suffices to prove that an iteration of the for loop succeeds with constant probability, so we fix a particular j and omit the superscript in P_0 and P_2. From Lemma 4, $x^* \in B_q\left(\sqrt{\log \frac{1}{\rho}}\right) \subset B_q\left(\sqrt{\log \frac{1}{\epsilon\rho}}\right)$ for some $q \in P_2$. Let x_0^{**} be $\arg\max_{x \in \mathbb{R}^m} \mathcal{G}_{P_2}(x)$. By Lemma 6 with $r = 1$, we have $\overline{\mathcal{G}}_{P_2}(x'') \geq \overline{\mathcal{G}}_{P_2}(x_0^{**}) - \epsilon\rho \geq (1 - \epsilon)\overline{\mathcal{G}}_{P_2}(x_0^{**})$. The coreset result by [39] implies that, both $\left|\overline{\mathcal{G}}_{\Pi P_0}(x'') - \overline{\mathcal{G}}_{P_2}(x'')\right| \leq \epsilon\rho$ and $\left|\overline{\mathcal{G}}_{\Pi P_0}(x^{**}) - \overline{\mathcal{G}}_{P_2}(x^{**})\right| \leq \epsilon\rho$ which implies $\overline{\mathcal{G}}_{\Pi P_0}(x'') \geq$

$(1 - O(\epsilon))\overline{\mathcal{G}}_{\Pi P_0}(x^{**})$. Now, let x_0^* be $\arg\max_{x \in \mathbb{R}^d} \mathcal{G}_{\Pi P_0}(x)$. By Theorem 9, with constant probability we have $\mathcal{G}_{P_0}(x') \geq (1 - O(\epsilon))\mathcal{G}_{P_0}(x_0^*)$. By [34], a random sample $P_0 \subset P$ of size $n_0 = O(\frac{1}{\epsilon^2 \rho^2})$ is sufficient to have the guarantee of $\left|\overline{\mathcal{G}}_P(x) - \overline{\mathcal{G}}_{P_0}(x)\right| \leq \epsilon\rho$ for any $x \in \mathbb{R}^d$. If we combine this inequality and the guarantee of random sampling, we can conclude that $\overline{\mathcal{G}}_P(x') \geq (1 - \epsilon)\overline{\mathcal{G}}_P(x^*)$.

We now analyze the running time. Reading the input and constructing the coresets P_0^j take $O(nd + n_0 \log \frac{1}{\delta})$ time in total. Evaluating all the solutions in Step 9 takes $O(n_0 d \log^2 \frac{1}{\delta})$ time, since there are $O(\log \frac{1}{\delta})$ many candidates evaluated over a coreset of size $O(n_0 \log \frac{1}{\delta})$ in d dimensions. From Theorem 9, the runtime of a single iteration of the loop is $O(n_0 dm) + T_{m,\epsilon/2}$, where $T_{m,\epsilon/2}$ is the runtime of solving the approximate mode finding problem in m dimensions. In our case, $T_{m,\epsilon/2}$ consists of the runtime of the second coreset result as well as Algorithm 1. It takes time $O(n_0 \text{poly}(1/\epsilon\rho))$ to compute the second coreset P_2^j. Then, Theorem 7 implies that Algorithm 1 requires $O(n_2 \log n_2 \cdot (2\sqrt{2e\pi})^m + n_2 \cdot \left(\log \frac{m}{\epsilon\rho}\right)^{O(m)})$.

The single-loop runtime is dominated by the runtime of Algorithm 1. Writing out the runtime of Algorithm 1 gives

$$O\left(n_2 \log n_2 \cdot (2\sqrt{2e\pi})^m + n_2 \cdot \left(\log \frac{m}{\epsilon\rho}\right)^{O(m)}\right) = O\left(n_2 \cdot \left(\log \frac{m}{\epsilon\rho}\right)^{O(m)}\right)$$
$$= O\left(\frac{1}{\epsilon^2\rho} \log^2 \frac{1}{\epsilon\rho} \cdot \left(\log \frac{1}{\epsilon^3\rho} \log \frac{1}{\epsilon\rho} \log^2 \frac{1}{\epsilon\rho}\right)^{O(\frac{1}{\epsilon^2} \log^3 \frac{1}{\epsilon\rho})}\right) = O\left(\left(\log \frac{1}{\epsilon\rho}\right)^{O(\frac{1}{\epsilon^2} \log^3 \frac{1}{\epsilon\rho})}\right).$$

Combining with the runtimes for reading the input, coreset construction and evaluating solutions in Step 9, then repeating the loop for $O(\log \frac{1}{\delta})$ times gives the bound in the theorem statement. If $n < n_0 \log \frac{1}{\delta}$, we use the full set P as each P_0^j. ◀

4 KDE mode finding for Two Dimensional Case

In this subsection, we assume that $P \subset \mathbb{R}^2$ and $p = (p_1, p_2)$ for each $p \in P$. We can improve our low-dimensional analysis that used a set of systems of polynomials by about a logarithmic factor using a different approach. This shows how to approximate each Gaussian by a weighted set of rectangles. After sampling by these weights, we can quickly retrieve the point of maximum depth in these rectangles as an approximation of the maximum.

We first define the following notation. We let $s = \frac{\epsilon\rho}{6}$ be a minimal additive error we will allow for the spatial approximation, and then $m = \lceil \frac{1}{s} \rceil$ will be the number of discretizations we will need. A Gaussian has infinite support, but we will only need to consider m such widths defined $r_j = \sqrt{\log \frac{1}{l_j}}$ with $l_j = 1 - \frac{j}{m}$ for $j = 0, 1, \ldots, m$. As a special case we set $r_m = \infty$ (note that this allows $e^{-r_j^2} = l_j$). We can now define a series of axis-parallel rectangles centered at a point $p = (p_1, p_2) \in P$ as $\mathcal{R}_p = \left\{[p_1 - r_{a_1}, p_1 + r_{a_1}] \times [p_2 - r_{a_2}, p_2 + r_{a_2}] \mid (a_1, a_2) \in \{0, 1, \ldots, m-1\}^2\right\}$. It enumerates all widths $r_0, r_1, \ldots, r_{m-1}$ on both directions, so its size is m^2. Also, let $\mathcal{R}$ be $\cup_{p \in P} \mathcal{R}_p$.

Given any $x \in \mathbb{R}^d$ and any finite collection $\mathcal{C}$ subsets of $\mathbb{R}^2$, denote $N(\mathcal{C}, x)$ as the number of $C \in \mathcal{C}$ that $x \in C$, known as the *depth* or *ply* of x. And we can show that the depth, normalized by $1/(nm^2)$, approximates the KDE value $\overline{\mathcal{G}}_P(x)$.

▶ **Lemma 14.** $\overline{\mathcal{G}}_P(x) \geq \frac{N(\mathcal{R},x)}{nm^2} \geq \overline{\mathcal{G}}_P(x) - \frac{1}{3}\epsilon\rho$

The main idea is to show that Gaussian kernel can be approximated by a collection of axis-parallel rectangle where m controls precision. Observe that $|\mathcal{R}| = nm^2$. However, $|\mathcal{R}|$ (and therefore m) does not show up in the running time of our algorithm since, we perform the random sampling on $\mathcal{R}$ in the first step of Algorithm 4.

Proof. For any $p \in P$ and $i \in \{1,2\}$, let a_i be the integer such that $r_{a_i-1} \leq |p_i - x_i| \leq r_{a_i}$ which implies $e^{-r_{a_i-1}^2} \geq e^{-(p_i-x_i)^2} \geq e^{-r_{a_i}^2} = 1 - \frac{a_i}{m}$. Then, we have

$$e^{-\|p-x\|^2} = e^{-(p_1-x_1)^2-(p_2-x_2)^2} \geq (1-\frac{a_1}{m})(1-\frac{a_2}{m}) = \frac{N(\mathcal{R}_p,x)}{m^2}$$

Note that $N(\mathcal{R},x) = \sum_{i=1}^{d} N(\mathcal{R}_p,x)$. Now,

$$\mathcal{G}_P(x) = \sum_{p\in P} e^{-\|p-x\|^2} \geq \sum_{p\in P} \frac{N(\mathcal{R}_p,x)}{m^2} = \frac{N(\mathcal{R},x)}{m^2}$$

On the other hand, let $\Delta_{p,x,i} = e^{-(p_i-x_i)^2} - (1-\frac{a_i}{m})$ which is larger than 0,

$$\begin{aligned}\frac{N(\mathcal{R}_p,x)}{m^2} &= (1-\frac{a_1}{m})(1-\frac{a_2}{m}) = \left(e^{-(p_1-x_1)^2} - \Delta_{p,x,1}\right)\left(e^{-(p_2-x_2)^2} - \Delta_{p,x,2}\right) \\ &= e^{-(p_1-x_1)^2}e^{-(p_2-x_2)^2} - \Delta_{p,x,1}e^{-(p_2-x_2)^2} - \Delta_{p,x,2}e^{-(p_1-x_1)^2} + \Delta_{p,x,1}\Delta_{p,x,2} \\ &\geq e^{-(p_1-x_1)^2}e^{-(p_2-x_2)^2} - \Delta_{p,x,1}e^{-(p_2-x_2)^2} - \Delta_{p,x,2}e^{-(p_1-x_1)^2}\end{aligned}$$

Recall that $e^{-r_{a_i-1}^2} \geq e^{-(p_i-x_i)^2} \geq e^{-r_{a_i}^2}$ which implies $\Delta_{p,x,i} \leq e^{-r_{a_i-1}^2} - e^{-r_{a_i}^2} = s$. The above equation becomes

$$\begin{aligned}\frac{N(\mathcal{R}_p,x)}{m^2} &\geq e^{-(p_1-x_1)^2}e^{-(p_2-x_2)^2} - \Delta_{p,x,1}e^{-(p_2-x_2)^2} - \Delta_{p,x,2}e^{-(p_1-x_1)^2} \\ &\geq e^{-\|p-x\|^2} - 2s\end{aligned}$$

Finally, we have

$$\frac{N(\mathcal{R},x)}{m^2} = \sum_{p\in P}\frac{N(\mathcal{R}_p,x)}{m^2} \geq \sum_{p\in P}(e^{-\|p-x\|^2} - 2s) = \mathcal{G}_P(x) - \frac{1}{3}\epsilon n\rho.$$ ◀

Now consider $(X,\mathcal{S})$ be a range space with VC dimension ν. Given $\epsilon > 0$ and $\alpha > 0$, we call a subset Z of X a relative (α,ϵ)-approximation for $(X,\mathcal{S})$ if, for any $\tau \in \mathcal{S}$, $\left|\frac{|X\cap\tau|}{|X|} - \frac{|Z\cap\tau|}{|Z|}\right| \leq \epsilon M$ when $M = \max\{\frac{|X\cap\tau|}{|X|}, \alpha\}$. A random sample of size $O\left(\frac{1}{\epsilon^2\alpha}(\nu\log\frac{1}{\alpha} + \log\frac{1}{\delta})\right)$ is an (α,ϵ)-approximation with probability at least $1-\delta$ [25]. The range space $(\mathbb{R}^2,\mathcal{B})$ where $\mathcal{B}$ is the set of all axis-parallel box in $\mathbb{R}^2$ has VC dimension 4. Thus its dual range space $(\mathcal{B},\mathcal{D})$ where $\mathcal{D} = \left\{\{B\in\mathcal{B} \mid x\in B\} \mid x\in\mathbb{R}^2\right\}$, has VC dimension is $O(1)$.

Given a set $\mathcal{B}_0$ of λ axis-aligned rectangles in $\mathbb{R}^2$, [12] finds a maximal depth point, that maximizes $N(\mathcal{B}_0,x)$, in $O(\lambda\log\lambda)$ time. This leads to Algorithm 4 and Theorem 3.

Algorithm 4 Computing Depth.

```
input: a point set $P \subset \mathbb{R}^2$, parameter $\epsilon, \rho, \delta > 0$
generate a random subset $\mathcal{R}_0$ of $\mathcal{R}$ of size $O\left(\frac{1}{\epsilon^2\rho}(\log\frac{1}{\rho} + \log\frac{1}{\delta})\right)$.
compute $x' \in \mathbb{R}^2$ such that $x' = \arg\max_{x\in\mathbb{R}^d} N(\mathcal{R}_0,x)$ using the algorithm by [12].
return $x'$
```

Proof of Theorem 3. First, by Lemma 14, $\frac{N(\mathcal{R},x^*)}{|\mathcal{R}|} \geq \overline{\mathcal{G}}_P(x^*) - \frac{1}{3}\epsilon\rho = \Omega(\rho)$. Let M be $\max\{\frac{N(\mathcal{R},x')}{|\mathcal{R}|},\rho\}$. We also have $M = \max\{\frac{N(\mathcal{R},x')}{|\mathcal{R}|},\rho\} \leq \overline{\mathcal{G}}_P(x^*)$. By Lemma 14 and the construction of $\mathcal{R}_0$, we have $\overline{\mathcal{G}}_P(x') \geq \frac{N(\mathcal{R},x')}{|\mathcal{R}|} \geq \frac{N(\mathcal{R}_0,x')}{|\mathcal{R}_0|} - \frac{1}{3}\epsilon M$. Since x' is the optimal solution, the term $\frac{N(\mathcal{R}_0,x')}{|\mathcal{R}_0|}$ is larger than $\frac{N(\mathcal{R}_0,x^*)}{|\mathcal{R}_0|}$.

$$
\begin{aligned}
\overline{\mathcal{G}}_P(x') &\geq \frac{N(\mathcal{R}_0, x^*)}{|\mathcal{R}_0|} - \frac{1}{3}\epsilon M \\
&\geq \frac{(1-\frac{1}{3}\epsilon)N(\mathcal{R}, x^*)}{|\mathcal{R}|} - \frac{1}{3}\epsilon M && \text{by } \frac{N(\mathcal{R}, x^*)}{|\mathcal{R}|} = \Omega(\rho) \text{ and construction of } \mathcal{R}_0 \\
&\geq (1-\frac{1}{3}\epsilon)(\overline{\mathcal{G}}_P(x^*) - \frac{1}{3}\epsilon\rho) - \frac{1}{3}\epsilon M && \text{by [12]}
\end{aligned}
$$

Finally, by the assumption of $\rho \leq M \leq \overline{\mathcal{G}}_P(x^*)$, we have $\overline{\mathcal{G}}_P(x') \geq (1-\epsilon)\overline{\mathcal{G}}_P(x^*)$.

To see the running time, note that the size of input $\lambda = O\left(\frac{1}{\epsilon^2\rho}(\log\frac{1}{\rho} + \log\frac{1}{\delta})\right)$ in our context and $O(n)$ time to create a sample. Therefore, the total running time is

$$
O\left(n + \frac{1}{\epsilon^2\rho}(\log\frac{1}{\rho} + \log\frac{1}{\delta})\log(\frac{1}{\epsilon\rho}\log\frac{1}{\delta})\right). \qquad ◀
$$

5 Experiments

In this section, we present two sets of experiments, demonstrating the efficacy of 1) our dimensionality reduction approach and 2) our 2D combinatorial algorithm. We did not have a ground truth, so we took the best run of any algorithm as the optimal (OPT), and present the "Error in %" as $(x - \mathsf{OPT})/\mathsf{OPT}$. The experiments were run in Python on Google Colab instances with GPU.

Dimensionality Reduction. The first experiment shows the speedup attained via dimensionality reduction, while sacrificing little in solution quality. As noted in Section 3, dimensionality reduction can be combined with any algorithm for mode finding; we compare the state-of-the-art mean-shift heuristic (described also in Section 3) with applying mean-shift after reducing the dimensionality in the data. We use a subset of the CelebA images [33]: $n = 20{,}000$ aligned and cropped face 178×218 pixel images of celebrities. We converted each image to greyscale, and treat as $(d = 38804)$-dimensional vectors.

Given the KDE, we pick 10 random starting points and run mean-shift starting at each of them until the KDE value improves by less than 0.001. Then we return to the original dimensionality by running a single iteration of mean shift, to get a final value. We output the best solution, and report the total time of all restarts. For each target dimension (20–800), we report 500 trials as separate marks in the plot. Each trial with reduced dimensionality uses a single JL matrix across all restarts.

Figure 1 (Left) shows that, even if we reduce from 38804 dimensions down to 20 dimensions, the solution quality loss is only in the order of 0.1%. For reference, the solution quality is roughly 6550. The runtime savings are significant, from roughly 130 seconds in the original 38804 dimensions, to 8-9 seconds in 20 dimensions.

The theory demands a JL matrix with one-sided error; the random Gaussian matrix should be divided by some factor of $1 - \epsilon$. We did not do so because: (1) for very low target dimensions (say, 20), there is no valid $\epsilon \in (0, 1)$; and (2) even when such ϵ exists, this ϵ is large enough that a division by $1 - \epsilon$ introduces significant bias and worsens the solution.

2D Algorithm. Figure 1 (Right) shows the comparison of our 2D combinatorial algorithm and heuristics for mode finding. It shows both the best heuristic from [54] of evaluating random points, and then also the mean-shift iterative improvement on top of these [9, 10].

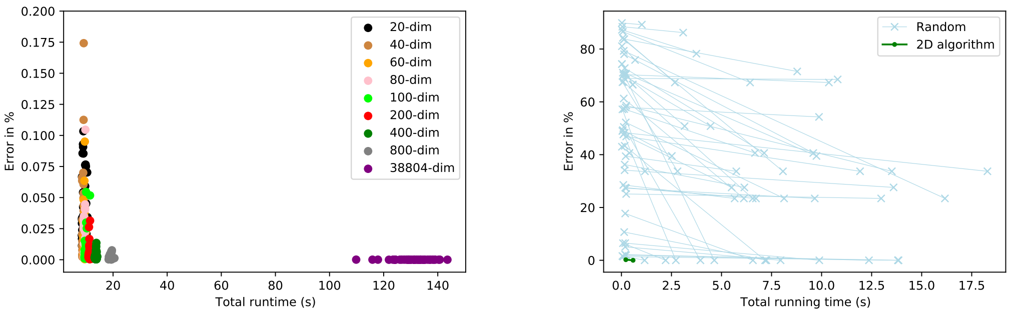

Figure 1 Percentage Error of algorithms as a function of runtime. (Left): Scatterplot for the dimensionality reduction, then mean-shift with 10 restarts. For each target dimension, we show 500 trials. (Right): 2D experiments, our algorithm versus choosing random starting point, then mean-shift. In each segment top point is error before mean-shift, and bottom one is after mean-shift.

We use the entire "Iowa_highway" dataset in [54], which has $n = 1{,}155{,}101$ points in $\mathbb{R}^2$ denoting all waypoints in Iowa from Open Street Maps. It is very multi-modal.

To compare our algorithm, we start with the best heuristic [54] of evaluating the KDE at k data points, and selecting the best. We use k between 1 and 10 random data points, and repeat 5 times for each, and select the lowest error. These are the top blue xs of each segment in the figure. Then for each initial data point, we run mean-shift to improve the error, and report the lowest error (out of each set of k starting points). These are the lower blue x of each segment in the figure. Note that the initial data point sampling heuristic occasionally obtains near-optimal error, but is typically much worse. The blue line segments showing the improvement of mean-shift indicate again it sometimes obtains near-optimal error, but not consistently. It also takes several seconds.

Our algorithm is shown as green dots. The top dot of the green segment is the cost/error of our algorithm, and the lower one is after optimizing with mean shift. We observe that our algorithm is significantly more efficient than the heuristics, taking less than one second, and achieves near-optimality, basically the same error as the best of prior heuristics.

References

1 Dimitris Achlioptas. Database-friendly random projections: Johnson-Lindenstrauss with binary coins. *J. Comput. Syst. Sci.*, 66(4):671–687, 2003.

2 Pankaj K Agarwal, Haim Kaplan, and Micha Sharir. Union of random minkowski sums and network vulnerability analysis. In *Proceedings of the twenty-ninth annual symposium on Computational geometry*, pages 177–186. ACM, 2013.

3 Nir Ailon and Bernard Chazelle. The fast johnson-lindenstrauss transform and approximate nearest neighbors. *SIAM J. Comput.*, pages 302–322, 2009.

4 Nir Ailon and Edo Liberty. An almost optimal unrestricted fast johnson-lindenstrauss transform. *ACM Trans. Algorithms*, 9(3):21:1–21:12, 2013.

5 Carlos Améndola, Alexander Engström, and Christian Haase. Maximum number of modes of gaussian mixtures. *Information and Inference: A Journal of the IMA*, 9(3):587–600, 2020.

6 Arturs Backurs, Moses Charikar, Piotr Indyk, and Paris Siminelakis. Efficient density evaluation for smooth kernels. In *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 615–626. IEEE, 2018.

7 Arturs Backurs, Piotr Indyk, and Tal Wagner. Space and time efficient kernel density estimation in high dimensions. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, pages 15799–15808, 2019.

8 Luca Becchetti, Marc Bury, Vincent Cohen-Addad, Fabrizio Grandoni, and Chris Schwiegelshohn. Oblivious dimension reduction for k-means: beyond subspaces and the Johnson-Lindenstrauss lemma. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, pages 1039–1050, 2019.

9 Miguel Á. Carreira-Perpiñán. Mode-finding for mixtures of gaussian distributions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(11):1318–1323, 2000.

10 Miguel Á. Carreira-Perpiñán. Gaussian mean-shift is an em algorithm. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(5):767–776, 2007.

11 Miguel A Carreira-Perpinán and Christopher KI Williams. On the number of modes of a gaussian mixture. In *International Conference on Scale-Space Theories in Computer Vision*, pages 625–640. Springer, 2003.

12 Timothy M Chan. Klee's measure problem made easy. In *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*, pages 410–419. IEEE, 2013.

13 Cheng Chang and R. Ansari. Kernel particle filter for visual tracking. *IEEE Signal Processing Letters*, 12:242–245, 2005.

14 Frédéric Chazal, Brittany Terese Fasy, Fabrizio Lecci, Bertrand Michel, Alessandro Rinaldo, and Larry Wasserman. Robust topolical inference: Distance-to-a-measure and kernel distance. Technical report, arXiv:1412.7197, 2014.

15 Michael Cohen, Sam Elder, Cameron Musco, Christopher Musco, and Madalina Persu. Dimensionality reduction for k-means clustering and low rank approximation. In *In Proceedings of the 47th Annual ACM Symposium on Theory of Computing (STOC)*, pages 163–172, 2015.

16 Michael B. Cohen, T.S. Jayram, and Jelani Nelson. Simple analyses of the sparse johnson-lindenstrauss transform. In *The 1st Symposium on Simplicity in Algorithms*, pages 15:1–15:9, 2018.

17 Sanjoy Dasgupta and Anupam Gupta. An elementary proof of a theorem of johnson and lindenstrauss. *Random Structures & Algorithms*, 22(1):60–65, 2003.

18 Luc Devroye and László Györfi. *Nonparametric Density Estimation: The L_1 View*. Wiley, 1984.

19 Luc Devroye and Gábor Lugosi. *Combinatorial Methods in Density Estimation*. Springer-Verlag, 2001.

20 Herbert Edelsbrunner, Brittany Terese Fasy, and Günter Rote. Add isotropic Gaussian kernels at own risk: More and more resiliant modes in higher dimensions. *Proceedings 28th Annual Symposium on Computational Geometry*, pages 91–100, 2012.

21 Kenji Fukumizu, Le Song, and Arthur Gretton. Kernel bayes' rule: Bayesian inference with positive definite kernels. *Journal of Machine Learning Research*, 2013.

22 Theo Gasser, Peter Hall, and Brettt Presnell. Nonparametric estimation of the mode of a distribution of random curves. *Journal of the Royal Statistical Society: Series B*, 60:681–691, 1997.

23 Arthur Gretton, Karsten M. Borgwardt, Malte J. Rasch, Bernhard Schölkopf, and Alexander Smola. A kernel two-sample test. *Journal of Machine Learning Research*, 13:723–773, 2012.

24 Mingxuan Han, Michael Matheny, and Jeff M. Phillips. The kernel spatial scan statistic. In *ACM International Conference on Advances in Geographic Information Systems*, 2019.

25 Sariel Har-Peled and Micha Sharir. Relative (p, ε)-approximations in geometry. *Discrete & Computational Geometry*, 45(3):462–496, 2011.

26 Chi Jin, Yuchen Zhang, Sivaraman Balakrishnan, Martin J. Wainwright, and Michael Jordan. Local maxima in the likelihood of gaussian mixture models: Structural results and algorithmic consequences. In *NeurIPS*, 2016.

27 Geoge H. John and Pat Langley. Estimating continuous distributions in bayesian classifiers. In *Proceedings of the Eleventh conference on Uncertainty in artificial intelligence*, 1995.

28 Sarang Joshi, Raj Varma Kommaraji, Jeff M Phillips, and Suresh Venkatasubramanian. Comparing distributions and shapes using the kernel distance. In *Proceedings of the twenty-seventh annual symposium on Computational geometry*, pages 47–56. ACM, 2011.

29 Daniel M. Kane and Jelani Nelson. Sparser Johnson-Lindenstrauss transforms. *Journal of the ACM*, 61(1):4, 2014.

30 M. Kirszbraun. Über die zusammenziehende und lipschitzsche transformationen. *Fundamenta Mathematicae*, 22(1):77–108, 1934.

31 Felix Krahmer and Rachel Ward. New and improved johnson-lindenstrauss embeddings via the restricted isometry property. *SIAM Journal on Mathematical Analysis*, 43(3):1269–1281, 2011.

32 Yi Li, Philip M. Long, and Aravind Srinivasan. Improved bounds on the samples complexity of learning. *Journal of Computer and System Science*, 62:516–527, 2001.

33 Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

34 David Lopaz-Paz, Krikamol Muandet, Bernhard Schölkopf, and Ilya Tolstikhin. Towards a learning theory of cause-effect inference. In *International Conference on Machine Learning*, 2015.

35 Konstantin Makarychev, Yury Makarychev, and Ilya Razenshteyn. Performance of Johnson-Lindenstrauss transform for k-means and k-medians clustering. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing (STOC)*, pages 1027–1038, 2019.

36 Krikamol Muandet, Kenji Fukumizu, Bharath Sriperumbudur, and Bernhard Schölkopf. Kernel mean embedding of distributions: A review and beyond. *Foundations and Trends in Machine Learning*, 10:1–141, 2017.

37 Shyam Narayanan and Jelani Nelson. Optimal terminal dimensionality reduction in euclidean space. In *Proceedings of the 51st Annual ACM Symposium on Theory of Computing (STOC)*, pages 1064–1069, 2019.

38 Jeff M Phillips and Wai Ming Tai. Improved coresets for kernel density estimates. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 2718–2727. SIAM, 2018.

39 Jeff M Phillips and Wai Ming Tai. Near-optimal coresets of kernel density estimates. In *34th International Symposium on Computational Geometry (SoCG 2018)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2018.

40 Jeff M. Phillips, Bei Wang, and Yan Zheng. Geometric inference on kernel density estimates. In *International Symposium on Computational Geometry*, 2015.

41 Kent Quanrud. Spectral sparsification of metrics and kernels. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1445–1464. SIAM, 2021.

42 James Renegar. On the computational complexity of approximating solutions for real algebraic formulae. *SIAM Journal on Computing*, 21(6):1008–1025, 1992.

43 Alessandro Rinaldo, Larry Wasserman, et al. Generalized density clustering. *The Annals of Statistics*, 38(5):2678–2722, 2010.

44 Bernhard Schölkopf and Alexander J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond.* MIT Press, 2002.

45 David W. Scott. *Multivariate Density Estimation: Theory, Practice, and Visualization.* Wiley, 1992.

46 Chunhua Shen, Michael J. Brooks, and Anton van den Hengel. Fast global kernel density mode seeking: Applications to localization and tracking. *IEEE Transactions on Image Processing*, 16:1457–1469, 2007.

47 Bernard W. Silverman. Using kernel density estimates to investigate multimodality. *Journal of the Royal Statistical Society: Series B*, 43:97–99, 1981.

48 Bernard W. Silverman. *Density Estimation for Statistics and Data Analysis.* Chapman & Hall/CRC, 1986.

49 Alex J. Smola, Arthur Gretton, Le Song, and Bernhard Schölkopf. A Hilbert space embedding for distributions. In *Proceedings of Algorithmic Learning Theory*, 2007.

50 Bharath K. Sriperumbudur, Arthur Gretton, Kenji Fukumizu, Bernhard Schölkopf, and Gert R. G. Lanckriet. Hilbert space embeddings and metrics on probability measures. *Journal of Machine Learning Research*, 11:1517–1561, 2010.

51 F. A. Valentine. A lipschitz condition preserving extension for a vector function. *American Journal of Mathematics*, 67(1):83–93, 1945.

52 Vladimir Vapnik and Alexey Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theo. of Prob and App*, 16:264–280, 1971.

53 Grace Wahba. Support vector machines, reproducing kernel Hilbert spaces, and randomization GACV. In *Advances in Kernel Methods – Support Vector Learning*, pages 69–88. Bernhard Schölkopf and Alezander J. Smola and Christopher J. C. Burges and Rosanna Soentpiet, 1999.

54 Yan Zheng and Jeff M Phillips. L_infty error and bandwidth selection for kernel density estimates of large data. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1533–1542. ACM, 2015.

55 Yan Zheng and Jeff M Phillips. Coresets for kernel regression. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 645–654. ACM, 2017.

56 Shaofeng Zou, Yingbin Liang, H Vincent Poor, and Xinghua Shi. Unsupervised nonparametric anomaly detection: A kernel method. In *2014 52nd Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 836–841. IEEE, 2014.