Applications of Chebyshev Polynomials to Low-Dimensional Computational Geometry*

Timothy M. Chan[1]

1 **Department of Computer Science, University of Illinois at Urbana-Champaign**
tmc@illinois.edu

Abstract

We apply the *polynomial method*—specifically, Chebyshev polynomials—to obtain a number of new results on geometric approximation algorithms in low constant dimensions. For example, we give an algorithm for constructing ε-*kernels* (coresets for approximate width and approximate convex hull) in close to optimal time $O(n + (1/\varepsilon)^{(d-1)/2})$, up to a small near-$(1/\varepsilon)^{3/2}$ factor, for any d-dimensional n-point set. We obtain an improved data structure for Euclidean *approximate nearest neighbor search* with close to $O(n \log n + (1/\varepsilon)^{d/4} n)$ preprocessing time and $O((1/\varepsilon)^{d/4} \log n)$ query time. We obtain improved approximation algorithms for *discrete Voronoi diagrams*, *diameter*, and *bichromatic closest pair* in the L_s-metric for any even integer constant $s \geq 2$. The techniques are general and may have further applications.

1998 ACM Subject Classification F.2.2 Nonnumerical Algorithms and Problems

Keywords and phrases diameter, coresets, approximate nearest neighbor search, the polynomial method, streaming

Digital Object Identifier 10.4230/LIPIcs.SoCG.2017.26

1 Introduction

This paper presents new results on a number of fundamental problems in low-dimensional geometric approximation algorithms. Let P be a set of n points in d-dimensional Euclidean space where d is a constant. Let $\varepsilon > 0$ be a user-specified parameter (not necessarily a constant). As a shorthand, let $E := \lceil 1/\varepsilon \rceil$. Below, the O notation may hide factor that depends on d but not ε. The notation O^* will be used to suppress small factors of the form E^c for some constant c independent of d.

Diameter. We present a new algorithm to compute a $(1+\varepsilon)$-approximation of the *diameter* of P (the farthest pair distance) in $O^*(n + E^{d/2})$ time.

There have been a long series of prior results:

$O^*(E^{d/2} n)$ time	by Agarwal, Matoušek, and Suri [3] ('91);
$O^*(n + E^{2d})$	by Barequet and Har-Peled [14] (SODA'99);
$O^*(n + E^{3d/2})$	by combining the two algorithms [19];
$O^*(n + E^d)$	by Chan [19] (SoCG'00);
$O^*(n + E^{d/2}\sqrt{n})$	by Arya and Chan [6] (SoCG'14).

Our new result is a substantial improvement, and provides a near $E^{d/4}$-factor speedup in the case when n is near $E^{d/2}$, for example.

* This work was done while the author was at the Cheriton School of Computer Science, University of Waterloo.

33rd International Symposium on Computational Geometry (SoCG 2017).
Editors: Boris Aronov and Matthew J. Katz; Article No. 26; pp. 26:1–26:15

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

ε-Kernels. We obtain an algorithm to compute an *ε-kernel* of P with worst-case optimal size $O(E^{(d-1)/2})$ in $O^*(n + E^{d/2})$ expected time; ε-kernels [1] provide coresets for a variety of problems such as diameter, width, minimum enclosing cylinder, minimum bounding box, and convex hull.

This is again a substantial improvement in terms of ε-dependencies over prior results:

$O^*(n + E^{3d/2})$ time	by Agarwal, Har-Peled, and Varadarajan [1] (SODA'01/FOCS'01);
$O^*(n + E^d)$	by Chan [19] (SoCG'00);
$O^*(n + E^{d/2}\sqrt{n})$	by Arya and Chan [6] (SoCG'14).

More importantly, since the size of the ε-kernel may be $\Omega(E^{(d-1)/2})$, our result for this problem is near worst-case optimal, up to an $O^*(1)$ factor (more precisely, an $O(E^{3/2}\log^{O(1)} E)$ factor).

Bichromatic closest pair. Assuming each input point is colored red or blue, we present a new algorithm to compute a $(1+\varepsilon)$-approximation of the *bichromatic closest pair* of P in $O^*(E^{d/4}n)$ expected time.

This improves a series of prior results:

$O^*(E^d n \log n)$ time	by Arya et al. [13] (SODA'04);
$O^*(E^{d/2} n \log n)$	by Chan [18] (SoCG'97);
$O^*(E^{d/3} n)$	by Arya and Chan [6] (SoCG'14).

Approximate nearest neighbors. More generally, we can preprocess P in $O(n \log n) + O^*(E^{d/4}n)$ expected time so that $(1+\varepsilon)$-factor *approximate nearest neighbor* queries can be answered in $O^*(E^{d/4} \log n)$ query time.

This improves prior results with:

$O(n \log n)$ preprocessing time &	$O^*(E^d \log n)$ query time	by Arya et al. [13] (SODA'04);
$O^*(E^{d/2} n \log n)$	$O^*(E^{d/2} \log n)$	by Chan [18] (SoCG'97);
$O(n \log n) + O^*(E^{d/3} n)$	$O^*(E^{d/3} \log n)$	by Arya and Chan [6] (SoCG'14).

There were more previous results by Arya et al. [12, 7, 11] giving space/query-time tradeoffs. For example, one method already achieved $O^*(E^{d/4}n)$ space and $O^*(E^{d/4} \log n)$ query time, but preprocessing time has large ε-dependencies, making the method unsuitable for bichromatic closest pair, for example.

Streaming diameter. In the insertion-only streaming model, we can maintain a $(1+\varepsilon)$-approximation of the diameter with $O(E^{(d-1)/2})$ space and $O^*(E^{d/4})$ time per insertion of point.

This improves prior results with:

$O(E^{(d-1)/2})$ space &	$O^*(E^{d/2})$ time	(folklore);
$O(E^{(d-1)/2})$	$O^*(E^{d/3})$	by Arya and Chan [6] (SoCG'14).

Streaming ε-kernels. In the insertion-only streaming model, we can maintain an ε-kernel of $O(E^{(d-1)/2})$ size with $O(E^{(d-1)/2})$ space and $O^*(1)$ time per insertion of point.

This improves prior results with:

$O(E^{(d-1)/2})$ space &	$O^*(E^d \log^d n)$ time	by Agarwal, Har-Peled, and Varadarajan [1] ('01);
$O^*(E^d)$	$O(1)$	by Chan [20] (SoCG'04);
$O(E^{(d-1)/2})$	$O^*(E^{d/2})$	by Zarrabi-Zadeh [31] (ESA'08);
$O(E^{(d-1)/2})$	$O^*(E^{d/4})$	by Arya and Chan [6] (SoCG'14).

Since our result has $O^*(1)$ time, it is near optimal (up to an $O(E^{3/2}\log^{O(1)} E)$ factor).

Discrete upper envelopes and discrete Voronoi diagrams. Most of the above results are obtained by solving the following key subproblem of independent interest, called *discrete upper envelope* (introduced in [20]): the problem is to find extreme points along various uniformly spaced directions, or more precisely, find the point $p_\xi \in P$ that maximizes $p_\xi \cdot \xi$ for each $\xi \in \Xi \times \{1\}$, where Ξ is the set of all points in a uniform grid of side length δ over $[0,1]^{d-1}$. To explain the name, note that after dualization the problem corresponds to evaluating the upper envelope of a set of hyperplanes (i.e., pointwise maximum of $(d-1)$-variate linear functions) at the vertical lines through all grid points in Ξ.

In the related $(d-1)$-dimensional *discrete Voronoi diagram* problem [20] (also called the "Euclidean distance transform" [17, 27]), we want to find the nearest neighbor p'_ξ in P to each grid point $\xi \in \Xi \times \{0\}$.

In both problems, we allow approximation with an additive error of $O(\varepsilon)$, given that $P \subset [0,1]^d$. We present an algorithm with $O^*(n + E^{d/2} + F^d)$ time where $F := \lceil 1/\delta \rceil$. Since the output size is $\Theta^*(F^d)$, our algorithm is near-optimal up to $O^*(1)$ factors if $F \geq \sqrt{E}$ (indeed, in applications, the main case of interest is when $F = \sqrt{E}$). This improves prior results (assuming $F \leq E$) with:

$O^*(F^d n)$ time	(trivial);
$O^*(n + E^d)$	by Chan [20] (SoCG'04);
$O^*(\min_{k=0}^{d} F^{d-k}(n + E^k))$	by Arya and Chan [6] (SoCG'14).

The last bound is $O^*(n + E^{d/2}\sqrt{n})$ in the case $F = \sqrt{E}$. Interestingly, these prior algorithms actually solve the discrete upper envelope problem *exactly* after an initial rounding of P to a uniform grid of side length ε (in other words, error solely comes from rounding). Our new approach will make more powerful use of approximation.

Significance. For specific small constants d, our improvements may not be dramatic when factors hidden in the O^* notation are taken into account. On the other hand, for large constants d, the time bound may become impractically large as E grows (not to mention that hidden constant factors, of the form $d^{O(d)}$, may become an issue). For example, for diameter with $d = 15$ and $n = E^7$, the old bound [20] was $O(E^{13} \log E)$, the most recent previous bound [6] was $O(E^{9.5} \log E)$, and the new bound is $O(E^8 \log^{O(1)} E)$. For bichromatic closest pair with $d = 25$, the old bound [18] was $O(E^{12} n \log n)$, the most recent previous bound [6] was $O(E^{7.5} n \log E)$, and the new bound is $O(E^{5.75} n \log^{O(1)} E)$. Also, for a problem such as diameter, there are existing alternatives that do not necessarily have good worst-case running time but performs much better on "realistic input" [24].

However, we believe that more significant than the results are the techniques. Surprisingly, we bring in algebraic techniques—namely, the *polynomial method*—to tackle computational geometry problems that have traditionally been solved using geometric techniques only. Specifically, our algorithms use *Chebyshev polynomials*.

The polynomial method and Chebyshev polynomials have found applications before in theoretical computer science, numerical analysis, and other areas. Two recent lines of research are particularly relevant:

- Andoni and Nguyen [5] (SODA'12) applied the polynomial method to obtain *dynamic streaming* algorithms for the width problem. This was followed up by Chan [21] (SoCG'16) for the ε-kernel problem. Chebyshev polynomials were not used. In regards to traditional (or insertion-only streaming) algorithms, these ideas seem to give poorer results than what is already known.

- Valiant [28] (FOCS'12) applied Chebyshev polynomials to obtain faster algorithms for approximate bichromatic closest pair and offline approximate nearest neighbor search *in high dimensions*. This was later improved by Alman, Chan, and Williams [4] (FOCS'16). These ideas do not seem directly useful for low-dimensional nearest neighbor search, as the target time bounds are vastly different in low vs. high dimensions.

Interestingly, our work will combine ideas from these two research threads, along with existing geometric techniques on low-dimensional ε-kernels and approximate nearest neighbor search. The connection may be simple in hindsight (none of our ideas are original in isolation), but honestly the author did not anticipate that these threads could come together so neatly!

Although we draw on algebraic techniques, our algorithms for discrete upper envelopes and diameter are easy to understand and do not require advanced background. Our first algorithm does not even need fast Fourier transform or fast matrix multiplication, just simple arithmetic on roughly $\sqrt{E}$-bit-long numbers (our time bounds already account for the bit complexity of such operations). Section 2 giving a self-contained description of the first algorithm is about two pages long. Our second algorithm for diameter, which uses fast Fourier transform, as described in Section 4, is even shorter.

Note. After completing a preliminary draft of this paper, the author has learned that Arya, de Fonseca, and Mount (personal communication, late Nov. 2016) have independently obtained similar results [10]. In fact, their time bounds are a little better in the hidden $O^*(1)$ factors (for example, for diameter, we obtain $O((n\sqrt{E} + E^{(d+1)/2}) \log^{O(1)} E)$ time, whereas they obtain $O(n \log E + E^{(d-1)/2+\delta})$ time for an arbitrarily small constant $\delta > 0$). The fact that the techniques are completely different makes the independent discovery all the more exciting. Arya et al.'s techniques build on a long series of their earlier work involving *Macbeath regions* [7, 8, 9, 11], and the analysis in these previous papers appears complicated. In contrast, our algorithms make minimal use of geometry, and are more general in some sense. For instance, our second algorithm for diameter works in the L_s metric for any integer constant $s \geq 2$ (or other similarly behaved distance functions) with the same running time, whereas the approach by Arya et al. does not appear to generalize because of its reliance on a certain lifting transformation. The polynomial method is very powerful, and we anticipate more applications will follow.

2 First Algorithm

Our first algorithm solves a generalization of the discrete upper envelope problem:

▶ **Problem 1.** (Generalized Discrete Upper Envelope) *Let d be a constant and let $\psi_1, \ldots, \psi_{d-1}$ be bivariate $O(1)$-degree polynomials with integer coefficients. Given a set P of n points in $\mathbb{Z}^d$, we want to compute*

$$f(x_1, \ldots, x_{d-1}) := \max_{(a_1,\ldots,a_d)\in P} (\psi_1(a_1, x_1) + \cdots + \psi_{d-1}(a_{d-1}, x_{d-1}) + a_d) \tag{1}$$

for all[1] $(x_1, \ldots, x_{d-1}) \in [F]^{d-1}$*, while allowing additive error $O(\varepsilon U)$, where U is a given upper bound on $|\psi_1(a_1, x_1) + \cdots + \psi_{d-1}(a_{d-1}, x_{d-1}) + a_d|$ over all $(a_1, \ldots, a_d) \in P$ and $(x_1, \ldots, x_{d-1}) \in [F]^{d-1}$.*

[1] $[m]$ denotes the integer set $\{0, 1, \ldots, m-1\}$.

For example, for the discrete upper envelope problem as defined in the Introduction, we can round the given point set $P \subset [0,1]^d$ to a uniform grid of side length ε and then rescale so that $\Xi = [F]^{d-1}$ and $P \subset [E]^{d-1} \times [EF]$. We then get an instance of Problem 1 with $\psi_i(a_i, x_i) = a_i x_i$. Here, $U = O(EF)$, and $n \le (EF)^{O(1)}$ after removing duplicates.

For the $(d-1)$-dimensional discrete Voronoi diagram problem as defined in the Introduction, approximating the distance with additive error $O(\varepsilon)$ is equivalent to approximating the squared distance with additive error $O(\varepsilon)$. We can round and rescale so that $\Xi = [F]^{d-1}$ and $P \subset [E]^d$. We then get an instance of Problem 1 with $\psi_i(a_i, x_i) = (\frac{E}{F} x_i - a_i)^2$ (assuming that F divides E). Here, $U = O(E^2)$, and $n \le E^{O(1)}$ after removing duplicates. We can also take $\psi_i(a_i, x_i) = (\frac{E}{F} x_i - a_i)^s$ for the analogous problem under the L_s metric for any even integer constant s.

We now solve Problem 1 using the polynomial method. We start with basic properties about Chebyshev polynomials (e.g., see [28, 4] for quick proofs):

▶ **Lemma 1.** *Let*

$$T_q(x) := \sum_{i=0}^{\lfloor q/2 \rfloor} \binom{q}{2i} (x^2-1)^i x^{q-2i}$$

be the degree-q Chebyshev polynomial (of the first kind).

(i) *If* $|x| \le 1$, *then* $|T_q(x)| \le 1$.
(ii) *If* $x > 1$, *then* $T_q(x) > 1$.
(iii) *If* $x \ge 1 + \varepsilon$, *then* $T_q(x) > \frac{1}{2} e^{q\sqrt{\varepsilon}}$.

Set $q := \left\lceil \sqrt{E} \ln(4n) \right\rceil$ and $D := U^q$. Let $T(x) := D \cdot T_q(1 + \frac{x}{U})$, which is a polynomial with integer coefficients. Our main idea is to work with the following function instead of f:

$$\widetilde{f}(x_1, \ldots, x_{d-1}, t) := \sum_{(a_1,\ldots,a_d) \in P} T(\psi_1(a_1, x_1) + \cdots + \psi_{d-1}(a_{d-1}, x_{d-1}) + a_d - t) \qquad (2)$$

where $x_1, \ldots, x_{d-1} \in [F]$ and $|t| \in [U]$. The function $\widetilde{f}$ is "nicer" since it is a sum (instead of a max) of polynomials, and is thus itself a polynomial. The following observations explain the relationship between the two functions:

- Case 1: $f(x_1, \ldots, x_{d-1}) \le t$. Then $-2U \le \psi_1(a_1, x_1) + \cdots + \psi_{d-1}(a_{d-1}, x_{d-1}) + a_d - t \le 0$ for all $(a_1, \ldots, a_d) \in P$. By Lemma 1(i), all n terms in the sum (2) are at most D, and so $\widetilde{f}(x_1, \ldots, x_{d-1}, t) \le Dn$.

- Case 2: $f(x_1, \ldots, x_{d-1}) \ge t + \varepsilon U$. Then $\psi_1(a_1, x_1) + \cdots + \psi_{d-1}(a_{d-1}, x_{d-1}) + a_d - t \ge \varepsilon U$ for at least one $(a_1, \ldots, a_d) \in P$. By Lemma 1(iii), at least one term in the sum (2) exceeds $D \cdot \frac{1}{2} e^{q\sqrt{\varepsilon}}$. By Lemma 1(i,ii), all other terms are at least $-D$. So, $\widetilde{f}(x_1, \ldots, x_{d-1}, t) \ge D(\frac{1}{2} e^{q\sqrt{\varepsilon}} - (n-1)) > Dn$ by our choice of q.

Thus, we can approximately compare $f(x_1, \ldots, x_{d-1})$ against t with additive error εU, by evaluating $\widetilde{f}(x_1, \ldots, x_{d-1}, t)$. So, we can approximately compute $f(x_1, \ldots, x_{d-1})$ with additive error $O(\varepsilon U)$ by binary search, using $O(\log E)$ evaluations of $\widetilde{f}$. The total number of evaluations over all $(x_1, \ldots, x_{d-1}) \in [F]^{d-1}$ is $O(F^{d-1} \log E)$.

To evaluate $\widetilde{f}$, one could expand the expression into monomials, as $\widetilde{f}$ is a low-degree multivariate polynomial, but this approach seems too costly. Instead, we use the Chinese remainder theorem. In the stated domain, $\widetilde{f}$ is upper-bounded by $M := DnT_q(3) \le Dn2^{O(q)} \le 2^{O(\sqrt{E} \log^{O(1)}(nEU))}$. Let $\mathcal{P}$ be a set of primes whose product exceeds M; by

known bounds, we can choose such a set so that $|\mathcal{P}| = O(\log M / \log\log M)$ and each prime in $\mathcal{P}$ is at most $O(\log M)$. We describe how to evaluate $\widetilde{f}(x_1, \ldots, x_{d-1}, t) \bmod p$ for each $p \in \mathcal{P}$. Afterwards, we can reconstruct each value $\widetilde{f}(x_1, \ldots, x_{d-1}, t)$ by the Chinese remainder theorem, which takes at most $O(\log^2 M)$ time by elementary methods (for example, by repeated application of Euclid's algorithm, although faster, more sophisticated methods are possible). The total time of this step is $O(F^{d-1} \log E \log^2 M) = O(F^{d-1} E \log^{O(1)}(nEU))$.

Fix a prime $p \in \mathcal{P}$. For each $a_1, \ldots, a_d \in [p]$, let $w_{a_1,\ldots,a_d}$ be the number of points $(a'_1, \ldots, a'_d) \in P$ such that $a'_1 \equiv a_1,\ \ldots,\ a'_d \equiv a_d \pmod p$; we can precompute all these counts by a linear scan over P, using $O(n)$ arithmetic operations. Now,

$$\widetilde{f}(x_1, \ldots, x_{d-1}, t) \equiv \sum_{a_1,\ldots,a_d \in [p]} w_{a_1,\ldots,a_d} T(\psi_1(a_1, x_1) + \cdots + \psi_{d-1}(a_{d-1}, x_{d-1}) + a_d - t) \pmod p.$$

To generate all $\widetilde{f} \bmod p$ values, we use dynamic programming. For each $i \in \{1, \ldots, d\}$ and each $a_1, \ldots, a_{i-1}, x_i, \ldots, x_{d-1}, t \in [p]$, define

$$g^{(i)}_{a_1,\ldots,a_{i-1}}(x_i, \ldots, x_{d-1}, t) := \sum_{a_i,\ldots,a_d \in [p]} w_{a_1,\ldots,a_d} T(\psi_i(a_i, x_i) + \cdots + \psi_{d-1}(a_{d-1}, x_{d-1}) + a_d - t) \pmod p.$$

Then $g^{(1)}$ gives us $\widetilde{f} \bmod p$.

For the base case, we can compute $g^{(d)}$ using the formula

$$g^{(d)}_{a_1,\ldots,a_{d-1}}(t) \equiv \sum_{a_d \in [p]} w_{a_1,\ldots,a_d} T(a_d - t) \pmod p \tag{3}$$

for all $a_1, \ldots, a_{d-1}, t \in [p]$. For $i = d-1, \ldots, 1$, we can compute $g^{(i)}$ using the recursive formula

$$g^{(i)}_{a_1,\ldots,a_{i-1}}(x_i, \ldots, x_{d-1}, t) \equiv \sum_{a_i \in [p]} g^{(i+1)}_{a_1,\ldots,a_i}(x_{i+1}, \ldots, x_{d-1}, t - \psi_i(a_i, x_i)) \pmod p \tag{4}$$

for all $a_1, \ldots, a_{i-1}, x_i, \ldots, x_{d-1}, t \in [p]$.

The resulting dynamic program requires $O(p^d)$ table entries, each computed using $O(p)$ arithmetic operations by (3) and (4). This assumes that we have precomputed $T(x)$ for all $x \in [p]$ (which straightforwardly requires $O(pq)$ arithmetic operations). All arithmetic operations are done modulo p, each costing at most $O(\log^2 p)$ by elementary methods. Thus, the running time of the dynamic program is $O(p^{d+1} \log^2 p) = O(\log^{d+1} M \log^2 \log M) = O(E^{(d+1)/2} \log^{O(1)}(nEU))$.

Including the cost of computing the counts, the running time is $O((n + E^{(d+1)/2}) \log^{O(1)}(nEU))$ for each fixed prime p. The total over all $O(\log M / \log\log M) = O(\sqrt{E} \log^{O(1)}(nEU))$ primes $p \in \mathcal{P}$ becomes $O((n\sqrt{E} + E^{d/2+1}) \log^{O(1)}(nEU))$.

▶ **Theorem 2.** *Problem 1 can be solved in $O((n\sqrt{E} + E^{d/2+1} + F^{d-1}E) \log^{O(1)}(nEU))$ time, where $E = \lceil 1/\varepsilon \rceil$.*

In the case $F = \sqrt{E}$, the bound is $O((n\sqrt{E} + E^{d/2+1}) \log^{O(1)}(nEU))$, which nearly matches the lower bound $\Omega(n + E^{(d-1)/2})$ up to a factor of about $E^{3/2}$.

Note that the above method actually solves a data structure version of Problem 1: after preprocessing in $O((n\sqrt{E} + E^{d/2+1}) \log^{O(1)}(nEU))$ time, we can approximate $f(x_1, \ldots, x_{d-1})$ for any query point $(x_1, \ldots, x_{d-1}) \in [F]^{d-1}$ in $O(E \log^{O(1)}(nEU))$ time. This data structure problem is similar to the approximate polytope membership problem studied by Arya et al. [7, 8, 9, 11, 10].

Appendix B describes one small improvement of the $E^{d/2+1}$ term to a bound approaching $E^{(d+1)/2}$ as d gets large. This improvement requires fast rectangular matrix multiplication, however.

3 Applications

We now sketch how our new algorithm for discrete upper envelopes and discrete Voronoi diagrams automatically leads to better algorithms for various problems, by combining with existing techniques from computational geometry.

Diameter. Given a set P of n points in d dimensions, we consider the problem of computing a $(1 + O(\varepsilon))$-factor approximation of the diameter, i.e, the distance of the farthest pair of points.

We can adopt the following standard algorithm, described in [20] (see also [3, 19]): First compute a constant-factor approximation in $O(n)$ time (e.g., by picking any arbitrary point of P and taking the farthest neighbor distance from that point). By translation and scaling, we may assume that the diameter is $\Theta(1)$ and $P \subset [0, 1]^d$. Let Ξ be the set of all grid points over $\partial[-1, 1]^d$ with side length $\delta := \sqrt{\varepsilon}$. For each $\xi \in \Xi$, find a point $p_\xi \in P$ that maximizes $p_\xi \cdot \xi$ and a point $q_\xi \in P$ that maximizes $-q_\xi \cdot \xi$, while allowing additive error $O(\varepsilon)$. Return the maximum of $p_\xi \cdot \xi - q_\xi \cdot \xi$ over all $\xi \in \Xi$. See [20] for the correctness proof.

Observe that computing all the p_ξ's and q_ξ's corresponds to $O(1)$ instances of the discrete upper envelope problem (one per facet of $\partial[-1, 1]^d$). By applying Theorem 2 with $F = \sqrt{E}$, $U = O(EF)$, and $n \le E^{O(1)}$, we immediately obtain:

▶ **Corollary 3.** *Given n points in constant dimension d, we can compute a $(1 + \varepsilon)$-approximation of the diameter in $O((n\sqrt{E} + E^{d/2+1}) \log^{O(1)} E)$ time, where $E = \lceil 1/\varepsilon \rceil$.*

ε-kernels. Given a set P of n points in d dimensions, an *ε-kernel* is, roughly speaking, a subset $Q \subset P$ whose width approximates the width of P to within a factor of $1 + \varepsilon$ along every direction simultaneously. Alternatively, it can be viewed as a "coreset" for approximate convex hulls. The concept was introduced by Agarwal, Har-Peled, and Varadarajan [1] and has a plethora of applications; see [1, 2] for the precise definition and background.

Previous work [20, 30] suggested the following algorithm which computes an ε-kernel of worst-case optimal size $O((1/\varepsilon)^{(d-1)/2})$: First find an affine transformation that makes P "fat" and lie in $[-1, 1]^d$; this is known to be doable in $O(n)$ time. Let Ξ be the set of all grid points over $\partial[-2, 2]^d$ with side length $\sqrt{\varepsilon}$. For each $\xi \in \Xi$, find a nearest neighbor $p_\xi \in P$ to ξ, while allowing additive error $O(\varepsilon)$. Return the subset $\{p_\xi : \xi \in \Xi\}$. See [20] for the correctness proof.

Observe that computing all the p_ξ's reduces to $O(1)$ instances of the $(d - 1)$-dimensional discrete Voronoi diagram problem. (Technically, we need *witness finding*; see Appendix A for a solution, requiring Las Vegas randomization.) By Theorem 2, we immediately obtain:

▶ **Corollary 4.** *Given n points in constant dimension d, we can compute an ε-kernel of size $O(E^{(d-1)/2})$ in $O((n\sqrt{E} + E^{d/2+1}) \log^{O(1)} E)$ expected time, where $E = \lceil 1/\varepsilon \rceil$.*

Bichromatic closest pair. Given a set P of n red points and Q of n blue points in d dimensions, we next examine the problem of finding a $(1 + O(\varepsilon))$-factor approximation of the closest red-blue pair.

We first consider the "well-separated" case, where by translation, rotation, and scaling, we can make $P \subset [-1,1]^{d-1} \times [-2,-1]$ and $Q \subset [-1,1]^{d-1} \times [1,2]$. Arya and Chan [6] suggested the following algorithm to solve this case: Let Ξ be the set of all grid points over $[-1,1]^{d-1} \times \{0\}$ with side length $\delta := \sqrt{\varepsilon}$. For each $\xi \in \Xi$, find a nearest neighbor $p_\xi \in P$ to ξ and a nearest neighbor $q_\xi \in Q$ to ξ, while allowing additive error $O(\varepsilon)$. Return the closest pair (p_ξ, q_ξ) over all $\xi \in \Xi$. See [6] for the correctness proof.

Observe that computing all the p_ξ's reduces to $O(1)$ instances of the $(d-1)$-dimensional discrete Voronoi diagram problem. (Technically, we again need witness finding.) By Theorem 2, the running time is $O((n\sqrt{E} + E^{d/2+1}) \log^{O(1)} E)$. An alternative upper bound is $O(n^2)$, by brute-force search. The smaller of the two bounds is always at most $O(nE^{d/4+1/2} \log^{O(1)} E)$.

As observed by Arya and Chan [6], a simple grid approach can reduce the general problem to a number of well-separated instances whose input sizes sum to $O(n)$. Thus, the total time is at most $O(nE^{d/4+1/2} \log^{O(1)} E)$.

▶ **Corollary 5.** *Given n red and blue points in constant dimension d, we can compute a $(1+\varepsilon)$-approximate bichromatic closest pair in $O(nE^{d/4+1/2} \log^{O(1)} E)$ expected time, where $E = \lceil 1/\varepsilon \rceil$.*

Approximate nearest neighbor search. The result for bichromatic closest pair can be extended to (offline or online) approximate nearest neighbor search, by following Arya and Chan [6]. The techniques are more involved, requiring balanced box decomposition trees and ideas from earlier papers of Arya, da Fonseca, Malamatos, and Mount [12, 7]. We omit the details, but by reexamining [6] closely and incorporating our new time bound for discrete Voronoi diagrams, we obtain:

▶ **Corollary 6.** *We can preprocess n points in a constant dimension d in $O(n \log n) + O^*(nE^{d/4})$ expected time so that we can find a $(1+\varepsilon)$-approximate nearest neighbor to any query point in $O^*(E^{d/4} \log n)$ time, where $E = \lceil 1/\varepsilon \rceil$.*

Streaming diameter and ε-kernels. The same paper [6] also described an application to insertion-only streaming algorithms for approximating the diameter. Their solution requires first designing a data structure for approximate farthest neighbor queries using techniques similar to [12, 7], and then combining with Bentley and Saxe's logarithmic method (or "merge-and-reduce") [15]. We omit the details, but by examining [6] closely and incorporating our new time bound for discrete Voronoi diagrams, we obtain:

▶ **Corollary 7.** *Given a stream of n points in constant dimension d, we can maintain a $(1+\varepsilon)$-approximation of the diameter using $O(E^{(d-1)/2})$ space and supporting insertions in $O^*(E^{d/4})$ expected time, where $E = \lceil 1/\varepsilon \rceil$.*

The paper [6] also studied the insertion-only streaming algorithms for ε-kernels. Here, the solution is easier. We first consider the special case where the point set P is promised to be fat and lie in $[0,1]^d$ at all times. If we insist on $O(E^{(d-1)/2})$ space, we can handle insertions lazily until a block of $E^{(d-1)/2}$ points is read. Then following Section 3, we can recompute all the p_ξ's and q_ξ's by running our discrete upper envelope algorithm on $E^{(d-1)/2}$ points, taking $O(E^{d/2+1} \log^{O(1)} E)$ time. The amortized insertion time is $O(E^{d/2+1} \log^{O(1)} E)/E^{(d-1)/2} = O(E^{3/2} \log^{O(1)} E)$. (Deamortization is straightforward.)

Building on an earlier streaming algorithm in [20], Zarrabi-Zadeh [31] has given a reduction of the general problem to the above special case that does not increase the processing time or space in the insertion-only streaming model. As a result, we obtain:

▶ **Corollary 8.** *Given a stream of n points in constant dimension d, we can maintain an ε-kernel using $O(E^{(d-1)/2})$ space and supporting insertions in $O(E^{3/2}\log^{O(1)} E)$ expected time, where $E = \lceil 1/\varepsilon \rceil$.*

4 Second Algorithm

We now present an alternative algorithm for the diameter problem, which is also based on Chebyshev polynomials, but bypasses dynamic programming, instead using fast Fourier transform. It is slightly faster (the $E^{d/2+1}$ term in the time bound is reduced to $E^{(d+1)/2}$). It is also more direct, without going through discrete upper envelopes. The algorithm can also be applied to the bichromatic closest pair problem. An advantage is that it can be generalized to the L_s metric for any even integer constant s (although the algorithm for discrete Voronoi diagrams in Section 2 works also for L_s, the reductions from diameter and bichromatic closest pair in Section 3 rely on properties of Euclidean space).

▶ **Problem 2.** (Generalized Diameter) *Let d be a constant and φ be a d-variate $O(1)$-degree polynomial with integer coefficients. Given two sets P and Q of n points in $\mathbb{Z}^d$, we want to compute*

$$Z := \max_{(a_1,\ldots,a_d)\in P,\ (b_1,\ldots,b_d)\in Q} \varphi(a_1-b_1,\ldots,a_d-b_d)$$

while allowing additive error $O(\varepsilon U)$, where U is a known upper bound on $|\varphi(a_1-b_1,\ldots,a_d-b_d)|$ over all $(a_1,\ldots,a_d)\in P,\ (b_1,\ldots,b_d)\in Q$.

For example, for diameter in the L_s metric for an even integer constant s, we can first compute a constant-factor approximation in $O(n)$ time. By translation, scaling, and rounding, we may assume that the diameter is $\Theta(E)$ and $P \subset [E]^d$. Approximating the diameter with additive error $O(\varepsilon E)$ is equivalent to approximate the s-th power of the diameter with additive error $O(\varepsilon E^s)$. We then get an instance of Problem 2 with $\varphi(x_1,\ldots,x_d) = x_1^s + \cdots + x_d^s$. Here, $U = O(E^s)$, and $n \le E^{O(1)}$ after removing duplicates.

We now solve Problem 2 using the polynomial method. It suffices to solve the decision problem, of deciding whether the maximum is at least a given value t (with additive error $O(\varepsilon U)$), since the original problem can then be solved by binary search with $O(\log E)$ calls to the decision algorithm.

Reset $q := \left\lceil \sqrt{E}\ln(4n^2) \right\rceil$ and let D and the degree-q polynomial T be as in Section 2. Define

$$\widetilde{Z} := \sum_{(a_1,\ldots,a_d)\in P,\ (b_1,\ldots,b_d)\in Q} T(\varphi(a_1-b_1,\ldots,a_d-b_d)-t).$$

By a similar analysis as in Section 2, we have:

- Case 1: $Z \le t$. Then $\widetilde{Z} \le Dn^2$.
- Case 2: $Z \ge t + \varepsilon U$. Then $\widetilde{Z} > D(\frac{1}{2}e^{q\sqrt{\varepsilon}} - (n^2-1)) > Dn^2$ by our choice of q.

It suffices to compute $\widetilde{Z}$. At first $\widetilde{Z}$ appears expensive to compute, since we are summing n^2 polynomials. We follow the approach in Section 2 and use the Chinese remainder theorem. Define the set $\mathcal{P}$ of primes as before, with $M := Dn^2T_q(3) \le 2^{O(\sqrt{E}\log^{O(1)}(nEU))}$. We describe how to compute $\widetilde{Z}$ mod p for each $p \in \mathcal{P}$. Afterwards, we can reconstruct $\widetilde{Z}$ as before in at most $O(\log^2 M) = O(E\log^{O(1)}(nEU))$ time.

Fix a prime $p \in \mathcal{P}$. As before, for each $a_1,\ldots,a_d \in [p]$, let $w_{a_1,\ldots,a_d}$ be the number of points $(a'_1,\ldots,a'_d) \in P$ such that $a'_1 \equiv a_1,\ \ldots,\ a'_d \equiv a_d \pmod p$. Similarly, for each $b_1,\ldots,b_d \in [p]$, let $v_{b_1,\ldots,b_d}$ be the number of points $(b'_1,\ldots,b'_d) \in Q$ such that $b'_1 \equiv b_1,\ \ldots,\ b'_d \equiv b_d \pmod p$.

We can precompute all these counts by a linear scan over P and Q, using $O(n)$ arithmetic operations. Then

$$\widetilde{Z} \equiv \sum_{a_1,\ldots,a_d,b_1,\ldots,b_d \in [p]} w_{a_1,\ldots,a_d} v_{b_1,\ldots,b_d} T(\varphi(a_1-b_1,\ldots,a_d-b_d)-t) \pmod p \quad (5)$$

$$\equiv \sum_{c_1,\ldots,c_d \in [p]} u_{c_1,\ldots,c_d} T(\varphi(c_1,\ldots,c_d)-t) \pmod p, \quad (6)$$

where

$$u_{c_1,\ldots,c_d} := \sum_{a_1,\ldots,a_d \in [p]} w_{a_1,\ldots,a_d} v_{(a_1-c_1) \bmod p,\ldots,(a_d-c_d) \bmod p} \pmod p.$$

The key is to recognize this expression as a d-dimensional convolution (with wraparound indices modulo p). This can be converted to standard 1-dimensional convolution as follows: Initialize arrays $A[0,\ldots,(2p)^d]$ and $B[0,\ldots,(2p)^d]$ to 0. For each $a_1,\ldots,a_d \in [p]$, set $A[a_1(2p)^{d-1}+a_2(2p)^{d-2}+\cdots+a_d] = w_{a_1,\ldots,a_d}$. For each $b_1,\ldots,b_d \in [p]$, set $B[(p-b_1)(2p)^{d-1}+(p-b_2)(2p)^{d-2}+\cdots+(p-b_d)] = v_{b_1,\ldots,b_d}$. Compute the convolution $C[0,\ldots,(2p)^d]$ where $C[i] := \sum_{k=0}^{i} A[k]B[i-k] \pmod p$. Then for each $c_1,\ldots,c_d \in [p]$, set $u_{c_1,\ldots,c_d} = \sum_{j_1,\ldots,j_d \in \{0,1\}} C[(c_1+j_1p)(2p)^{d-1}+(c_2+j_dp)(2p)^{d-2}+\cdots+(c_d+j_dp)] \pmod p$.

By fast Fourier transform, we can compute all $u_{c_1,\ldots,c_d}$ values using $O(p^d \log p)$ arithmetic operations. Afterwards, we can compute $\widetilde{Z} \bmod p$ by (6) using $O(p^d)$ arithmetic operations. This assumes that we have precomputed $T(x)$ for all $x \in [p]$ (which straightforwardly requires $O(pq)$ arithmetic operations). All arithmetic operations are done modulo p, each costing at most $O(\log^2 p)$ time. The running time is thus $O(p^d \log^3 p) = O(\log^d M \log^3 \log M) = O(E^{d/2} \log^{O(1)}(nEU))$.

Including the cost of computing the counts, the running time is $O((n+E^{d/2}) \log^{O(1)}(nEU))$ for each fixed prime p. The total over all $O(\log M/\log\log M) = O(\sqrt{E}\log^{O(1)}(nEU))$ primes $p \in \mathcal{P}$ is $O((n\sqrt{E}+E^{(d+1)/2}) \log^{O(1)}(nEU))$.

▶ **Theorem 9.** *Problem 2 can be solved in $O((n\sqrt{E}+E^{(d+1)/2}) \log^{O(1)}(nEU))$ time, where $E = \lceil 1/\varepsilon \rceil$.*

5 Applications

L_s-diameter. The algorithm in Section 4 can immediately be applied to approximate the diameter in the L_s metric for any even integer constant s. For the case of odd s, we can use standard range-tree divide-and-conquer to reduce to bichromatic instances (P,Q) such that P and Q are separated along all d axis directions, in which case the preceding algorithm can be applied. The divide-and-conquer increases the running time by a factor of $O(\log^d n)$, which is $O(\log^d E)$ since $n \le E^{O(1)}$ (after initial rounding and removal of duplicates).

▶ **Corollary 10.** *Given n points in constant dimension d and any integer constant $s \ge 2$, we can compute a $(1+\varepsilon)$-approximation of the L_s-diameter in $O((n\sqrt{E}+E^{(d+1)/2}) \log^{O(1)} E)$ time, where $E = \lceil 1/\varepsilon \rceil$.*

Bichromatic L_s-closest pair. For bichromatic closest pair in the L_s metric for any even integer constant s, it suffices to solve the well-separated case, as noted in Section 3, where $P \subset B_P$ and $Q \subset B_Q$ for two unit hypercubes B_P and B_Q of distance $\Theta(1)$ apart. (Note that we can no longer rotate.) Approximating the closest pair distance with additive error $O(\varepsilon)$ is equivalent to approximating the s-th power of the closest pair distance with additive error $O(\varepsilon)$. We can round and rescale so that $P,Q \subset [E]^d$. We then get an instance of

Problem 2 with $\varphi(x_1, \ldots, x_d) = -(x_1^s + \cdots + x_d^s)$. Here, $U = O(E^s)$, and $n \leq E^{O(1)}$ after removing duplicates. The rest of the analysis is as in Section 3. The case of odd s can again be handled by incorporating range-tree divide-and-conquer.

▶ **Corollary 11.** *Given n red and blue points in constant dimension d and any integer constant $s \geq 2$, we can compute a $(1+\varepsilon)$-approximate bichromatic L_s-closest pair in $O(nE^{(d+1)/4} \log^{O(1)} E)$ time, where $E = \lceil 1/\varepsilon \rceil$.*

6 Final Remarks

We now reveal the origins of the ideas behind our first algorithm in Section 2.

- The application of the polynomial method to approximately find extreme points along arbitrary directions was first proposed by Andoni and Nguyen [5], specifically for the *dynamic streaming* model. This line of work was continued in [21] for the ε-kernel problem; in fact, the idea of applying the Chinese remainder theorem and keeping the counts $w_{a_1,\ldots,a_d}$ (which are easy to maintain in the dynamic streaming setting) is taken from [21]. However, it has not been realized before that the approach could give better algorithms in the standard nonstreaming setting. Also, these previous algorithms [5, 21] constructed polynomials by summing q-th powers rather than degree-q Chebyshev polynomials, which caused a larger degree bound on q (of the order E instead of $\sqrt{E}$) and thus larger ε-dependencies in time and space complexity.
- The theoretical computer science literature contains a number of earlier applications of Chebyshev polynomials. The closest to our work are perhaps the papers by Valiant [28] and Alman, Chan, and Williams [4] on approximate closest pair and offline nearest neighbor search in *high dimensions*. The latter paper also played with sums of Chebyshev polynomials, but the algorithms were put together quite differently. For example, they dealt primarily with polynomials with Boolean variables, they needed to expand polynomials into monomials, and they relied on fast matrix multiplication rather than dynamic programming.
- Related is another polynomial-method-based algorithm for #SAT by Chan and Williams [22]. There, a multivariate polynomial is evaluated over all points in $\{0,1\}^m$ in near 2^m time, without fast matrix multiplication. This subproblem in Boolean space reduces to computing a *Möbius* or *zeta transform*, for which a standard dynamic programming algorithm by Yates can be invoked [29, 16]. Our dynamic programming algorithm, to evaluate a polynomial over all points in the space $[E]^d$, is not entirely "original" and can be viewed as a variant of Yates' algorithm.

Our second algorithm in Section 4 which exploits fast Fourier transform seems more original, although the idea is simple in hindsight.

The main advantage of the polynomial method is its generality. For example, the approach in our second algorithm might potentially be applicable to *kinetic* variants of the diameter decision problem where points are moving according to $O(1)$-degree polynomial functions in time.

We can consider a still more general version of the diameter problem than Problem 2, where φ can be any $(2d)$-variate polynomial with integer coefficients and we seek $Z := \max_{(a_1,\ldots,a_d)\in P,\ (b_1,\ldots,b_d)\in Q} \varphi(a_1, \ldots, a_d, b_1, \ldots, b_d)$. Fast Fourier transform does not seem applicable here, and we have to adapt (5): $\widetilde{Z} \equiv \sum_{a_1,\ldots,a_d,b_1,\ldots,b_d\in[p]} w_{a_1,\ldots,a_d} v_{b_1,\ldots,b_d} T(\varphi(a_1, \ldots, a_d, b_1, \ldots, b_d) - t) \pmod p$, which can be

evaluated using $O(p^{2d})$ arithmetic operations by brute force (instead of $O(p^d \log p)$). This yields a slower (but still new) running time of $O((n\sqrt{E} + E^{d+(1/2)}) \log^{O(1)} E) = O^*(n + E^d)$.

To close, we mention two specific open problems:

- Can we approximate the width of a point set in $O^*(n + E^{d/2})$ time? The issue is that knowing an ε-kernel, we still need to compute the width of the kernel efficiently.
- Can we approximate the diameter in $O^*(n + E^{\alpha d})$ time for some absolute constant $\alpha < 1/2$?

References

1. P. K. Agarwal, S. Har-Peled, and K. R. Varadarajan. Approximating extent measures of points. *J. ACM*, 51(4):606–635, 2004. Preliminary version in SODA'01 and FOCS'01.
2. P. K. Agarwal, S. Har-Peled, and K. R. Varadarajan. Geometric approximation via coresets. In E. Welzl, editor, *Current Trends in Combinatorial and Computational Geometry*, pages 1–30. Cambridge University Press, 2005.
3. P. K. Agarwal, J. Matoušek, and S. Suri. Farthest neighbors, maximum spanning trees and related problems in higher dimensions. *Comput. Geom. Theory Appl.*, 1:189–201, 1991.
4. J. Alman, T. M. Chan, and R. Williams. Polynomial representation of threshold functions and algorithmic applications. In *Proc. 57th IEEE Sympos. Found. Comput. Sci. (FOCS)*, pages 467–476, 2016.
5. A. Andoni and H. L. Nguyen. Width of points in the streaming model. In *Proc. 23rd ACM–SIAM Sympos. Discrete Algorithms (SODA)*, pages 447–452, 2012. *ACM Trans. Algorithms*, to appear.
6. S. Arya and T. M. Chan. Better ε-dependencies for offline approximate nearest neighbor search, Euclidean minimum spanning trees, and ε-kernels. In *Proc. 30th Sympos. Comput. Geom. (SoCG)*, pages 416–425, 2014.
7. S. Arya, G. D. da Fonseca, and D. M. Mount. Approximate polytope membership queries. In *Proc. 43rd ACM Sympos. Theory Comput. (STOC)*, pages 579–586, 2011. *SIAM J. Comput.*, to appear.
8. S. Arya, G. D. da Fonseca, and D. M. Mount. Optimal area-sensitive bounds for polytope approximation. In *Proc. 28th Sympos. Comput. Geom. (SoCG)*, pages 363–372, 2012.
9. S. Arya, G. D. da Fonseca, and D. M. Mount. On the combinatorial complexity of approximating polytopes. In *Proc. 32nd Sympos. Comput. Geom. (SoCG)*, pages 11:1–11:15, 2016.
10. S. Arya, G. D. da Fonseca, and D. M. Mount. Near-optimal ε-kernel construction and related problems. In *Proc. 33rd Sympos. Comput. Geom. (SoCG)*, 2017.
11. S. Arya, G. D. da Fonseca, and D. M. Mount. Optimal approximate polytope membership. In *Proc. 28th ACM–SIAM Sympos. Discrete Algorithms (SODA)*, pages 270–288, 2017.
12. S. Arya, T. Malamatos, and D. M. Mount. Space-time tradeoffs for approximate nearest neighbor searching. *J. ACM*, 57:1–54, 2009. Preliminary version in SODA'02 and STOC'02.
13. S. Arya, D. M. Mount, N. Netanyahu, R. Silverman, and A. Y. Wu. An optimal algorithm for approximate nearest neighbor searching in fixed dimensions. *J. ACM*, 45:891–923, 1998. Preliminary version in SODA'94.
14. G. Barequet and S. Har-Peled. Efficiently approximating the minimum-volume bounding box of a point set in three dimensions. *J. Algorithms*, 38(1):91–109, 2001. Preliminary version in SODA'99.

15 J. L. Bentley and J. B. Saxe. Decomposable searching problems I: Static-to-dynamic transformation. *J. Algorithms*, 1(4):301–358, 1980.

16 A. Björklund, T. Husfeldt, and M. Koivisto. Set partitioning via inclusion-exclusion. *SIAM J. Comput.*, 39(2):546–563, 2009.

17 H. Breu, J. Gil, D. Kirkpatrick, and M. Werman. Linear time Euclidean distance transform algorithms. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 17:529–533, 1995.

18 T. M. Chan. Approximate nearest neighbor queries revisited. *Discrete Comput. Geom.*, 20:359–373, 1998. Preliminary version in SoCG'97.

19 T. M. Chan. Approximating the diameter, width, smallest enclosing cylinder, and minimum-width annulus. *Int. J. Comput. Geom.Appl.*, 12(1-2):67–85, 2002. Preliminary version in SoCG'00.

20 T. M. Chan. Faster core-set constructions and data-stream algorithms in fixed dimensions. *Comput. Geom. Theory Appl.*, 35(1-2):20–35, 2006. Preliminary version in SoCG'04.

21 T. M. Chan. Dynamic streaming algorithms for ε-kernels. In *Proc. 32nd Sympos. Comput. Geom. (SoCG)*, pages 27:1–27:11, 2016.

22 T. M. Chan and R. Williams. Deterministic APSP, orthogonal vectors, and more: Quickly derandomizing Razborov–Smolensky. In *Proc. 27th ACM–SIAM Sympos. Discrete Algorithms (SODA)*, pages 1246–1255, 2016.

23 D. Coppersmith. Rapid multiplication of rectangular matrices. *SIAM J. Comput.*, 11(3):467–471, 1982.

24 S. Har-Peled. A practical approach for computing the diameter of a point set. In *Proc. 17th Sympos. Comput. Geom. (SoCG)*, pages 177–186, 2001.

25 X. Huang and V. Y. Pan. Fast rectangular matrix multiplication and applications. *J. Complexity*, 14(2):257–299, 1998.

26 R. Motwani and P. Raghavan. *Randomized Algorithms*. Cambridge University Press, 1995.

27 O. Schwarzkopf. Parallel computation of distance transforms. *Algorithmica*, 6(5):685–697, 1991.

28 G. Valiant. Finding correlations in subquadratic time, with applications to learning parities and the closest pair problem. *J. ACM*, 62(2):13, 2015. Preliminary version in FOCS'12.

29 F. Yates. The design and analysis of factorial experiments. *Technical Communication No. 35, Commonwealth Bureau of Soil Science, Harpenden, UK*, 1937.

30 H. Yu, P. K. Agarwal, R. Poreddy, and K. R. Varadarajan. Practical methods for shape fitting and kinetic data structures using coresets. *Algorithmica*, 52(3):378–402, 2008. Preliminary version in SoCG'04.

31 H. Zarrabi-Zadeh. An almost space-optimal streaming algorithm for coresets in fixed dimensions. *Algorithmica*, 60(1):46–59, 2011. Preliminary version in ESA'08.

Appendix

A Finding Witnesses

One technical issue not addressed in Section 2 is how to find a *witness* point $(a_1, \ldots, a_d) \in P$ that approximately attains the maximum in (1), for every $(x_1, \ldots, x_{d-1}) \in [F]^{d-1}$. This is needed in some of the applications from Section 3. One standard approach to find such witnesses is via binary search, using a binary tree of subsets of P, but this seems to hurt

the $E^{d/2}$ term in the running time. We adopt another standard approach, using *random sampling* to isolate witnesses [26]. In the approximate setting, the details are trickier, but have been worked out in the previous paper [21]. Although that paper dealt with q-th powers instead of degree-q Chebyshev polynomials, the same ideas can be applied, as we now explain. (In fact, the details get a little simpler when we are not working in the streaming model.)

We assume that $P \subset [U]^d$ (which is true in all our applications). First let $\ell(a_1, \ldots, a_d) = a_1U^{d-1} + a_2U^{d-2} + \cdots + a_d + 1$ denote the *label* of a point $(a_1, \ldots, a_d) \in P$.

Let $k := \lceil \log n \rceil$. For each $j \in [k]$, draw a random sample $R_j \subset P$ where each point is chosen with probability $1/2^j$.

Reset $q := \left\lceil \sqrt{kE} \ln(10nU^{2d}) \right\rceil$. Define the polynomial functions

$$\widetilde{f}_j(x_1, \ldots, x_{d-1}, t) := \sum_{(a_1,\ldots,a_d)\in R_j} T(\psi_1(a_1, x_1) + \cdots + \psi_{d-1}(a_{d-1}, x_{d-1}) + a_d - t)$$

$$f^*_j(x_1, \ldots, x_{d-1}, t) := \sum_{(a_1,\ldots,a_d)\in R_j} \ell(a_1, \ldots, a_d)T(\psi_1(a_1, x_1) + \cdots + \psi_{d-1}(a_{d-1}, x_{d-1}) + a_d - t)$$

where $x_1, \ldots, x_{d-1} \in [F]$ and $|t| \in [U]$. We can evaluate $\widetilde{f}_j$ and f^*_j by the same approach as in Section 2 (after resetting $M := U^d \cdot DnT_q(3)$). The running time remains the same up to polylogarithmic factors, since the number of choices for j is $k = O(\log n)$, and the degree q increases only by a polylogarithmic factor.

Suppose we want to find a witness for a given $(x_1, \ldots, x_{d-1}) \in [F]^{d-1}$. We first find an approximation t to the maximum, with $t \le f(x_1, \ldots, x_{d-1}) \le t + \lceil \varepsilon U \rceil$, by the method in Section 2. Intuitively, if the number of witnesses is near 2^j, then with good probability exactly one witness is in R_j and the ratio $\frac{f^*_j(x_1,\ldots,x_{d-1},t)}{\widetilde{f}_j(x_1,\ldots,x_{d-1},t)}$ rounded to the nearest integer should give us the label to a witness, because the sums in the numerator and denominator are both dominated by a single term which corresponds to the witness. More care is needed in the approximate setting, however.

Let $\Delta := \lceil \lceil \varepsilon U \rceil / k \rceil$. More precisely, we claim that with probability $\Omega(1)$, the label of a witness can be found among the following ratios after rounding:

$$\frac{f^*_j(x_1, \ldots, x_{d-1}, t - i\Delta)}{\widetilde{f}_j(x_1, \ldots, x_{d-1}, t - i\Delta)} \qquad (i, j \in [k]).$$

To prove the claim, let $P_i = \{(a_1, \ldots, a_d) \in P : \psi_1(a_1, x_1) + \cdots + \psi_{d-1}(a_{d-1}, x_{d-1}) + a_d \ge t - i\Delta\}$. Any point in P_i for any $i \le k$ may be used as a witness with additive error $O(k\Delta) = O(\varepsilon U)$. Since $|P_0| \ge 1$, there exists $i \le k$ such that $|P_i| \le 2|P_{i-1}|$ (for otherwise, $|P_k| > 2^k \ge n$, a contradiction). Suppose $2^j \le |P_{i-1}| \le 2^{j+1}$. Then $|P_i| \le 2^{j+2}$. Let $\mathcal{E}$ be the event that exactly one point of P_{i-1} is chosen to be in R_j and no point of $P_i \setminus P_{i-1}$ is chosen to be in R_j. Then $\Pr(\mathcal{E}) \ge |P_{i-1}|\frac{1}{2^j}(1 - \frac{1}{2^j})^{|P_i|-1} \ge (1 - \frac{1}{2^j})^{2^{j+2}} \ge \Omega(1)$. Suppose that $\mathcal{E}$ is true. Let $(a_1, \ldots, a_d) \in P$ be the unique point of P_{i-1} that is chosen to be in R_j. Let $\ell = \ell(a_1, \ldots, a_d)$ and $T = T(\psi_1(a_1, x_1) + \cdots + \psi_{d-1}(a_{d-1}, x_{d-1}) + a_d - (t - i\Delta))$. Since $\psi_1(a_1, x_1) + \cdots + \psi_{d-1}(a_{d-1}, x_{d-1}) + a_d - (t - i\Delta) \ge \Delta$, by Lemma 1, $T \ge D \cdot \frac{1}{2}e^{\sqrt{\varepsilon/kq}} \ge 5DnU^{2d} \ge 5nU^dD\ell$; in other words, $nU^dD \le T/(5\ell)$. Thus,

$$\begin{aligned}\frac{f^*_j(x_1, \ldots, x_{d-1}, t - i\Delta)}{\widetilde{f}_j(x_1, \ldots, x_{d-1}, t - i\Delta)} &\in \left[\frac{\ell T - (n-1)U^dD}{T + (n-1)D}, \frac{\ell T + (n-1)U^dD}{T - (n-1)D}\right] \\ &\subset \left[\frac{\ell - 1/(5\ell)}{1 + 1/(5\ell)}, \frac{\ell + 1/(5\ell)}{1 - 1/(5\ell)}\right] \subset \left(\ell - \frac{1}{2}, \ell + \frac{1}{2}\right),\end{aligned}$$

as desired.

Since each division costs at most $O(\log^2 M)$, each witness can be found in $O(\log^2 M \log^2 n)$ time with success probability $\Omega(1)$. The total time for all $(x_1, \ldots, x_{d-1}) \in [F]^{d-1}$ is $O(F^{d-1} \log^2 M \log^2 n) = O(F^{d-1} E \log^{O(1)}(nEU))$. We can find all witnesses correctly by repeating the algorithm an expected $O(\log(F^{d-1}))$ number of times (since verifying a given witness is easy). Thus, the total time of the entire randomized (Las Vegas) algorithm remains the same in expectation, up to polylogarithmic factors.

B Small Improvement

In this appendix, we note a small speedup to the algorithm in Section 2 by exploiting fast Fourier transform and fast matrix multiplication. This improvement is mainly of theoretical interest.

For the base case of the dynamic program, observe that when $a_1, \ldots, a_{d-1}$ are fixed, equation (3) can be rewritten as a convolution of two p-dimensional vectors: letting $A[a_d] := w_{a_1,\ldots,a_d}$ and $B[x] = T(-x)$, we have $g^{(d)}_{a_1,\ldots,a_{d-1}}(t) \equiv \sum_{a_d \in [p]} A[a_d] B[t - a_d]$. By fast Fourier transform, the $O(p^{d-1})$ convolutions require $O(p^{d-1} \cdot p \log p)$ arithmetic operations.

For the main dynamic program, observe that equation (4) can be rewritten as a product of a $p^{d-2} \times p^2$ matrix and a $p^2 \times p^2$ matrix: letting

$$\begin{aligned}
C[(a_1, \ldots, a_{i-1}, x_{i+1}, \ldots, x_{d-1}), (x_i, t)] &:= g^{(i)}_{a_1,\ldots,a_{i-1}}(x_i, \ldots, x_{d-1}, t) \\
A[(a_1, \ldots, a_{i-1}, x_{i+1}, \ldots, x_{d-1}), (a_i, z)] &:= g^{(i+1)}_{a_1,\ldots,a_i}(x_{i+1}, \ldots, x_{d-1}, z) \\
B[(a_i, z), (x_i, t)] &:= \begin{cases} 1 & \text{if } z \equiv t - \psi_i(a_i, x_i) \pmod p \\ 0 & \text{else,} \end{cases}
\end{aligned}$$

we have $C[\xi, \eta] \equiv \sum_\zeta A[\xi, \zeta] B[\zeta, \eta]$. The computation requires $O(p^{\omega(d-2,2,2)})$ arithmetic operations, where $\omega(\alpha, \beta, \gamma)$ denotes the matrix multiplication exponent for multiplying an $n^\alpha \times n^\beta$ and an $n^\beta \times n^\gamma$ matrix. All arithmetic operations are done modulo p. The running time of the dynamic program is then $O(p^{\omega(d-2,2,2)} \log^2 p) = O(E^{\omega(d/2-1,1,1)} \log^{O(1)} E)$. The total over all $O(\sqrt{E} \log^{O(1)} E)$ primes $p \in \mathcal{P}$ gives $O(E^{\omega(d/2-1,1,1)+1/2} \log^{O(1)} E)$.

▶ **Theorem 12.** *Problem 1 can be solved in* $O((n\sqrt{E} + E^{\omega(d/2-1,1,1)+1/2} + F^{d-1}E) \log^{O(1)} E)$ *time, where* $E = \lceil 1/\varepsilon \rceil$.

Note that as d grows, $\omega(d/2-1, 1, 1) - d/2$ approaches 0, by known results on rectangular matrix multiplication [23, 25].