International Journal of Computational Geometry & Applications

OPTIMAL ADAPTIVE ALGORITHMS FOR FINDING THE NEAREST AND FARTHEST POINT ON A PARAMETRIC BLACK-BOX CURVE

ILYA BARAN and ERIK D. DEMAINE

Computer Science and Artificial Intelligence Laboratory
Massachusetts Institute of Technology
32 Vassar Street
Cambridge, MA 02139, USA
{*ibaran,edemaine*}*@mit.edu*

Received (received date)
Revised (revised date)
Communicated by (Name)

We consider a general model for representing and manipulating parametric curves, in which a curve is specified by a black box mapping a parameter value between 0 and 1 to a point in Euclidean d-space. In this model, we consider the *nearest-point-on-curve* and *farthest-point-on-curve* problems: given a curve C and a point p, find a point on C nearest to p or farthest from p. In the general black-box model, no algorithm can solve these problems. Assuming a known bound on the speed of the curve (a Lipschitz condition), the answer can be estimated up to an additive error of ε using $O(1/\varepsilon)$ samples, and this bound is tight in the worst case. However, many instances can be solved with substantially fewer samples, and we give algorithms that adapt to the inherent difficulty of the particular instance, up to a logarithmic factor. More precisely, if $\mathrm{OPT}(C, p, \varepsilon)$ is the minimum number of samples of C that every correct algorithm must perform to achieve tolerance ε, then our algorithm performs $O(\mathrm{OPT}(C, p, \varepsilon) \log(\varepsilon^{-1}/\mathrm{OPT}(C, p, \varepsilon)))$ samples. Furthermore, any algorithm requires $\Omega(k \log(\varepsilon^{-1}/k))$ samples for some instance C' with $\mathrm{OPT}(C', p, \varepsilon) = k$; except that, for the nearest-point-on-curve problem when the distance between C and p is less than ε, OPT is 1 but the upper and lower bounds on the number of samples are both $\Theta(1/\varepsilon)$. When bounds on relative error are desired, we give algorithms that perform $O(\mathrm{OPT} \cdot \log(2 + (1 + \varepsilon^{-1}) \cdot m^{-1}/\mathrm{OPT}))$ samples (where m is the exact minimum or maximum distance from p to C) and prove that $\Omega(\mathrm{OPT} \cdot \log(1/\varepsilon))$ samples are necessary on some problem instances.

Keywords: Adaptive algorithms; curves.

1. Introduction

Computational geometry has traditionally been focused on polygonal objects made up of straight line segments. In contrast, applications of geometric algorithms to computer-aided design and computer graphics usually involve more complex curves and surfaces. In recent years, this gap has received growing attention with algorithms for manipulating more general curves and surfaces, such as circular arcs,[6]

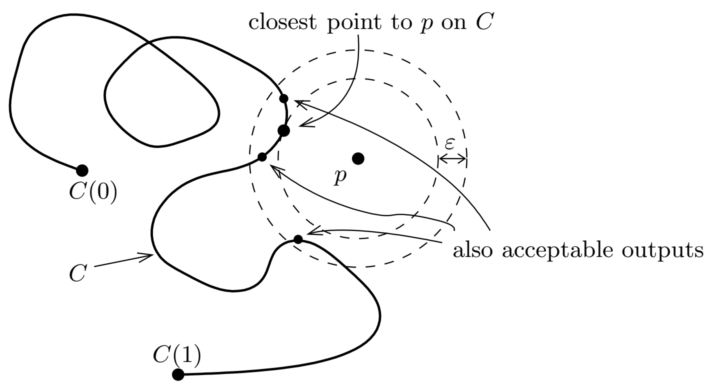

Fig. 1. An instance of the nearest-point-on-curve problem.

conic arcs,[3,17] and quadratic surfaces.[13] The most general type of curve commonly considered in this algorithmic body of work is a piecewise bounded-degree polynomial (algebraic) curve, although such curves are not usually manipulated directly and are more typically assumed to govern some process such as the motion of a polygon in kinetic collision detection.[2]

Parametric Black-Box Curves. A much more general model for specifying curves is the *parametric black-box model* that represents a curve in Euclidean d-space as a function $C\colon [0,1] \to \mathbb{R}^d$. The only operation that can be performed is to *sample* (evaluate) the function at a given parameter value $x \in [0,1]$.

Solving any nontrivial problem on a black-box curve requires some additional conditions on the behavior of the curve. We assume the *Lipschitz condition* that $\|C(x_1) - C(x_2)\| \le L|x_1 - x_2|$ for all $x_1, x_2 \in [0,1]$, for a known constant L. Any piecewise-C^1 curve has such a parameterization. By uniformly scaling the curve in $\mathbb{R}^d$, we can assume that the Lipschitz constant L is 1.

Nearest- and Farthest-Point-on-Curve Problems. In this paper, we solve two of the most basic proximity queries about black-box Lipschitz curves: given a curve C and a point p, find a point on C that is closest to p (*nearest point*), and find a point on C that is farthest from p (*farthest point*). In the black-box model, these problems are impossible to solve exactly, because an algorithm will never, in general, sample the nearest or farthest point. Thus, a problem instance also specifies an additive error tolerance ε, and our goal is to find a point on the curve C whose distance to the point p is within $\pm\varepsilon$ of the minimum or maximum possible. See Figure 1. Although we focus on absolute (additive) error in this paper, we show in Section 6 how to modify the absolute-error algorithms to obtain relative-error algorithms (whose output is accurate within a factor of $1 + \varepsilon$) that have nearly optimal adaptive performance.

Hard and Easy Instances. Any nearest-point-on-curve or farthest-point-on-curve instance can be solved using $1/2\varepsilon + O(1)$ samples: $C(0), C(2\varepsilon), C(4\varepsilon), \ldots, C(1)$. Unfortunately, this many samples can be necessary in the worst case. For example, when $C(x) = q$ for all x outside an interval of length 2ε where at speed 1 the curve moves toward p and then returns to q, we need $1/2\varepsilon - O(1)$ samples to find the interval. Thus, worst-case analysis is not very enlightening for this problem.

On the other hand, many instances are substantially easier. As an extreme example, if C is a unit-length line segment, then two samples, at $C(0)$ and $C(1)$, completely determine the curve by the Lipschitz condition.

Adaptive Analysis. Because the instance-specific optimal number of samples varies widely from $\Theta(1)$ to $\Theta(1/\varepsilon)$, we use the *adaptive analysis* framework, considered before in the context of boolean set operations[5] as well as sorting[7] and aggregate ranking.[8] In the adaptive analysis framework, the performance of an algorithm on a problem instance is compared to OPT, the performance of the best possible algorithm for that specific problem instance. By definition, for every problem instance, there exists an algorithm that achieves OPT on that instance. The question is whether *one* adaptive algorithm uses roughly $\mathrm{OPT}(C, p, \varepsilon)$ samples for *every* instance (C, p, ε).

Our Results. We develop adaptive algorithms that solve the nearest-point-on-curve and farthest-point-on-curve problems using $O(\mathrm{OPT}(C, p, \varepsilon) \log(\varepsilon^{-1}/\mathrm{OPT}(C, p, \varepsilon)))$ samples; except that, for the nearest-point-on-curve problem when the distance between C and p is less than ε, the number of samples may be $\Theta(1/\varepsilon)$, yet $\mathrm{OPT} = 1$. We also prove that these algorithms are *optimally adaptive* in the sense that no adaptive algorithm can achieve a strictly better bound (up to constant factors) with respect to OPT and ε. Specifically, we show that, for any $\varepsilon > 0$ and $k > 0$, there is a family of curves C each with $\mathrm{OPT}(C, p, \varepsilon) = k$ such that every algorithm (even randomized) requires $\Omega(k \log(\varepsilon^{-1}/k))$ samples on average for a curve C selected uniformly from the family; and there is a family of instances of the nearest-point-on-curve problem where the distance between C and p is less than ε such that every algorithm requires $\Omega(1/\varepsilon)$ samples on average, but OPT is 1.

Related Work. Because our curve model is a black box, the problems that we consider here have natural formulations in information-based complexity terms (see Traub *et. al.*[16] for an overview). However, information-based complexity is primarily concerned with worst-case, average-case, or randomized analysis of more difficult problems, rather than adaptive analysis of algorithms for easier problems as in this paper. Information-based complexity does consider adaptive algorithms (algorithms for which a query may depend on the answers to previous queries), but primarily when they are more powerful than non-adaptive algorithms in the worst (or average, etc.) case, such as for binary search.

The problem of maximizing a Lipschitz function has been studied in the context of global optimization. This problem essentially corresponds to the special case of the nearest- or farthest-point-on-curve problem in which $d = 1$. Beyond worst-case analysis, many algorithms for this problem have been studied only experimentally,[10] but Piyavskii's algorithm[14] has been previously analyzed in what is essentially the adaptive framework, first by Danilin.[4] The analysis was sharpened by Hansen *et. al.*[11] to show that the number of samples the algorithm performs on (C, ε) is at most 4 times $\mathrm{OPT}(C, \varepsilon)$. As Theorem 3 shows, this analysis cannot generalize to $d > 1$.

Practitioners who manipulate curves and surfaces typically use numerical algorithms, which are extremely general but sometimes fail or perform poorly, or specialized algorithms for specific types of curves, such as B-splines. Some algorithms for manipulating general parametric curves and surfaces guarantee correctness, but the theoretical performance of these algorithms is either not analyzed[15] or analyzed only in the worst-case.[12,9] At the heart of our algorithm is Günther and Wong's[9] observation that the portion of a Lipschitz curve between two nearby sample points can be bounded by a small ellipse, as described in Section 3.1.

2. Problem Statement

We use the real RAM model, which can store and manipulate exact real numbers in $O(1)$ time and space. Manipulation of real numbers includes basic arithmetic ($+$, $-$, $\times$, $\div$), comparisons, and nth roots. We separately analyze the number of samples and the additional computation time. Although we describe and analyze our algorithms in $\mathbb{R}^2$, both the algorithms and their analyses trivially carry over to $\mathbb{R}^d$ for $d > 2$.

We assume without loss of generality that the Lipschitz constant is 1, that the parameter space is the unit interval, and that the query point p is the origin O. Throughout our discussion of nearest-point-on-curve, $d_{\min}$ refers to the minimum distance from C to the origin. Analogously, $d_{\max}$ denotes the maximum distance from C to the origin. We assume that ε is smaller than $1/2$ because otherwise, a single sample at $1/2$ immediately solves both problems. The two problems we consider are to find a point on C whose distance to O is approximately $d_{\min}$ or $d_{\max}$:

Problem Nearest-Point-On-Curve *Given a Lipschitz curve C and an $0 < \varepsilon < 1/2$, find a parameter x such that $\|C(x)\| \leq d_{\min} + \varepsilon$.*

Problem Farthest-Point-On-Curve *Given a Lipschitz curve C and an $0 < \varepsilon < 1/2$, find a parameter x such that $\|C(x)\| \geq d_{\max} - \varepsilon$.*

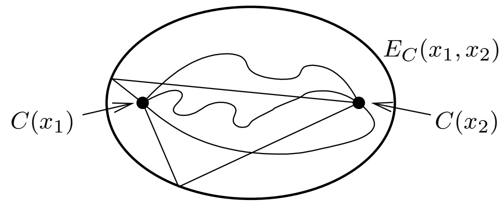

Fig. 2. Some possible curves C inside an ellipse.

3. Nearest-Point-On-Curve: Adaptive Algorithm and its Analysis

3.1. *Main Idea*

The main observation is that, if we have sampled C at x_1 and x_2, then for x between x_1 and x_2, $\|C(x)-C(x_1)\|+\|C(x)-C(x_2)\| \leq |x_2-x_1|$. This means that when the parameter x is between x_1 and x_2, $C(x)$ stays within an ellipse with foci at $C(x_1)$ and $C(x_2)$, whose major axis (sum of distances to foci from a boundary point) has length $|x_2 - x_1|$. See Figure 2. Note that this ellipse is tight: by changing C only between x_1 and x_2, we can force it to pass through any point in the ellipse while keeping C Lipschitz. The following propositions formalize this idea:

Definition 1. Given a Lipschitz curve C and an interval $[x_1, x_2] \subseteq [0, 1]$, define the ellipse

$$E_C(x_1, x_2) = \Big\{ p \in \mathbb{R}^2 \;\Big|\; \|C(x_1) - p\| + \|C(x_2) - p\| \leq x_2 - x_1 \Big\}.$$

Proposition 1. *For an interval $J = [x_1, x_2] \subseteq [0, 1]$, $C(J) \subseteq E_C(x_1, x_2)$.*

Proof. Let $x \in J$. By the Lipschitz condition, $\|C(x_1) - C(x)\| \leq x - x_1$ and similarly $\|C(x_2)-C(x)\| \leq x_2-x$. Adding these, we get $\|C(x_1)-C(x)\|+\|C(x_2)-C(x)\| \leq x_2 - x_1$, so $C(x) \in E_C(x_1, x_2)$. □

Proposition 2. *Let $J = (x_1, x_2) \subseteq [0, 1]$ and let C be a Lipschitz curve. Then for every point p in $E_C(x_1, x_2)$, there is a Lipschitz curve C' such that $C(x) = C'(x)$ for $x \notin J$ and for some $x \in J$, $C'(x) = p$.*

Proof. We can make C' on J consist of a line segment from $C(x_1)$ to p and another one from p to $C(x_2)$. Because the total length of these line segments is at most $x_2 - x_1$, we can parametrize C' at unit speed (or less) on J. □

The following proposition will often be used implicitly in our reasoning:

Proposition 3. *If $J' = [x'_1, x'_2]$ and $J = [x_1, x_2]$ and $J' \subseteq J$, then $E_C(x'_1, x'_2) \subseteq E_C(x_1, x_2)$.*

Proof. If $p \in E_C(x_1', x_2')$ then $\|C(x_1') - p\| + \|C(x_2') - p\| \le x_2' - x_1'$ by definition. We have $\|C(x_1) - C(x_1')\| \le x_1' - x_1$ and $\|C(x_2) - C(x_2')\| \le x_2 - x_2'$ by the Lipschitz condition on C. Adding the three inequalities and applying the triangle inequality twice, we get:

$$\|C(x_1) - p\| + \|p - C(x_2)\| \le \|C(x_1) - C(x_1')\| + \\ + \|C(x_1') - p\| + \|p - C(x_2')\| + \|C(x_2') - C(x_2)\| \le x_2 - x_1.$$

So $p \in E_C(x_1, x_2)$, as required. □

For notational convenience, let *closest-possible*(x_1, x_2) denote the minimum distance from a point in $E_C(x_1, x_2)$ to the origin.

3.2. *Proof Sets*

The properties of E_C immediately suggest a criterion for determining whether a set of points on a curve is sufficient to guarantee that a point sufficiently close to O is among those in the set: the distance from O to the nearest sampled point and the distance from O to the nearest ellipse (around adjacent points) should differ by at most ε.

Definition 2. Let $P = \{x_1, x_2, \ldots, x_n\}$ be a set of parameters in $[0, 1]$ so that $0 = x_1 < x_2 < \cdots < x_n = 1$. Let $x_{\min} \in P$ be an element that minimizes $\|C(x_i)\|$. Then P is a *proof set* if $\|C(x_{\min})\| - \varepsilon \le \text{closest-possible}(x_i, x_{i+1})$ for all i.

The following proposition shows that producing a proof set is the only way an algorithm can guarantee correctness.

Proposition 4. *Let $P = \{x_1, x_2, \ldots, x_n\} \subseteq [0, 1]$ so that $0 = x_1 < x_2 < \cdots < x_n = 1$. Let $x_{\min}$ be an element of P that minimizes $\|C(x_i)\|$. If P is a proof set, then for any curve C' such that $C'(x_i) = C(x_i)$, $x_{\min}$ is a solution to nearest-point-on-curve. Conversely, if P is not a proof set, there is a curve C' such that $C'(x_i) = C(x_i)$ for all i and for which $x_{\min}$ is not a solution.*

Proof. For any curve C' for which $C'(x_i) = C(x_i)$, P is a proof set for C' precisely when it is a proof set for C. Applying Proposition 1, we find that $C'([0, 1])$ is contained in the union of the ellipses $E_C(x_i, x_{i+1})$. So, if P is a proof set, $\|C'(x_{\min})\| - \varepsilon \le \|C'(x)\|$ for all $x \in [0, 1]$, which implies that $x_{\min}$ is a solution for C'.

Conversely, if P is not a proof set, then there is a point p in some ellipse $E_C(x_i, x_{i+1})$ such that $\|C(x_{\min})\| - \varepsilon > \|p\|$. By Proposition 2, we can construct a curve that coincides with C except in (x_i, x_{i+1}) and passes through p. For this curve, $x_{\min}$ will not be a solution. □

The requirement that $x_1 = 0$ and $x_n = 1$ allows the analysis to avoid special cases. An algorithm could guarantee correctness without sampling these endpoints, but because this saves only a constant amount of work, we ignore this possibility in favor of simpler analysis.

3.3. *Algorithm Description and Correctness*

As we sample the curve, we maintain a set of ellipses around the unsampled intervals. At each step, we take the interval whose ellipse is closest to the origin and sample in the middle of it, thus replacing it with two smaller intervals (with smaller ellipses). When the sampled points form a proof set, we terminate and output the closest point of those sampled.

Let Q be a priority queue that stores triples of real numbers (d, x_1, x_2) sorted by d. The algorithm is as follows:

CLOSEST-POINT(C, ε)

1. Add (closest-possible$(0, 1), 0, 1)$ to Q
2. If $\|C(0)\| < \|C(1)\|$ then $(\hat{x}_{\min}, \hat{d}_{\min}) \leftarrow (0, \|C(0)\|)$
   else $(\hat{x}_{\min}, \hat{d}_{\min}) \leftarrow (1, \|C(1)\|)$
3. Do until finished:
   4. $(d, x_1, x_2) \leftarrow$ EXTRACT-MIN(Q)
   5. If $\hat{d}_{\min} - \varepsilon \leq d$ then output $\hat{x}_{\min}$ and STOP
   6. $x \leftarrow (x_1 + x_2)/2$
   7. If $\|C(x)\| < \hat{d}_{\min}$ then $(\hat{x}_{\min}, \hat{d}_{\min}) \leftarrow (x, \|C(x)\|)$
   8. Add (closest-possible$(x_1, x), x_1, x)$ to Q
   9. Add (closest-possible$(x, x_2), x, x_2)$ to Q

Correctness follows from Proposition 4: the algorithm stops when the points sampled form a proof set and outputs the closest point. To show termination, we note that no interval of length 2ε or less is ever subdivided:

Proposition 5. *If in line 5, $x_2 - x_1 \leq 2\varepsilon$,* CLOSEST-POINT *terminates at this line.*

Proof. Because $\hat{d}_{\min}$ stores the minimum known distance to a point, $\hat{d}_{\min} \leq \|C(x_1)\|$ and $\hat{d}_{\min} \leq \|C(x_2)\|$. Let p be a point in $E_C(x_1, x_2)$ such that $\|p\| = d$. Then by the definition of E_C, $\|C(x_1) - p\| + \|C(x_2) - p\| \leq 2\varepsilon$. This means that at least one of $\|C(x_1) - p\| \leq \varepsilon$ or $\|C(x_2) - p\| \leq \varepsilon$ is true. If $\|C(x_1) - p\| \leq \varepsilon$, then, by the triangle inequality, $\|C(x_1)\| - \|p\| \leq \varepsilon$. This implies that $\hat{d}_{\min} - d \leq \varepsilon$ so the algorithm stops. Similarly for the other possibility. □

From this proposition, we can conclude that CLOSEST-POINT stops after at most $O(1/\varepsilon)$ loop iterations because only $O(1/\varepsilon)$ sample points at least ε apart can fit in $[0, 1]$, and in each iteration of the loop, the algorithm always samples one new point in an interval of width at least 2ε.

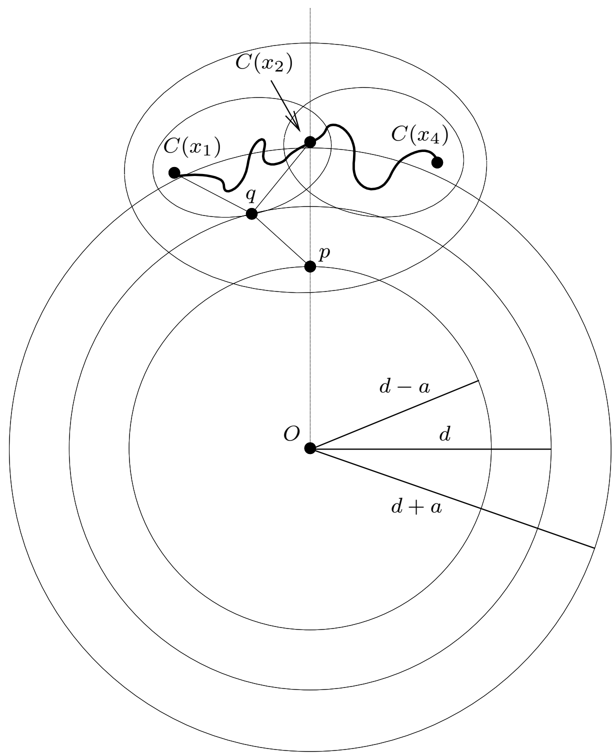

Fig. 3. Ellipse Lemma

3.4. *Ellipse Lemma*

To analyze CLOSEST-POINT, we will make use of one geometric fact in three incarnations:

Ellipse Lemma. *Let* $0 \leq x_1 \leq x_2 \leq x_3 \leq x_4 \leq 1$*. Also, let* $d, a \in \mathbb{R}$ *with* $0 < a < d$*. If closest-possible*$(x_1, x_2) \leq d$*, closest-possible*$(x_3, x_4) \leq d$*, and* $\|C(x_2)\| \geq d + a$*, then closest-possible*$(x_1, x_4) \leq d - a$*.*

Proof. See Figure 3. We may assume without loss of generality that $x_2 = x_3$, because if closest-possible$(x_3, x_4) \leq d$, then closest-possible$(x_2, x_4) \leq d$. Let p be the intersection of the circle $\|v\| = d - a$ and the ray from the origin through $C(x_2)$. Obviously, $\|p\| = d - a$. We will show that $\|C(x_1) - p\| \leq x_2 - x_1$ and $\|C(x_4) - p\| \leq x_4 - x_2$. This will prove that $p \in E_C(x_1, x_4)$, and therefore closest-possible$(x_1, x_4) \leq d - a$.

Because closest-possible$(x_1, x_2) \leq d$, there is a point q such that $\|q\| \leq d$ and $\|C(x_1) - q\| + \|q - C(x_2)\| \leq x_2 - x_1$. We may set the axes so that $C(x_2)$ is on the y axis. So let $C(x_2) = (0, y)$. Then $y \geq d + a$ and $p = (0, d - a)$. Now if $q = (x_q, y_q)$, then $\|q - C(x_2)\| = \sqrt{x_q^2 + (y - y_q)^2}$ and $\|q - p\| = \sqrt{x_q^2 + (y_q - (d-a))^2}$. Because $\|q\| \leq d$, $y_q \leq d$, we have $(y_q - (d-a))^2 \leq ((d+a) - y_q)^2 \leq (y - y_q)^2$, which means that $\|q - p\| \leq \|q - C(x_2)\|$. Using this, the triangle inequality, and the construction requirement of q, we get:

$$\begin{aligned} \|C(x_1) - p\| &\leq \|C(x_1) - q\| + \|q - p\| \leq \\ &\leq \|C(x_1) - q\| + \|q - C(x_2)\| \leq x_2 - x_1. \end{aligned}$$

The argument that $\|C(x_4) - p\| \leq x_4 - x_2$ is symmetric. □

When generalizing this lemma from $\mathbb{R}^2$ to $\mathbb{R}^d$, we consider separately the planes through $O, C(x_1), C(x_2)$ and through $O, C(x_2), C(x_4)$.

We will use the Ellipse Lemma in three different places in the analysis, so we prove three simple corollaries:

Corollary 1. *Let $[x_1, x]$ and $[x, x_2]$ be intervals. Let $0 < \varepsilon < d$. If closest-possible$(x_1, x_2) \geq d - \varepsilon$ and $\|C(x)\| \geq d$, then closest-possible$(x_1, x) \geq d - \varepsilon/2$ or closest-possible$(x, x_2) \geq d - \varepsilon/2$ or both.*

Proof. Suppose for contradiction that this is not the case: that we have closest-possible$(x_1, x) < d - \varepsilon/2$ and closest-possible$(x, x_2) < d - \varepsilon/2$. Let d' be the larger of closest-possible(x_1, x) and closest-possible(x, x_2). We have $d' < d - \varepsilon/2$. Now let $a = d - d'$ and apply the Ellipse Lemma to $[x_1, x]$, $[x, x_2]$, d' and a to get closest-possible$(x_1, x_2) \leq d' - a$. But $d' - a = 2d' - d < d - \varepsilon$, which contradicts the assumption that closest-possible$(x_1, x_2) \geq d - \varepsilon$. □

Corollary 2. *Let $[x_1, x_2]$ and $[x_3, x_4]$ be intervals with $x_3 \geq x_2$. Also let $0 < \varepsilon/2 < d$. If closest-possible$(x_1, x_2) \leq d$, closest-possible$(x_3, x_4) \leq d$, and $\|C(x_2)\| > d + \varepsilon/2$, then closest-possible$(x_1, x_4) < d - \varepsilon/2$.*

Proof. Let $a = \|C(x_2)\| - d$ so $a > \varepsilon/2$. Now apply the Ellipse Lemma to $[x_1, x_2]$, $[x_3, x_4]$, d, and a to get closest-possible$(x_1, x_4) \leq d - a < d - \varepsilon/2$. □

Corollary 3. *Let $[x_1, x]$ and $[x, x_2]$ be intervals and suppose that $0 < \varepsilon/2 < d$. If closest-possible$(x_1, x_2) \geq d - \varepsilon/2$ and $\|C(x)\| > d + \varepsilon/2$, then closest-possible$(x_1, x) > d$ or closest-possible$(x, x_2) > d$ or both.*

Proof. Suppose that closest-possible$(x_1, x) \leq d$ and closest-possible$(x, x_2) \leq d$. Apply Corollary 2 to get that closest-possible$(x_1, x_2) < d - \varepsilon/2$, which is a contradiction. □

3.5. *OPT*

We define the OPT of a problem instance to be the number of samples that the best possible algorithm makes on that instance. In our analysis, we use the fact that

OPT is equal to the size of the smallest proof set, which follows from Proposition 4. Note that OPT depends on C and ε, but we write OPT or OPT(ε) instead of OPT(C, ε) when the arguments are clear. For the analysis of CLOSEST-POINT with $\varepsilon < d_{\min}$ we need the following estimate: for any curve C, $\text{OPT}(\varepsilon/2) = O(\text{OPT}(\varepsilon))$. We prove this by starting with a proof set for ε, inserting a new sample point in between every pair of sample points in the proof set, and using the Ellipse Lemma with a continuity/connectedness argument to show that we can force the result to be a proof set for $\varepsilon/2$.

Proposition 6. *If* $\varepsilon < d_{\min}$*, for any problem instance* (C, ε)*,* $\text{OPT}(C, \varepsilon/2) \leq 2\text{OPT}(C, \varepsilon)$*.*

Proof. Consider a proof set P of size OPT(ε). Let x_i be the i^{th} smallest element of P. Because P is a proof set, closest-possible$(x_i, x_{i+1}) \geq d_{\min} - \varepsilon$. Now let

$$A = \{x \in [x_i, x_{i+1}] \mid \text{closest-possible}(x_i, x) \geq d_{\min} - \varepsilon/2\}$$
$$B = \{x \in [x_i, x_{i+1}] \mid \text{closest-possible}(x, x_{i+1}) \geq d_{\min} - \varepsilon/2\}$$

By Corollary 1 to the Ellipse Lemma, $A \cup B = [x_i, x_{i+1}]$. Also, because closest-possible$(x_i, x_i) = \|x_i\| \geq d_{\min}$ and closest-possible$(x_{i+1}, x_{i+1}) = \|x_{i+1}\| \geq d_{\min}$, $A \neq \emptyset$ and $B \neq \emptyset$. Because closest-possible is continuous in both variables, A is closed relative to $[x_i, x_{i+1}]$, being the preimage of the closed set $\{t \mid t \geq d_{\min} - \varepsilon\}$ under $t =$ closest-possible(x_i, x) with respect to the second variable. Similarly, B is closed relative to $[x_i, x_{i+1}]$. Because $[x_i, x_{i+1}]$ is connected, $A \cap B \neq \emptyset$, so let $x \in A \cap B$. This means closest-possible$(x_i, x) \geq d_{\min} - \varepsilon/2$ and closest-possible$(x, x_{i+1}) \geq d_{\min} - \varepsilon/2$. So for every pair of adjacent samples in P, we can insert a new sample x between them (x may, of course, coincide with one of the samples already in P, in which case we ignore it) so that in the resulting set, closest-possible$(x_j, x_{j+1}) \geq d_{\min} - \varepsilon/2$ for all j. Thus, we will have inserted at most an additional $|P| - 1$ elements. In order to make the result a proof set, we may need to insert one more element, x such that $\|C(x)\| = d_{\min}$. This will make the result into a proof set for $\varepsilon/2$ with $2|P|$ elements. □

3.6. *Phases*

We split an execution of CLOSEST-POINT into two phases and analyze each phase separately, giving an upper bound on the number of curve samples. The phases are a construction for the analysis only; the algorithm does not know which phase it is in. The algorithm starts out in Phase 1, and switches to Phase 2 when all of the ellipses around intervals stored in Q are no closer than $d_{\min} - \varepsilon/2$ to the origin. The distance from the ellipses in Q to the origin can only grow (as ellipses close to the origin are replaced by ellipses farther away), so once the algorithm enters Phase 2, it can never leave it. Let P be a proof set for $\varepsilon/2$ whose size is OPT$(\varepsilon/2)$. We show that in each phase, the number of samples is $O(|P| \log(\varepsilon^{-1}/|P|))$. We make use of the following easy fact:

Proposition 7. *Let a_i for $1 \leq i \leq |P|$ be positive real numbers. If we have $\sum_{i=1}^{|P|} a_i \leq \varepsilon^{-1}$, then $\sum_{i=1}^{|P|} \log a_i \leq |P| \log(\varepsilon^{-1}/|P|)$.*

Proof. By the arithmetic-geometric mean inequality, we have $\sqrt[|P|]{\prod_{i=1}^{|P|} a_i} \leq \frac{\sum_{i=1}^{|P|} a_i}{|P|} \leq \frac{\varepsilon^{-1}}{|P|}$. Taking the logarithm of both sides gives us $\frac{\sum_{i=1}^{|P|} \log a_i}{|P|} \leq \log(\varepsilon^{-1}/|P|)$. Multiplying both sides by $|P|$ gives us the desired result. □

3.7. *Phase 1*

Proposition 8. *If closest-possible$(x_1, x_2) < d_{\min} - \varepsilon/2$, then P must have a point in the open interval (x_1, x_2).*

Proof. For contradiction, suppose that P has no point in (x_1, x_2). This means that P has two consecutive points, x'_1 and x'_2, such that $[x_1, x_2] \subseteq [x'_1, x'_2]$. So $E_C(x_1, x_2) \subseteq E_C(x'_1, x'_2)$ and therefore, closest-possible$(x_1, x_2) \geq$ closest-possible(x'_1, x'_2), which means that closest-possible$(x'_1, x'_2) < d_{\min} - \varepsilon/2$. Hence, P cannot be a proof set for $\varepsilon/2$. □

Let $J = [x_1, x_2] \subseteq [0, 1]$ be an interval that is subdivided in Phase 1. This implies that closest-possible$(x_1, x_2) < d_{\min} - \varepsilon/2$ so by Proposition 8, P must have a point in (x_1, x_2). This means that any interval that is subdivided in Phase 1 contains a point of P.

We need to count the samples in this phase. We achieve this by classifying every subdivision as either a "split" or a "squeeze". A subdivision is a *split* if both resulting intervals contain points from P and a *squeeze* if one of the resulting intervals has no points from P. Because the number of splits cannot be more than $|P| - 1$, we only need to count squeezes. If J is an interval in Q at some point in the execution of Phase 1, let $S(J)$ be the number of squeezes that have happened to intervals containing J and let $L(J)$ be the length of J. We want the following invariant:

Proposition 9. *If at some point during Phase 1 of the algorithm, the intervals that intersect P are $J_1, J_2, \ldots, J_k$, then $\sum_{i=1}^{k} 2^{S(J_i)} L(J_i) = 1$.*

Proof. We proceed by induction on the number of subdivisions. At the start of the execution of CLOSEST-POINT, $S([0, 1]) = 0$ and $L([0, 1]) = 1$ so the base case is clearly true. Suppose an interval J_i is split into J_{i1} and J_{i2}. Because no new squeezes have occurred, $S(J_{i1}) = S(J_{i2}) = S(J_i)$ and $L(J_{i1}) = L(J_{i2}) = L(J_i)/2$. So $2^{S(J_i)} L(J_i) = 2^{S(J_{i1})} L(J_{i1}) + 2^{S(J_{i2})} L(J_{i2})$ and the sum is not changed. If the interval J_i is squeezed into J_{i1}, then $S(J_{i1}) = 1 + S(J_i)$ and $L(J_{i1}) = L(J_i)/2$ so $2^{S(J_i)} L(J_i) = 2^{S(J_{i1})} L(J_{i1})$ and the sum is not changed in this case either. □

Proposition 10. *There are $O(|P| \log(\varepsilon^{-1}/|P|))$ samples in Phase 1.*

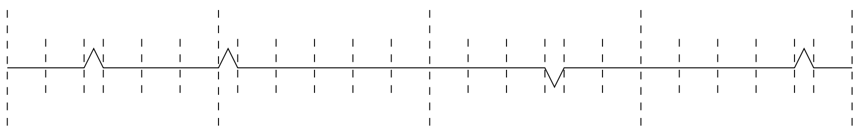

Fig. 4. An example C for the adaptive lower bound for $n = 24$ and $k = 4$.

Proof. As noted above, we only need to count the squeezes. By Proposition 9, at the end of Phase 1, if $J_1, \ldots, J_k$ contain points of P, then $\sum_{i=1}^{k} 2^{S(J_i)} L(J_i) = 1$. But because no interval of length ε or less ever appears, $L(J_i) > \varepsilon$ so $\sum_{i=1}^{k} 2^{S(J_i)} < \varepsilon^{-1}$ and $k \le |P|$. Using Proposition 7, we get $\sum_{i=1}^{k} S(J_i) \le k \log(\varepsilon^{-1}/k) = O(|P| \log(\varepsilon^{-1}/|P|))$. Every squeeze increases $\sum_{i=1}^{k} S(J_i)$ by 1 and no operation ever decreases it, so the number of squeezes is at most $O(|P| \log(\varepsilon^{-1}/|P|))$. □

3.8. *Phase 2*

If in Phase 2 a point x is sampled for which $\|C(x)\| \le d_{\min} + \varepsilon/2$, the algorithm stops. Because we are giving an upper bound on the running time, we may assume that every point sampled is farther than $d_{\min} + \varepsilon/2$ from the origin.

If closest-possible$(x_i, x_{i+1}) > d_{\min}$, then $[x_i, x_{i+1}]$ will never be chosen for subdivision. This is because an interval around a point that is $d_{\min}$ away from the origin has its ellipse at distance at most $d_{\min}$ from the origin and will be chosen over $[x_i, x_{i+1}]$. Thus, let us call an interval $[x_i, x_{i+1}]$ *alive* if closest-possible$(x_i, x_{i+1}) \le d_{\min}$ and call it *dead* otherwise. No dead interval is ever subdivided.

Proposition 11. *If $\varepsilon < d_{\min}$, then when the* CLOSEST-POINT *enters Phase 2, there are $O(|P|)$ alive intervals.*

Proof. Let the alive intervals at the start of Phase 2 be $[x_1, y_1], [x_2, y_2], \ldots, [x_k, y_k]$. From the assumption above, $\|C(x_i)\| > d_{\min} + \varepsilon/2$ and $\|C(y_i)\| > d_{\min} + \varepsilon/2$. Because the intervals are alive, closest-possible$(x_i, y_i) \le d_{\min}$. This means that we can apply Corollary 2 to the Ellipse Lemma to $[x_i, y_i]$ and $[x_{i+1}, y_{i+1}]$ to get closest-possible$(x_i, y_{i+1}) < d_{\min} - \varepsilon/2$. By Proposition 8, P has a point in (x_i, y_{i+1}). Because at most two segments of the form (x_i, y_{i+1}) can overlap, and each one has at least one point of P, there must be at most $2|P|$ of these segments. □

Now suppose we subdivide an interval $[x_1, x_2]$ into $[x_1, x]$ and $[x, x_2]$. Because the algorithm is in Phase 2, closest-possible$(x_1, x_2) \ge d_{\min} - \varepsilon/2$. By our assumption above, $\|C(x)\| > d_{\min} + \varepsilon/2$. Applying Corollary 3 to the Ellipse Lemma, we get that either closest-possible$(x_1, x) > d_{\min}$ or closest-possible$(x, x_2) > d_{\min}$ (or both). This implies that when the interval is subdivided, at most one of the resulting intervals can be alive.

Proposition 12. *If $\varepsilon < d_{\min}$, then* CLOSEST-POINT *performs at most $O(|P| \log(\varepsilon^{-1}/|P|))$ samples in Phase 2.*

Proof. Let $l_1, l_2 \ldots, l_k$ be the lengths of the alive intervals at time t. Define $p(t) = \sum_{i=1}^{k} \log_2(2l_i/\varepsilon)$. Because no interval of length 2ε or less is ever subdivided, $l_i > \varepsilon$ and so each term in the sum is at least 1. At every subdivision, an alive interval is replaced with at most one alive interval of half the length; therefore, each subdivision decreases $p(t)$ by at least 1. This implies that the total number of subdivisions cannot be greater than $p(t_0)$ where t_0 is the time when the algorithm enters Phase 2. Now consider the situation at time t_0. The total length of the alive intervals is at most 1, so we have $\sum_{i=1}^{k} 2l_i/\varepsilon \leq 2\varepsilon^{-1}$. Applying Proposition 7 to this inequality and to the definition of $p(t_0)$, we get $p(t_0) \leq k \log(2\varepsilon^{-1}/k)$. By Proposition 11, $k = O(|P|)$, so we get $p(t_0) = O(|P| \log(\varepsilon^{-1}/|P|))$, which means there are at most that many samples of C in phase 2. □

3.9. *Analysis Conclusion*

Theorem 1. *If on a problem instance with $\varepsilon < d_{\min}$, we let $n = \mathrm{OPT}(\varepsilon) \log(\varepsilon^{-1}/\mathrm{OPT}(\varepsilon))$, algorithm* CLOSEST-POINT *uses $O(n)$ samples and $O(n \log n)$ additional time, where the constant in the O notation is independent of the instance.*

Proof. Combining Propositions 10, 12, and 6, we get that the number of samples the algorithm makes is $O(n)$. Because the samples are stored in a priority queue, which may be implemented as a heap, it takes $O(\log n)$ time to insert or extract a sample. Hence the algorithm uses $O(n \log n)$ time for the heap operations. □

This theorem does not hold for $d_{\min} \leq \varepsilon$ because then the condition $a < d$ would not be satisfied when we invoke the Ellipse Lemma. The best conclusion we can make about the running time of CLOSEST-POINT when $d_{\min} \leq \varepsilon$ is that the number of samples is $O(1/\varepsilon)$. Below we prove that it is impossible to do better with respect to OPT.

4. Lower Bounds

4.1. *Worst-Case Lower Bound*

As mentioned in the introduction, in the worst case, we cannot do better than the trivial algorithm:

Theorem 2. *For any $\varepsilon > 0$, there is a problem instance of nearest-point-on-curve on which any algorithm requires $\Omega(\varepsilon^{-1})$ samples.*

Proof. Suppose we are given ε. Let C be the constant "curve", $C(x) = p$ for all $x \in [0, 1]$ with $\|p\| > \varepsilon$. Now, for any interval $[x_1, x_2] \subseteq [0, 1]$, $E_C(x_1, x_2)$ is a circle centered at p whose radius is $(x_2 - x_1)/2$. This means that in any proof set, every two points are less than $2\varepsilon^{-1}$ apart in the parameter space, so the OPT for this problem is $\Theta(\varepsilon^{-1})$. □

4.2. *Adaptive Lower Bound*

We prove that CLOSEST-POINT is optimal with respect to the number of samples of C.

Theorem 3. *For any algorithm, and for any $k \in \mathbb{N}$, and any $\varepsilon \in (0, 1/k)$, there is a problem instance with* OPT $= O(k)$ *on which that algorithm requires* $\Omega(k \log(\varepsilon^{-1}/k))$ *samples.*

Proof. Let ε and k be given. We will construct a problem instance family for which $k = \Omega(\text{OPT})$ and the number of samples required by any algorithm is $\Omega(k \log(\varepsilon^{-1}/k))$ on at least one instance of that family.

Let $n = \varepsilon^{-1}/3$. Divide the parameter space into n equal regions and group them into k groups of n/k regions each. In each group, let the curve have one spike in some region (and be flat in the other regions of that group). Let $k-1$ of the spikes point up, and let the remaining spike point down. See Figure 4. The origin is far below the curve and ε is less than the height of a spike, so that the only solutions to a nearest-point-on-curve instance of this form are on the spike pointing down. Because an omniscient adversary may force the last spike the algorithm examines to be the one pointing down, and the algorithm cannot determine whether a spike points up or down without sampling on it, the algorithm must find every spike. Note that if x is a point in parametric space that corresponds to the boundary between groups, $C(x)$ does not depend on where the spikes are chosen. Moreover, sampling inside one of the k groups (and not on a spike) only gives information about whether the spike in that group is to the left or to the right of the point sampled. This implies that the algorithm must perform a binary search on each of the k groups. The minimum number of samples to do this is indeed $\Omega(k \log(n/k))$.

To show that $k = \Omega(\text{OPT})$, we note that because the curve is piecewise-linear (and parametrized at unit speed), placing a point at every corner gives a proof set for any ε, because that completely determines the curve. Each spike has 3 corners and there are possibly two more endpoints, so OPT $\leq 3k + 2$. □

4.3. *Lower Bound for* $d_{\min} \leq \varepsilon$

Theorem 4. *For any algorithm and for any $\varepsilon > 0$, there is a problem instance with $d_{\min} \leq \varepsilon$ such that the algorithm requires* $\Omega(\text{OPT}(\varepsilon)\varepsilon^{-1})$ *samples to solve it.*

Proof. Because $d_{\min} \leq \varepsilon$, $\text{OPT}(\varepsilon) = 1$. To define C, let us split $[0, 1]$ into $\varepsilon^{-1}/4$ intervals of width 4ε. Fix one of these intervals, $J = (x_1, x_1 + 4\varepsilon)$. For $x \notin J$, let

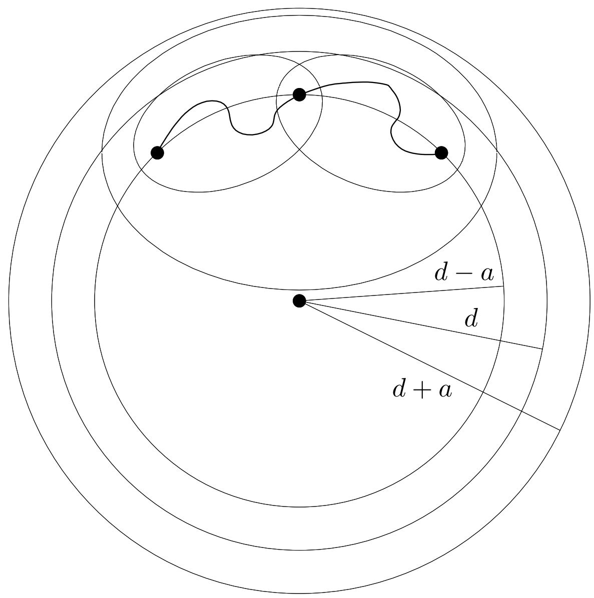

Fig. 5. A counterexample to the inverted Ellipse Lemma (ellipses are to scale)

$C(x) = (0, 2.5\varepsilon)$. For $x \in J$ let

$$C(x) = \begin{cases} (0, 2.5\varepsilon - (x - x_1)) & \text{for } x < x_1 + 2\varepsilon \\ (0, 2.5\varepsilon - (x_1 + 4\varepsilon - x)) & \text{for } x \geq x_1 + 2\varepsilon. \end{cases}$$

Informally, one of the intervals has a spike of height 2ε pointing at the origin. Now, at $x = x_1 + 2\varepsilon$, $C(x) = (0, \varepsilon/2)$, so $d_{\min} = \varepsilon/2$ and OPT $= 1$. The only valid outputs on such a problem instance are points on the spike. Because sampling the curve anywhere except J gives no information on the location of the spike, which could be in any of $\varepsilon^{-1}/4$ possible intervals, an algorithm is forced to do a linear search that requires $\Omega(\varepsilon^{-1})$ samples. $\square$

These lower bounds also work for randomized algorithms, because the reductions are from linear search and binary search, problems for which randomized algorithms can do no better than deterministic algorithms (up to constant factors).

5. Farthest-Point-on-Curve

It is natural to consider the symmetric problem of finding a point on C whose distance to a given point is within ε of the largest possible. It is straightforward to modify CLOSEST-POINT to FARTHEST-POINT, which solves the farthest-point-on-curve problem. The first two lower bounds for nearest-point-on-curve hold for farthest-point-on-curve as well. The analysis is also easy to carry over to FARTHEST-POINT, with one exception: the natural "inversion" of the Ellipse Lemma is false. Figure 5

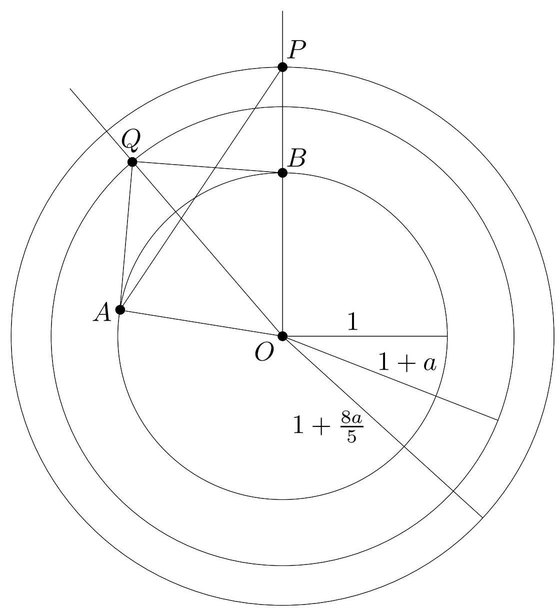

Fig. 6. Illustration for proof of Proposition 13

illustrates this. Nevertheless, the algorithm running time is the same (within a constant factor) because we can prove a modified inverted Ellipse Lemma. Note that farthest-possible(x_1, x_2) (the analogue of closest-possible(x_1, x_2)) refers to the maximum distance from a point in $E_C(x_1, x_2)$ to the origin. To simplify the proof, we impose an extra condition that $\|C(x_i)\| \leq d - a$, which was required only for $i = 2$ in the original lemma.

Inverted Ellipse Lemma. *Let* $0 \leq x_1 \leq x_2 \leq x_3 \leq x_4 \leq 1$*. Also, let* $d, a \in \mathbb{R}$ *such that* $0 < a < d$*. If farthest-possible*$(x_1, x_2) \geq d$*, farthest-possible*$(x_3, x_4) \geq d$*, and* $\|C(x_i)\| \leq d - a$ *for* $i \in \{1, 2, 3, 4\}$*, then farthest-possible*$(x_1, x_4) \geq d + \frac{3}{5}a$*.*

First, we state and prove what is essentially a special case of this lemma and then, to prove the lemma, we will reduce the general case to this special case.

Proposition 13. *Let* $a > 0$*, let* A *and* B *be points such that* $\|A\| = \|B\| = 1$ *and let* Q *be the intersection of the bisector of angle* AOB *with the circle* $\|v\| = 1+a$*. Let* P *be the intersection of ray* OB *with circle* $\|v\| = 1+8a/5$*. Then* $\|A-Q\|+\|Q-B\| \geq \|A - P\|$*.*

Proof. See Figure 6. First, note that $\|A - Q\| = \|Q - B\|$ so we need to show that $2\|A - Q\| \geq \|A - P\|$. Let c be the cosine of angle BOQ. Then the cosine of the angle BOA is $2c^2 - 1$ by the double angle formula. Using the Law of Cosines for the triangles AOQ and AOP, respectively, we write

$$\|A - Q\| = \sqrt{1 + (1+a)^2 - 2(1+a)c},$$

$$\|A - P\| = \sqrt{1 + \left(1 + \frac{8a}{5}\right)^2 - 2\left(1 + \frac{8a}{5}\right)(2c^2 - 1)}.$$

We prove the Proposition by expanding $(2\|A-Q\|)^2 - \|A-P\|^2$ and showing that it is positive:

$$4\|A-Q\|^2 - \|A-P\|^2 = 4 + 4(1+a)^2 - 8(1+a)c - 1 - \left(1 + \frac{8a}{5}\right)^2 + 2(2c^2-1)\left(1 + \frac{8a}{5}\right),$$

which simplifies to

$$4(c-1)^2 + \frac{36a^2}{25} + \frac{8a}{5}(c-1)(4c-1) \tag{1}$$

Notice that (1) is quadratic in c and the c^2 term has a positive coefficient. This means that it has a single global minimum as c varies. Because we are trying to prove that (1) is positive, it is sufficient to show that the expression takes on a positive value when c is at the minimum. To find the minimum, we differentiate (1) with respect to c to get

$$\frac{64a + 40}{5}c - 8a - 8,$$

which is equal to 0 for $c = \frac{5+5a}{5+8a}$. Substituting it back into (1), we get

$$\frac{36a^2}{(5+8a)^2} + \frac{36a^2}{25} - \frac{24a^2(15+12a)}{5(5+8a)^2} = \frac{288a^3}{25(5+8a)} \geq 0,$$

as desired. □

Proof of Inverted Ellipse Lemma. As in the original Ellipse Lemma, assume without loss of generality that $x_2 = x_3$. Let us also assume that the condition on the points is tight (it is straightforward to reduce to this case using the triangle inequality), that is, $\|C(x_1)\| = \|C(x_2)\| = \|C(x_4)\| = d - a$. Once again, let P be the intersection of the circle $\|v\| = d + \frac{3}{5}a$ and the ray from the origin through $C(x_2)$. We need to show that $|x_2 - x_1| \geq \|C(x_1) - P\|$ and the symmetric case ($|x_4 - x_2| \geq \|C(x_4) - P\|$) will follow. From combining them, we will be able to conclude the lemma. When applying Proposition 13, by scaling the entire picture, we can assume that $d - a = 1$ and so $d = 1 + a$.

In general, we only need to consider the case when the ellipse $E_C(x_1, x_2)$ is as small as possible, that is, it is tangent to the circle $\|v\| = d$. Let Q be a point of tangency. Then (from the optical properties of ellipses) the origin O is on the bisector of angle $C(x_1), Q, C(x_2)$. Therefore, if we reflect the line $Q, C(x_2)$ off the

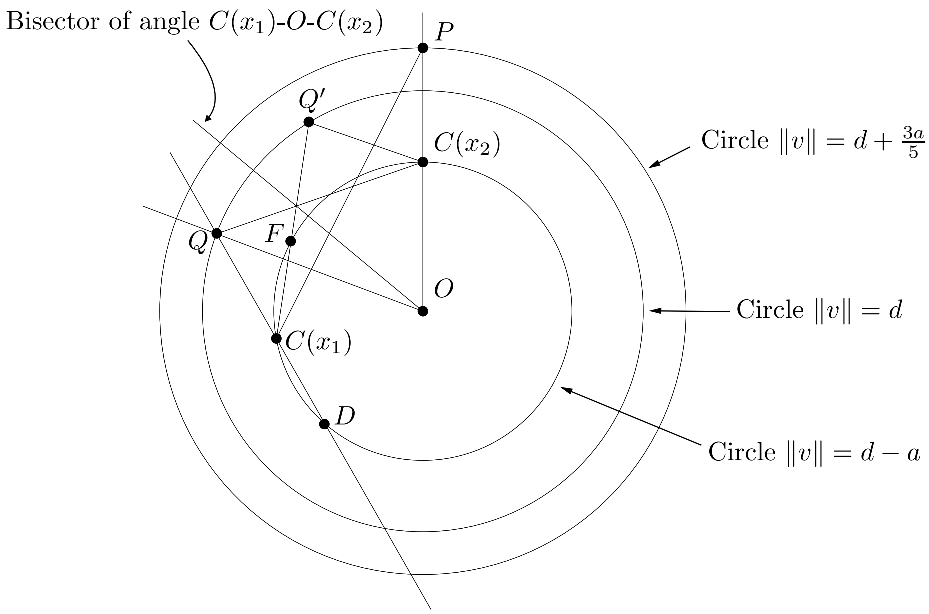

Fig. 7. Illustration for the second case of the proof

line QO, $C(x_1)$ must be on an intersection of the reflected line and the circle $\|v\| = d - a$. Let D be the other intersection. Because the reflection of $C(x_2)$ is both on the circle $\|v\| = d - a$ and on the reflected line, it could be either $C(x_1)$ or D. If it is $C(x_1)$, we can use Proposition 13, with $A \leftarrow C(x_1)$ and $B \leftarrow C(x_2)$ to conclude that $\|C(x_1) - Q\| + \|Q - C(x_2)\| \geq \|C(x_1) - P\|$. Otherwise, D is the reflection of $C(x_2)$ across QO and there are two possibilities:

1. If D is between $C(x_1)$ and Q, then we apply Proposition 13 with $A \leftarrow D$ and $B \leftarrow C(x_2)$ to get $\|D - Q\| + \|Q - C(x_2)\| \geq \|D - P\|$. We add $\|C(x_1) - D\|$ to both sides and apply the triangle inequality to get

$$|x_2 - x_1| \geq \|C(x_1) - Q\| + \|Q - C(x_2)\| = \|C(x_1) - D\| + \|D - Q\| + \\ + \|Q - C(x_2)\| \geq \|C(x_1) - D\| + \|D - P\| \geq \|C(x_1) - P\|.$$

2. Otherwise, $C(x_1)$ is between D and Q. See Figure 7. Let Q' be the reflection of Q across the bisector of angle $C(x_1), O, C(x_2)$. Then, $\|C(x_1) - Q'\| = \|C(x_2) - Q\|$ and $\|C(x_2) - Q'\| = \|C(x_1) - Q\|$. Now the line segment $C(x_1), Q'$ has a second intersection with the circle $\|v\| = d - a$ (besides the endpoint $C(x_1)$) because this line segment is the result of reflecting line segment DQ (which contains $C(x_1)$, by assumption) first across OQ and then across the bisector of $C(x_1), O, C(x_2)$. We will call this intersection F. F is the image of $C(x_1)$ under the two reflections described above, which implies that $\|C(x_1) - Q\| = \|F - Q'\|$. This means that $\|F - Q'\| = \|Q' - C(x_2)\|$ so we can apply Proposition 13 with $A \leftarrow F$, $B \leftarrow C(x_2)$,

and $Q \leftarrow Q'$ to get $\|F - Q'\| + \|Q' - C(x_2)\| \geq \|F - P\|$. We add $\|C(x_1) - F\|$ to both sides to get

$$|x_2 - x_1| \geq \|C(x_1) - Q'\| + \|Q' - C(x_2)\| = \|C(x_1) - F\| + \|F - Q'\| + \\ + \|Q' - C(x_2)\| \geq \|C(x_1) - F\| + \|F - P\| \geq \|C(x_1) - P\|,$$

as desired. □

Because the Inverted Ellipse Lemma has a weaker conclusion, in terms of the constant, than the original Ellipse Lemma, the analogue of Proposition 6 based on the Inverted Ellipse Lemma states that $\mathrm{OPT}(5\varepsilon/8) \leq 2\mathrm{OPT}(\varepsilon)$. This means that in order to get that $\mathrm{OPT}(3\varepsilon/8) = O(\mathrm{OPT}(\varepsilon))$, which we need for the analysis of Phase 2, we need to apply the analogue of Proposition 6 three times (because $(5/8)^3 < 3/8$).

The analysis of farthest-point-on-curve does not have the problem that nearest-point-on-curve has when $d_{\min} \leq \varepsilon$. Every time the Inverted Ellipse Lemma is used in the transformed proof, the condition that $a > d$ holds regardless of the curve or ε, unlike in nearest-point-on-curve.

Theorem 5. *On farthest-point-on-curve problem instance (C, ε), let $n = \mathrm{OPT}(\varepsilon)\log(\varepsilon^{-1}/\mathrm{OPT}(\varepsilon))$. Then algorithm* FARTHEST-POINT *uses $O(n)$ samples and $O(n \log n)$ additional time.*

6. Relative Error

We now examine modifications to our problems in which the goal is to guarantee a relative error bound instead of an absolute error bound. Specifically, for the nearest-point-on-curve problem, the objective is a parameter x such that $\|C(x)\| \leq (1 + \varepsilon)d_{\min}$; and for farthest-point-on-curve, we need $\|C(x)\| \geq d_{\max}/(1 + \varepsilon)$. We require that a nearest-point (farthest-point) problem instance has $d_{\min}$ ($d_{\max}$) nonzero, because otherwise the problem is unsolvable. It turns out that simple modifications to the absolute-error algorithms analyzed above yield adaptive relative-error algorithms. For proving an upper bound on the number of samples used by the algorithms, we focus on the nearest-point problem; for farthest-point, the upper bound analysis is analogous.

We start by defining a proof set for a relative-error nearest-point problem instance. Let $\hat{P}$ be a set of samples of C that includes 0 and 1, let $U_{\hat{P}}$ be the distance from the nearest point of $\hat{P}$ to the origin, and $L_{\hat{P}}$ be the distance from the nearest ellipse around adjacent points of $\hat{P}$ to the origin. We say that $\hat{P}$ is a proof set for the relative-error problem instance (C, ε) if $L_{\hat{P}} > 0$ and $U_{\hat{P}}/L_{\hat{P}} \leq 1 + \varepsilon$. It is easy to show the analogue to Proposition 4, that a proof set for relative error is required for a relative-error algorithm to guarantee correctness. So relative-error OPT is the size of a smallest proof set, minus at most 2 to account for the fact that including 0 and 1 may not be necessary.

To modify the absolute-error algorithm CLOSEST-POINT, first note that as it executes, $\hat{d}_{\min}$ is an upper bound on $d_{\min}$, and the top element of Q is a lower bound on $d_{\min}$. Let us call these values U and L, respectively. The termination condition in line 5 is that $U - L \leq \varepsilon$. If we replace it by the condition that $L > 0$ (to prevent division by zero) and $U/L \leq 1 + \varepsilon$, we get a relative-error algorithm.

Theorem 6. *The modified algorithm for the relative-error nearest-point-on-curve problem uses* $O(\mathrm{OPT} \cdot \log(2 + (1 + \varepsilon^{-1}) \cdot d_{\min}^{-1}/\mathrm{OPT}))$ *samples.*

Proof. Let $\varepsilon_{\text{ABS}} = \frac{\varepsilon \cdot d_{\min}}{1+\varepsilon}$. Notice that if $U - L \leq \varepsilon_{\text{ABS}}$, then because $L \leq d_{\min} \leq U$,

$$\frac{U}{L} \leq \frac{U}{U - \varepsilon_{\text{ABS}}} \leq \frac{d_{\min}}{d_{\min} - \varepsilon_{\text{ABS}}} = \frac{1}{1 - \frac{\varepsilon}{1+\varepsilon}} = 1 + \varepsilon.$$

So the relative-error algorithm with error ε terminates no later than a hypothetical execution of the absolute-error algorithm would with error ε_{ABS}. By Theorem 1, we know that such an absolute-error algorithm terminates after at most $O(\mathrm{OPT}_{\mathrm{ABS}} \cdot \log(2 + \varepsilon_{\text{ABS}}^{-1}/\mathrm{OPT}_{\mathrm{ABS}}))$ samples, where $\mathrm{OPT}_{\mathrm{ABS}}$ is the absolute-error OPT for ε_{ABS}. We now have an upper bound on the running time of the modified relative-error algorithm in terms of $\mathrm{OPT}_{\mathrm{ABS}}$. To complete the proof, we need to show a lower bound on OPT in terms of $\mathrm{OPT}_{\mathrm{ABS}}$.

In a relative-error proof set $\hat{P}$, $L_{\hat{P}} \geq d_{\min} - \frac{\varepsilon \cdot d_{\min}}{1+\varepsilon}$, because otherwise,

$$\frac{U_{\hat{P}}}{L_{\hat{P}}} \geq \frac{d_{\min}}{L_{\hat{P}}} > \frac{d_{\min}}{d_{\min} - \frac{\varepsilon \cdot d_{\min}}{1+\varepsilon}} = \frac{1}{1 - \frac{\varepsilon}{1+\varepsilon}} = 1 + \varepsilon.$$

So if we take a proof set for relative error ε and add a sample at distance $d_{\min}$ from the origin, we obtain a proof set for absolute error ε_{ABS}. This proves that $\mathrm{OPT}(\varepsilon) + 1 \geq \mathrm{OPT}_{\mathrm{ABS}}$. On the other hand, if we have an absolute-error proof set $\hat{P}$ for ε_{ABS}, we have $U_{\hat{P}} - L_{\hat{P}} \leq \varepsilon_{\text{ABS}}$, so $U_{\hat{P}}/L_{\hat{P}} \leq (1+\varepsilon)$, which implies that it is also a relative-error proof set for ε, and so $\mathrm{OPT}(\varepsilon) \leq \mathrm{OPT}_{\mathrm{ABS}}$. Therefore, the relative-error algorithm performs $O\left(\mathrm{OPT}(\varepsilon) \cdot \log\left(2 + \frac{1+\varepsilon}{\varepsilon \cdot d_{\min}} \middle/ \mathrm{OPT}(\varepsilon)\right)\right)$ samples. □

We modify the construction used in proving Theorem 3 to prove a lower bound for the relative-error problem.

Theorem 7. *For any algorithm and for any* $0 < \varepsilon < 1$ *and* $k \in \mathbb{N}$*, there is a problem instance with* $\mathrm{OPT} = O(k)$ *on which that algorithm requires* $\Omega(k \log(\varepsilon^{-1}))$ *samples to solve the relative-error problem.*

Proof. Consider a piecewise-linear curve segment as shown in Figure 8. Because such a segment is piecewise-linear, 5 samples are sufficient to obtain all information about it. We would like to show that for some combinations of S, L, and D, the only solutions to the relative-error problem are on the spike and it takes logarithmic time to find it.

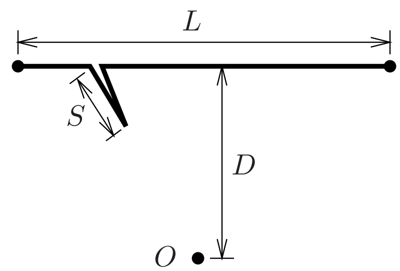

Fig. 8. An example curve segment on which the proof of Theorem 7 is based

In order for the only solutions to the relative-error nearest-point problem to be on the spike, the distance from O to the tip of the spike has to be smaller than $D/(1+\varepsilon)$. The distance from the tip of the spike to O is maximized when the spike is at one of the endpoints of the curve segment. In this case, the distance from the tip of the spike to O is $\sqrt{D^2+(L/2)^2}-S$. So we need $D > (1+\varepsilon)(\sqrt{D^2+L^2/4}-S)$, which is equivalent to $S/L > \sqrt{(D/L)^2+1/4}-D/(L+L\varepsilon)$. If $D/L = \frac{1}{2\sqrt{\varepsilon(2+\varepsilon)}}$, the inequality becomes:

$$\frac{S}{L} > \sqrt{\frac{1}{4\varepsilon^2+8\varepsilon}+\frac{1}{4}} - \frac{1}{(1+\varepsilon)(2\sqrt{\varepsilon^2+2\varepsilon})} = \frac{\sqrt{\varepsilon^2+2\varepsilon}}{2\varepsilon+2}.$$

So if we choose $S = L\sqrt{\varepsilon}$, the above inequality is satisfied (because $(2\varepsilon+2)\sqrt{\varepsilon} = \sqrt{4\varepsilon^3+8\varepsilon^2+4\varepsilon} > \sqrt{\varepsilon^2+2\varepsilon}$).

Therefore, we can construct a curve segment of arbitrarily small length $L + 2S$ with the only solutions to the nearest-point problem on a spike of size $2S$, which is no more than $2L\sqrt{\varepsilon}$. Sampling on the curve segment but not on the spike only gives information whether the spike is to the left or to the right of the point sampled. Therefore, a binary search taking $\Omega(\log((L+2S)/S)) = \Omega(\log(1/\varepsilon))$ steps is necessary to find the spike.

To construct the curve, simply paste k copies of curve segments, as described above (they may overlap), except make $k-1$ of the spikes point away from the origin and only one point toward it. Because the length of each curve segment can be arbitrarily small, the total length can be made exactly 1 (so, appropriately parameterized, the pasted curve is a valid input). The only solutions are on the spike pointing toward the origin. As in the argument for Theorem 3, a binary search is required to find each spike and a linear search on the curve segments is required to find the spike pointing toward the origin, giving a lower bound of $\Omega(k\log(1/\varepsilon))$. On the other hand, $\mathrm{OPT} \le 5k$.

For farthest-point, the construction is analogous to the one above, but “flipped”. On the curve segment on which the solution is located, the spike points away from the origin. To ensure that the only solutions are on the spike, the distance from the tip of the spike to O has to be at least $\sqrt{D^2+(L/2)^2}$. This distance is minimized

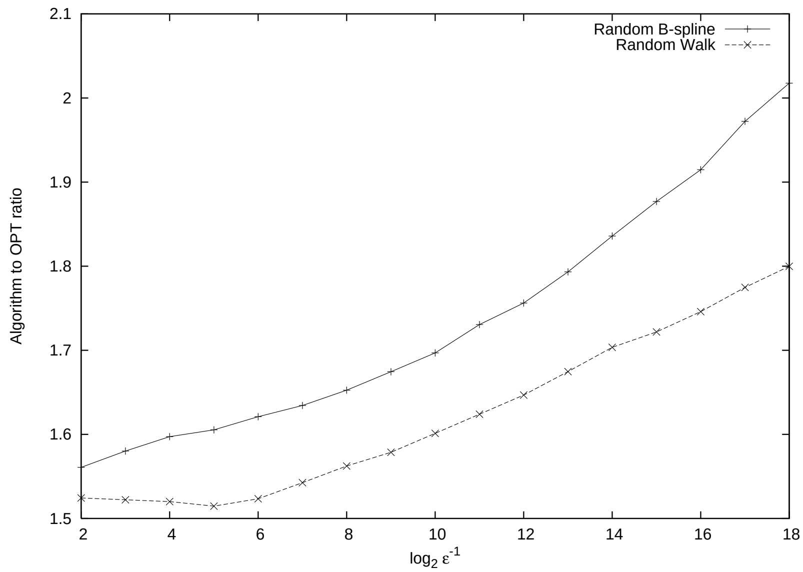

Fig. 9. The ratio of the number of samples CLOSEST-POINT performs to OPT

when the spike is in the middle of the curve segment and the distance from the tip to O is $D + S$. Thus, we need $D + S > (1+\varepsilon)\sqrt{D^2 + L^2/4}$, which is the same as $S/L > (1+\varepsilon)\sqrt{(D/L)^2 + 1/4} - D/L$. Notice that the right hand side is simply $(1+\varepsilon)$ times the right hand side of the analogous inequality for nearest-point. Therefore, if D/L is as for nearest point and $S = L(1+\varepsilon)\sqrt{\varepsilon}$ (which is still $O(L\sqrt{\varepsilon})$), the inequality is satisfied and a binary search on each curve segment requires $\Omega(\log(1/\varepsilon))$ samples. □

The upper and lower bounds for the relative-error problem do not match. We leave open the problem of finding an optimally adaptive algorithm in this setting.

7. Experiments

We have implemented CLOSEST-POINT and ran it on two datasets: one consisted of cubic B-splines with control points sampled uniformly from a square and the other consisted of random walks in the plane of 500 steps. Figure 9 shows the average ratio of CLOSEST-POINT's performance to OPT as a function of ε^{-1}. Notice that CLOSEST-POINT is within a factor of 2 of OPT on our datasets even for very small ε. The reason for this good performance is that when the curve is not carefully constructed to be difficult, the binary search (in which CLOSEST-POINT loses to OPT) needs to be performed on only a few sections of the curve. We believe that the reason CLOSEST-POINT did better relative to OPT on the random walks is that, on random walks, OPT is bigger and the algorithm performs binary searches on

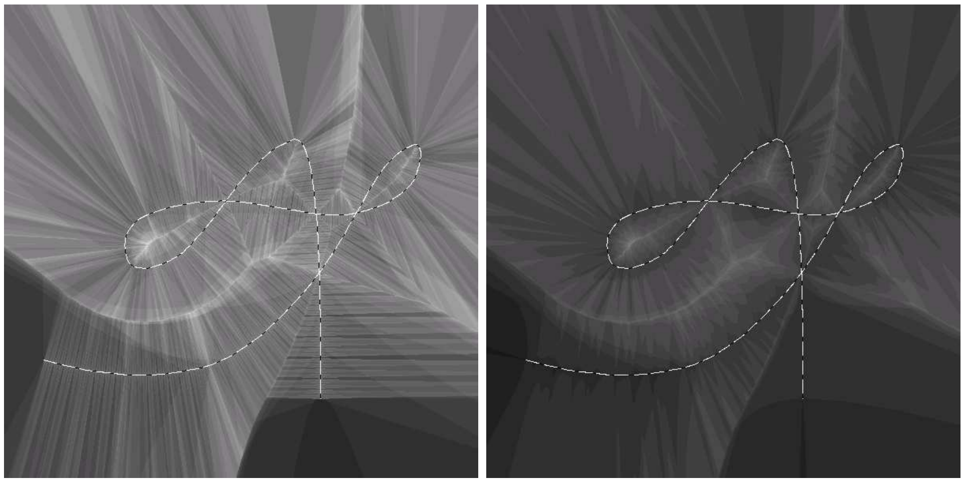

Fig. 10. Number of samples for CLOSEST-POINT(left) and OPT(right) to find the nearest point on the dashed curve. Darker shade indicates fewer samples. The colormap is the same in the images.

smaller segments. In general, the most difficult cases for both OPT and CLOSEST-POINT occur when O is near the medial axis of the curve. This effect can be seen in Figure 10 where every point on the plane is shaded depending on the number of samples CLOSEST-POINT and OPT would require if O were placed at that point.

8. Conclusion

The results in this paper give asymptotically tight bounds on the absolute-error nearest-point-on-curve and farthest-point-on-curve problems in the adaptive framework. We also show almost tight bounds in the relative-error setting.

A related problem is to preprocess a curve C and ε into a data structure that supports nearest-point and farthest-point queries for different points O. This problem can be solved using an adaptive approximation algorithm[1] that works similarly to CLOSEST-POINT to compute a parametric polyline that is within ε of C everywhere. Like CLOSEST-POINT, this computation requires a logarithmic overhead above the optimum in terms of the number of samples, but if unnecessary samples are pruned during preprocessing, the resulting polyline will have at most twice as many points as the optimal one. Exact queries to the polyline (achievable in $O(\log n)$ time using point location on the generalized Voronoi diagram) can then be used to answer queries to C with absolute error ε.

We believe that an analysis similar to ours can provide insight into the adaptive performance of algorithms for other curve problems based on Proposition 1, including those described by Günther and Wong.[9] A more difficult open problem is generalizing Proposition 1 from one-dimensional curves to two-dimensional surfaces in a way that allows algorithms based on the generalization. See the first author's

Master's Thesis[1] for related problems and results.

References

1. I. Baran. Adaptive algorithms for problems involving black-box lipschitz functions. Master's thesis, Massachusetts Institute of Technology, Cambridge, Massachusetts, May 2004.
2. J. Basch, J. Erickson, L. J. Guibas, J. Hershberger, and L. Zhang. Kinetic collision detection for two simple polygons. In *Proceedings of the 10th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 102–111, 1999.
3. E. Berberich, A. Eigenwillig, M. Hemmer, S. Hert, K. Mehlhorn, and E. Schömer. A computational basis for conic arcs and boolean operations on conic polygons. In *Proceedings of the 10th Annual European Symposium on Algorithms*, volume 2461 of *Lecture Notes in Computer Science*, pages 174–186, Rome, Italy, September 2002.
4. Y. Danilin. Estimation of the efficiency of an absolute-minimum-finding algorithm. *USSR Computational Mathematics and Mathematical Physics*, 11:261–267, 1971.
5. E. D. Demaine, A. López-Ortiz, and J. I. Munro. Adaptive set intersections, unions, and differences. In *Proceedings of the 11th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 743–752, San Francisco, California, January 2000.
6. O. Devillers, A. Fronville, B. Mourrain, and M. Teillaud. Algebraic methods and arithmetic filtering for exact predicates on circle arcs. *Computational Geometry: Theory and Applications*, 22:119–142, 2002.
7. V. Estivill-Castro and D. Wood. A survey of adaptive sorting algorithms. *ACM Computing Surveys*, 24(4):441–476, December 1992.
8. R. Fagin, A. Lotem, and M. Naor. Optimal aggregation algorithms for middleware. *Journal of Computer and System Sciences*, 66(4):614–656, 2003.
9. O. Günther and E. Wong. The arc tree: an approximation scheme to represent arbitrary curved shapes. *Computer Vision, Graphics, and Image Processing*, 51:313–337, 1990.
10. P. Hansen and B. Jaumard. Lipschitz optimization. In R. Horst and P. M. Pardalos, editors, *Handbook of Global Optimization*, pages 407–494. Kluwer, 1995.
11. P. Hansen, B. Jaumard, and S.-H. Lu. On the number of iterations of Piyavskii's global optimization algorithm. *Mathematics of Operations Research*, 16(2):334–350, May 1991.
12. D. E. Johnson and E. Cohen. A framework for efficient minimum distance computation. In *Proceedings of the IEEE Conference on Robotics and Animation*, pages 3678–3683, May 1998.
13. C. Lennerz and E. Schömer. Efficient distance computation for quadratic curves and surfaces. In *Proceedings of the 2nd Conference on Geometric Modeling and Processing*, pages 60–69, 2002.
14. S. Piyavskii. An algorithm for finding the absolute extremum of a function. *USSR Computational Mathematics and Mathematical Physics*, 12:57–67, 1972.
15. J. M. Snyder. Interval analysis for computer graphics. *ACM SIGGRAPH Computer Graphics*, 26(2):121–130, July 1992.
16. J. Traub, G. Wasilkowski, and H. Woźniakowski. *Information-Based Complexity*. Academic Press, New York, 1988.
17. R. Wein. High-level filtering for arrangements of conic arcs. In *Proceedings of the 10th Annual European Symposium on Algorithms*, volume 2461 of *Lecture Notes in Computer Science*, pages 884–895, Rome, Italy, September 2002.