Better Bounds for Online Line Chasing

Marcin Bienkowski
Institute of Computer Science, University of Wrocław, Poland
marcin.bienkowski@cs.uni.wroc.pl

Jarosław Byrka
Institute of Computer Science, University of Wrocław, Poland
jaroslaw.byrka@cs.uni.wroc.pl

Marek Chrobak
University of California at Riverside, CA, USA
marek@cs.ucr.edu

Christian Coester
University of Oxford, United Kingdom
christian.coester@cs.ox.ac.uk

Łukasz Jeż
Institute of Computer Science, University of Wrocław, Poland
lukasz.jez@cs.uni.wroc.pl

Elias Koutsoupias
University of Oxford, United Kingdom
elias@cs.ox.ac.uk

Abstract

We study online competitive algorithms for the *line chasing problem* in Euclidean spaces $\mathbb{R}^d$, where the input consists of an initial point P_0 and a sequence of lines $X_1, X_2, ..., X_m$, revealed one at a time. At each step t, when the line X_t is revealed, the algorithm must determine a point $P_t \in X_t$. An online algorithm is called c-competitive if for any input sequence the path $P_0, P_1, ..., P_m$ it computes has length at most c times the optimum path. The line chasing problem is a variant of a more general convex body chasing problem, where the sets X_t are arbitrary convex sets.

To date, the best competitive ratio for the line chasing problem was 28.1, even in the plane. We improve this bound by providing a simple 3-competitive algorithm for any dimension d. We complement this bound by a matching lower bound for algorithms that are memoryless in the sense of our algorithm, and a lower bound of 1.5358 for arbitrary algorithms. The latter bound also improves upon the previous lower bound of $\sqrt{2} \approx 1.412$ for convex body chasing in 2 dimensions.

2012 ACM Subject Classification Theory of computation → Online algorithms

Keywords and phrases convex body chasing, line chasing, competitive analysis

Digital Object Identifier 10.4230/LIPIcs.MFCS.2019.8

Funding Research supported by NSF grant CCF-1536026, by Polish National Science Centre grants 2016/22/E/ST6/00499 and 2015/18/E/ST6/0045, and by ERC Advanced Grant 321171 (ALGAME).

1 Introduction

Convex body chasing is a fundamental problem in online computation. It asks for an incrementally-computed path that traverses a given sequence of convex sets provided one at a time in an online fashion and is as short as possible. Formally, the input consists of an initial point $P_0 \in \mathbb{R}^d$ and a sequence $X_1, X_2, ..., X_m \subseteq \mathbb{R}^d$ of convex sets. The objective is to find a path $\mathbf{P} = (P_0, P_1, ..., P_m)$ with $P_t \in X_t$ for each $t = 1, 2, ..., m$ and minimum total length $\ell(\mathbf{P}) = \sum_{t=1}^{m} \ell_{P_{t-1}P_t}$. (Throughout the paper, by $\ell(P, Q)$ or ℓ_{PQ} we denote the Euclidean distance between points P and Q in $\mathbb{R}^d$.) This path $\mathbf{P}$ must be computed *online*, in the following sense: the sets X_t are revealed over time, one per time step. At step t, when set X_t is revealed, we need to immediately and irrevocably identify its visit point $P_t \in X_t$. Thus the choice of P_t does not depend on the future sets $X_{t+1}, ..., X_m$.

As can be easily seen, in this online scenario computing an optimal solution is not possible, and thus all we can hope for is to find a path whose length only approximates the optimum value. A widely accepted measure for the quality of this approximation is the competitive ratio. For a constant $c \geq 1$, we will say that an online algorithm $\mathcal{A}$ is *c-competitive* if it computes a path whose length is at most c times the optimum solution (computed offline). This constant c is called the *competitive ratio* of $\mathcal{A}$. Our objective is then to design an online algorithm whose competitive ratio is as close to 1 as possible.

The convex body chasing problem was originally introduced in 1993 by Friedman and Linial [11], who gave a constant-competitive algorithm for chasing convex bodies in $\mathbb{R}^2$ (the plane) and conjectured that it is possible to achieve constant competitiveness in any d-dimensional space $\mathbb{R}^d$. As shown in [11], this constant would have to depend on d; in fact it needs to be at least $\sqrt{d}$.

The Friedman-Linial conjecture has remained open for over two decades. In the last several years this topic has experienced a sudden increase in research activity, partly motivated by connections to machine learning (see [3, 7]), resulting in rapid progress. In 2016, Antoniadis *et al.* [1] gave a $2^{O(d)}$-competitive algorithm for chasing affine spaces of any dimension. In 2018, Bansal *et al.* [3] gave an algorithm with competitive ratio $2^{O(d \log d)}$ for nested families of convex sets, where the input set sequence satisfies $X_1 \supseteq X_2 \supseteq ... \supseteq X_m$. Soon later their bound was improved to $O(d \log d)$ by Argue *et al.* [2], and then to $O(\sqrt{d \log d})$ by Bubeck *et al.* [6]. Finally, Bubeck *et al.* [7] just recently announced a proof of the Friedman-Linial conjecture, providing an algorithm with competitive ratio $2^{O(d)}$ for arbitrary convex sets.

One other natural variant of convex body chasing that also attracted attention in the literature is *line chasing*, where all sets X_t are lines. Friedman and Linial [11] gave an online algorithm for line chasing in $\mathbb{R}^2$ with ratio 28.53. Their algorithm was simplified by Antoniadis *et al.* [1], who also slightly improved the ratio, to 28.1. Earlier, in 2014, Sitters [16] showed that a generalized work function algorithm has constant competitive ratio for line chasing, but he did not determine the value of the constant.

1.1 Our results

We study the line chasing problem discussed above. We give a 3-competitive algorithm for line chasing in $\mathbb{R}^d$, for any dimension $d \geq 2$, significantly improving the competitive ratios from [11, 1, 16]. Our algorithm is very simple and essentially memoryless, as it only needs to keep track of the last line in the request sequence. We start by providing the algorithm for line chasing in the plane, in Section 2, and later in Section 3 we extend it to an arbitrary dimension. In Section 4, we provide a matching lower bound of 3 for algorithms that are memoryless in the sense stated above and oblivious with respect to rotation, translation and uniform scaling of the metric space. We also provide a lower bound for arbitrary algorithms (see Section 5), showing that no online algorithm can achieve competitive ratio better than 1.5358. This improves the lower bound of $\sqrt{2} \approx 1.412$ for line chasing established in [11], which was previously also the best known lower bound for the more general problem of convex body chasing in the plane.

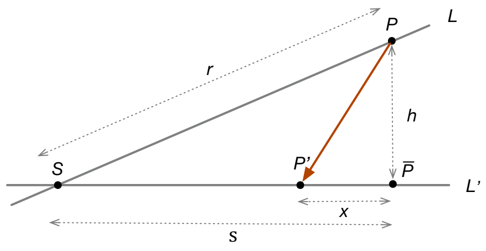

Figure 1 Algorithm DRIFT moves from P to P'.

1.2 Other related work

Set chasing problems are also known as *Metrical Service Systems* (see below) and belong to a very general class of problems for online optimization and competitive analysis called *Metrical Task Systems (MTS)* [5]. An instance of MTS specifies a metric space M, an initial point $P_0 \in M$, and a sequence of non-negative functions $\tau_1, \tau_2, ..., \tau_m$ over M called *tasks*. These tasks arrive online, one at a time. At each step t, the algorithm needs to choose a point $P_t \in M$ where it moves to "process" the current task τ_t. The goal is to minimize the total cost defined by $\sum_{t=1}^{m}(\mu(P_{t-1}, P_t) + \tau_t(P_t))$, where $\mu()$ is the metric in M. Thus in MTS, in addition to movement cost, at each step we also pay the cost of "processing" τ_t. For any metric space M with n points, if we allow arbitrary non-negative task functions then a competitive ratio of $2n - 1$ can be achieved and is optimal. This general bound is not particularly useful, because in many online optimization problems that can be modeled as an MTS, the metric space M has additional structure and only tasks of some special form are allowed, which makes it possible to design online algorithms with constant competitive ratios, independent of the size of M.

An MTS where $M = \mathbb{R}^d$ and all functions τ_t are convex is referred to as *convex function chasing*, and was studied in [1, 4, 13]. For the special case of convex functions on the real line, a 2-competitive algorithm was given in [4].

An MTS where each task function τ_t takes value 0 on a subset $X_t \subseteq M$ and ∞ elsewhere is called a *Metrical Service System (MSS)* [9]. In other words, in an MSS, in each step t the algorithm needs to move to a point in X_t. To achieve a competitive ratio independent of the size of M, it is generally required to restrict the sets X_t to be in some subset $\mathcal{X} \subsetneq \mathcal{P}(M)$. For instance, finite competitive ratios can be achieved when $\mathcal{X}$ is the set of sets of size at most k [10, 8, 15]. If $M = \mathbb{R}^d$ and $\mathcal{X}$ is the set of convex subsets, this is precisely the convex body chasing problem, and if $\mathcal{X}$ is the set of lines, it is the line chasing problem. One variant of MSS that has been particularly well studied is the famous *k-server problem* (see, for example, [14, 12]), in which one needs to schedule movement of k servers in response to requests arriving online in a metric space, where each request must be covered by one server. (In the MSS representation of the k-server problem, each set X_t consists of all k-tuples of points that include the request point at step t.)

2 A 3-Competitive Algorithm in the Plane

In this section, we present our online algorithm for line chasing in $\mathbb{R}^2$ with competitive ratio 3. The intuition is this: suppose that the last requested line is L and that the algorithm moved to point $P \in L$. Let L' be the new request line, S the intersection point of L and L', and $r = \ell_{SP}$. A naïve greedy algorithm would move to the point $\bar{P}$ on L' nearest to P (see

Figure 1) at cost $h = \ell_{P\bar{P}}$. If h is small, then $r - \ell_{S\bar{P}} = o(h)$, that is the distance between the greedy algorithm's point and S decreases only by a negligible amount. But the adversary can move to S, paying cost r, and then alternate requests on L and L'. On this sequence the overall cost of this algorithm would be $\omega(r)$, so it would not be constant-competitive. This example shows that if the angle between L and L' is small then the drift distance towards S needs to be roughly proportional to h. Our algorithm is designed so that this distance is roughly $h/\sqrt{2}$ if h is small (with the coefficient chosen to optimize the competitive ratio), and that it becomes 0 when L' is perpendicular to L.

Algorithm 1 Algorithm Drift.

Suppose that the last request is line L and that the algorithm is on point $P \in L$. Let the new request be L' and for any point $X \in L$, let $\bar{X}$ be the orthogonal projection of X onto L'. If L' does not intersect L, move to $P' = \bar{P}$. Otherwise, let $S = L \cap L'$ be the intersection point of L and L'. Let also $r = \ell_{SP}$, $h = \ell_{P\bar{P}}$, and $s = \ell_{S\bar{P}}$ (see Figure 1). Move to point $P' \in L'$ such that $\ell_{SP'} = s - x$, where $x = \frac{1}{\sqrt{2}}(h + s - r)$.

▶ **Theorem 1.** *Algorithm* Drift *is* 3*-competitive for the line chasing problem in* $\mathbb{R}^2$.

Proof. We establish an upper bound on the competitive ratio via amortized analysis, based on a potential function. The (always non-negative) value of this potential function, $\Phi(P, A)$, depends on locations $P, A \in L$ of the algorithm's and the adversary's point on the current line L. If L' is the new request line, and $P', A' \in L'$ are the new locations of the algorithm's and adversary's points, we want this function to satisfy

$$\ell_{PP'} + \Phi(P', A') - \Phi(P, A) \ \leq\ 3\,\ell_{AA'}. \tag{1}$$

Since initially the potential is 0 and is always non-negative, adding inequality (1) for all moves will establish 3-competitiveness of Algorithm Drift.

The potential function we use in our proof is $\Phi(P, A) = \sqrt{3}\,\ell_{AP}$. Substituting this formula, inequality (1) reduces to

$$\ell_{PP'} + \sqrt{3}\,(\,\ell_{A'P'} - \ell_{AP}\,) \ \leq\ 3\,\ell_{AA'}. \tag{2}$$

It thus remains to prove inequality (2). Let $g = \ell_{A\bar{A}}$, $z = \ell_{A'\bar{A}}$, and $v = \ell_{\bar{A}\bar{P}}$.

We first discuss the trivial case of non-intersecting L and L'. Keeping with the general notation, here we have $x = 0$ and thus $\ell_{PP'} = h$. Moreover, $g = \ell_{A\bar{A}} = h$ as well. For fixed z, we have $\ell_{AA'} = \sqrt{h^2 + z^2}$, i.e., the right hand side of (2) is fixed, whereas the left hand side is maximized if A' is on the other side of $\bar{A}$ than $\bar{P}$. The left hand side is thus at most

$$h + \sqrt{3}\,z \leq \sqrt{2}\,\sqrt{h^2 + 3z^2} \ \leq\ \sqrt{2}\,\sqrt{3\,(h^2 + z^2)} = \sqrt{6}\,\ell_{AA'} < 3\,\ell_{AA'},$$

where the first inequality follows from the power mean inequality (for powers 1 and 2), proving this easy case.

The situation when L' and L do intersect is illustrated in Figure 2. (The figure shows only the case when $\bar{A}$ is between S and P'.) Orient L' from left to right (with $\bar{P}$ being to the right of S), as shown in this figure. We want to express the distances in the above inequality in terms of s, h, v, and z (keeping in mind that x and r are functions of h and s):

$$\begin{aligned}
\ell_{PP'} &= \sqrt{x^2 + h^2} \\
\ell_{AP} &= vr/s \;=\; (v\sqrt{s^2 + h^2})/s \\
\ell_{AA'} &= \sqrt{z^2 + g^2}
\end{aligned}$$

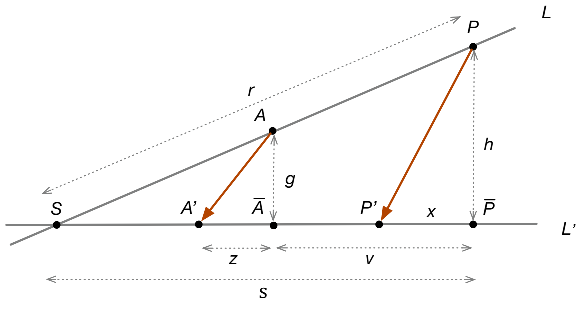

Figure 2 Notation for the analysis of Algorithm Drift.

The values of g and $\ell_{A'P'}$ depend on some cases, that we consider below.

Case 1. $\bar{A}$ is between S and P', as in Figure 2. Then $g = h(s-v)/s$. Our goal is first to find A' for which the bound in (2) is tightest. For a given z, among the two locations of A' at distance z from $\bar{A}$, the one on the left gives a larger value of the left-hand side of (2), while the right-hand side is the same for both. Thus we can assume that A' is to the left of $\bar{A}$, so $\ell_{A'P'} = z + v - x$. Then we can rewrite (2) as follows:

$$\tfrac{1}{\sqrt{3}}\,\ell_{PP'} - \ell_{AP} + v - x \;\le \sqrt{3}\,\sqrt{z^2+g^2} - z \tag{3}$$

By elementary calculus, the right-hand side is minimized for $z = \frac{1}{\sqrt{2}}\,g$, so we can assume that z has this value. Then inequality (3) reduces to

$$\tfrac{1}{\sqrt{3}}\,\ell_{PP'} - \ell_{AP} + v - x \;\le \sqrt{2}\,g. \tag{4}$$

After substituting $g = h(s-v)/s$ and $\ell_{AP} = vr/s$, inequality (4) reduces further to

$$s\,(\tfrac{1}{\sqrt{3}}\,\ell_{PP'} - x - \sqrt{2}h) \;\le v\,(r - s - \sqrt{2}\,h). \tag{5}$$

The expression in the parenthesis on the right-hand side of (5) is non-positive by triangle inequality, so the right-hand side is minimized when v is maximized, that is $v = s$, and then it reduces to

$$x^2 + h^2 \;\le 3(r - s + x)^2. \tag{6}$$

Recall that $x = \frac{1}{\sqrt{2}}(h + s - r)$. Since $r - h \le s \le r$, we have

$$\begin{aligned} x^2 + h^2 \;&=\; \tfrac{1}{2}(h+s-r)^2 + h^2 \\ &\le\; \tfrac{1}{2}h^2 + h^2 \\ &=\; \tfrac{3}{2}h^2 \;\le\; \tfrac{3}{2}[\,h + (\sqrt{2}-1)(r-s)\,]^2 \;=\; 3(r-s+x)^2, \end{aligned}$$

proving (6).

Case 2. $\bar{A}$ is before S. In this case we have $g = h(v-s)/s$. Just as in Case 1, we can assume that A' is to the left of $\bar{A}$, so that $\ell_{A'P'} = z + v - x$, and (2) reduces to

$$\tfrac{1}{\sqrt{3}}\,\ell_{PP'} - \ell_{AP} + v - x \;\le \sqrt{2}\,g. \tag{7}$$

After substituting $g = h(v-s)/s$ and $\ell_{AP} = vr/s$, inequality (4) reduces further to

$$s\,(\tfrac{1}{\sqrt{3}}\,\ell_{PP'} - x + \sqrt{2}h) \;\le v\,(r - s + \sqrt{2}\,h). \tag{8}$$

The expression in the parenthesis on the right-hand side of (8) is non-negative, so the right-hand side is minimized when $v = s$ (because in this case $v \geq s$), so (8) reduces to the same inequality (6) as in Case 1, completing the argument for Case 2.

Case 3. $\bar{A}$ is after $\bar{P}$. In this case we have $g = h(v+s)/s$. Symmetrically to Case 1, we can now assume that A' is to the right of $\bar{A}$, so that $\ell_{A'P'} = z + v + x$, and that $z = \frac{1}{\sqrt{2}}\, g$. Then, analogously to (4), we can rewrite (2) as follows:

$$\tfrac{1}{\sqrt{3}}\,\ell_{PP'} - \ell_{AP} + v + x \ \leq \sqrt{2}\, g \tag{9}$$

After substituting $g = h(v+s)/s$ and $\ell_{AP} = vr/s$, inequality (9) reduces further to

$$s\,(\tfrac{1}{\sqrt{3}}\,\ell_{PP'} + x - \sqrt{2}h) \ \leq v\,(r - s + \sqrt{2}\,h). \tag{10}$$

The expression in the parenthesis on the right-hand side of (10) is non-negative, so the right-hand side is minimized when $v = 0$, and then it reduces to

$$x^2 + h^2 \ \leq 3(\sqrt{2}h - x)^2. \tag{11}$$

To prove this, we proceed similarly as in Case 1:

$$x^2 + h^2 \ \leq \ \tfrac{3}{2}h^2 \ \leq \ \tfrac{3}{2}(h + r - s)^2 \ = \ 3(\sqrt{2}h - x)^2,$$

proving (11).

Case 4. $\bar{A}$ is between P' and $\bar{P}$. Then $g = h(s-v)/s$ (as in Case 1). Similar to Case 3, we can assume that A' is to the right of $\bar{A}$, so that now $\ell_{A'P'} = z - v + x$, and that $z = \frac{1}{\sqrt{2}}\, g$. Then, analogously to (4), we can rewrite (2) for this case as follows:

$$\tfrac{1}{\sqrt{3}}\,\ell_{PP'} - \ell_{AP} - v + x \ \leq \sqrt{2}\, g \tag{12}$$

After substituting $g = h(s-v)/s$ and $\ell_{AP} = vr/s$, inequality (12) reduces further to

$$s\,(\tfrac{1}{\sqrt{3}}\,\ell_{PP'} + x - \sqrt{2}h) \ \leq v\,(r + s - \sqrt{2}\,h). \tag{13}$$

We now have two sub-cases. If the expression in the parenthesis on the right-hand side of (13) is non-negative then the right-hand side is minimized when $v = 0$, so inequality (13) reduces to inequality (11) from Case 3. If this expression is negative (that is when $r + s < \sqrt{2}h$), then it is sufficient to prove (13) with v on the right-hand side replaced by s (because $v \leq s$). This reduces it to $\frac{1}{\sqrt{3}}\,\ell_{PP'} + x \leq r + s$. This last inequality follows from $\ell_{PP'} \leq r$ and $x \leq s$. ◀

3 An Algorithm for Arbitrary Dimension

In this section, we show how to extend Algorithm Drift to Euclidean spaces $\mathbb{R}^d$ for arbitrary dimension $d \geq 2$. This extension, that we call ExtDrift, is quite simple, and consists of projecting the whole space onto an appropriately chosen plane that contains the new request line. While such approach was suggested already by Friedman and Linial [11], their choice of plane may lose a constant factor in the competitive ratio. We project onto a different plane, which allows ExtDrift to also be 3-competitive.

Let P be the current ExtDrift position and L' the new request line. If $P \in L'$, ExtDrift makes no move. Otherwise, let U be the uniquely determined plane which contains both L' and P. ExtDrift makes the move prescribed by Drift in the plane U for P, L' and the projection of L onto U.

▶ **Theorem 2.** *Algorithm* ExtDrift *is 3-competitive for the line chasing problem in $\mathbb{R}^d$, for arbitrary dimension $d \geq 2$.*

Proof. We prove that (1) holds in arbitrary dimension. If $P \in L'$ then L and L' are co-planar, so the analysis from the previous section works directly.

So assume that $P \notin L'$. We first allow the adversary to perform a free move from its current position A to point $\ddot{A}$ defined as the orthogonal projection of A onto U, and then we analyze the move within U (that is, in a two-dimensional setting), as if the adversary started from point $\ddot{A}$.

We note that $\ell_{\ddot{A}X} \leq \ell_{AX}$ for any point $X \in U$, as $(\ell_{AX})^2 = (\ell_{\ddot{A}X})^2 + (\ell_{A\ddot{A}})^2$ by definition of $\ddot{A}$. It follows that:

- In the free adversary move from A to $\ddot{A}$ the potential function decreases (by taking $X = P$ in the above inequality) and both costs are 0. Further, in the move within U, with the adversary starting from $\ddot{A}$, Algorithm ExtDrift makes the same move as Drift, which implies that (1) is satisfied. Thus the complete move (combining the free adversary move and the move inside U) satisfies inequality (1) as well.
- The free move is only beneficial for the adversary: taking $X = A'$ shows that the cost of moving to A' from $\ddot{A}$ is no more costly for the adversary than moving to A' from A. ◀

4 Lower Bound for Memoryless Algorithms

We show that our algorithm achieves the optimal competitive ratio among a certain class of "memoryless" algorithms. For a metric space M, let $\mathcal{X} \subseteq \mathcal{P}(M)$ be the set of possible requests (i.e., lines in our case). In general, we can view an algorithm as a function $\mathcal{A}\colon M \times \mathcal{X}^* \to \mathcal{X}$ with $\mathcal{A}(P_0) = P_0$ and $\mathcal{A}(P_0, X_1, \ldots, X_t) \in X_t$ for each initial point $P_0 \in M$ and requests $X_1, \ldots, X_t \in \mathcal{X}$. We call an algorithm *memoryless* if $\mathcal{A}(P_0, X_1, \ldots, X_t)$ is a function of only the last position $\mathcal{A}(P_0, X_1, \ldots, X_{t-1})$, the last request X_{t-1} and the new request X_t.

However, memorylessness alone would not impose any limit on the power of line-chasing algorithms: By perturbing its positions very slightly, an algorithm could always encode the entire history in low significant bits of its current position. To get a meaningful notion of memorylessness, we therefore require an additional property, namely that the algorithm is oblivious with respect to rotation, translation or scaling of the metric space. More precisely, a *direct similarity* of $\mathbb{R}^d$ is a bijection $f\colon \mathbb{R}^d \to \mathbb{R}^d$ that is a composition of rotation, translation and scaling by some factor $r_f > 0$. In particular, for any $P, Q \in \mathbb{R}^d$, we have $\ell(f(P), f(Q)) = r_f \ell(P, Q)$. We call an algorithm $\mathcal{A}$ *rts-oblivious* if $\mathcal{A}(f(P_0), f(X_1), \ldots, f(X_t)) = f(\mathcal{A}(P_0, X_1, \ldots, X_t))$ for any $P_0 \in M$, $X_i \in \mathcal{X}$ and any direct similarity f. In general (when algorithms are allowed to use memory) there is no reason to behave differently when the input is transformed by such f, since it is just a renaming of points and scaling of distances by a uniform constant. For completeness, we provide a proof of this intuition via the following proposition:

▶ **Proposition 3.** *If there is a c-competitive algorithm for line-chasing, then there is a c-competitive rts-oblivious algorithm.*

Proof. For an initial position P_0 and request sequence $X_1, \ldots, X_t$, we assume without loss of generality that $P_0 \notin X_1$. For any such P_0 and X_1, there exists a unique direct similarity $g = g_{P_0 X_1}$ such that $g(P_0) = (0, 1)$ and $g(X_1) = \mathbb{R} \times \{0\}$. Given a c-competitive algorithm $\mathcal{A}$, we claim that the algorithm $\tilde{\mathcal{A}}$ given by

$$\tilde{\mathcal{A}}(P_0, X_1, \ldots, X_t) = g^{-1}(\mathcal{A}(g(P_0), g(X_1), \ldots, g(X_t)))$$

is rts-oblivious and c-competitive.

To see that $\tilde{\mathcal{A}}$ is rts-oblivious, consider an arbitrary direct similarity f. Notice that $g_{f(P_0)f(X_1)} = g \circ f^{-1}$. Thus,

$$\begin{aligned}\tilde{\mathcal{A}}(f(P_0), f(X_1), \ldots, f(X_t)) &= (f \circ g^{-1})(\mathcal{A}(g(P_0), g(X_1), \ldots, g(X_t)))\\ &= f(\tilde{\mathcal{A}}(P_0, X_1, \ldots, X_t)),\end{aligned}$$

as required. To see that $\tilde{\mathcal{A}}$ is c-competitive, consider an initial position P_0 and request sequence $X_1, \ldots, X_m$ along with an adversary's solution $A_0 = P_0, A_1 \in X_1, \ldots, A_m \in X_m$. The cost of $\tilde{\mathcal{A}}$ can be bounded via

$$\begin{aligned}&\sum_{t=1}^{m} \ell(\tilde{\mathcal{A}}(P_0, X_1, \ldots, X_{t-1}), \tilde{\mathcal{A}}(P_0, X_1, \ldots, X_t))\\ &\quad= \frac{1}{r_g} \sum_{t=1}^{m} \ell(\mathcal{A}(g(P_0), g(X_1), \ldots, g(X_{t-1})), \mathcal{A}(g(P_0), g(X_1), \ldots, g(X_t)))\\ &\quad\le \frac{c}{r_g} \sum_{t=1}^{m} \ell(g(A_{t-1}), g(A_t))\\ &\quad= c \sum_{t=1}^{m} \ell(A_{t-1}, A_t),\end{aligned}$$

where the inequality uses that $\mathcal{A}$ is c-competitive against the solution $g(A_0), \ldots, g(A_m)$ for the transformed input $g(P_0), g(X_1), \ldots, g(X_m)$. ◀

Intuitively, an rts-oblivious algorithm does not know the absolute coordinates of its positions and requests, but only relative to each other and up to scaling. If it is memoryless, in the plane this boils down to only knowing the angle between the new and the old request line. We show now that our algorithms DRIFT and EXTDRIFT achieve the optimal competitive ratio among rts-oblivious memoryless algorithms.

▶ **Theorem 4.** *Any rts-oblivious memoryless algorithm for line-chasing has competitive ratio at least* 3.

Proof. We will construct an initial point P_0 and lines $L_0, \ldots, L_m$ in $\mathbb{R}^2$ with the property that $P_0 \in L_0$ and L_t can be obtained by rotating L_{t-1} around some point $S_t \in L_{t-1}$ in clockwise direction by less than 90 degrees.

Let $P_0, \ldots, P_m$ be the sequence of points visited by a given algorithm. We use notation similar to that in Figure 1: Write $\bar{P}_{t-1}$ for the orthogonal projection of P_{t-1} onto L_t and let $h_t = \ell(P_{t-1}, \bar{P}_{t-1})$ and $s_t = \ell(\bar{P}_{t-1}, S_t)$. The movement from P_{t-1} to P_t can always be viewed as first moving to $\bar{P}_{t-1}$ and then moving some distance $x_t \in \mathbb{R}$ in the direction towards intersection S_t, for a total cost $\sqrt{h_t^2 + x_t^2}$. Here, $x_t < 0$ would constitute movement away from S_t and $x_t > s_t$ would constitute movement beyond S_t.

Observe that for rts-oblivious memoryless algorithms, $\frac{x_t}{h_t}$ is a function of only $\frac{h_t}{s_t}$, i.e. $\beta(\frac{h_t}{s_t}) = \frac{x_t}{h_t}$ for some function $\beta\colon (0, \infty) \to \mathbb{R}$. Any rts-oblivious memoryless algorithm for line-chasing in the plane is uniquely determined by its associated function β as well as similar functions for the cases of counter-clockwise rotations of at most 90 degrees and parallel lines.[1] Let $\beta(0) := \limsup_{a\to 0} \beta(a) \in \mathbb{R} \cup \{-\infty, \infty\}$. Let us first show that algorithms with $\beta(0) = \infty$ or $\beta(0) \le 0$ have unbounded competitive ratio.

[1] If we require algorithms to be oblivious also with respect to reflection (which would still satisfy Proposition 3), they would be uniquely determined by β alone. DRIFT is the algorithm corresponding to $\beta(a) = \frac{a+1-\sqrt{a^2+1}}{\sqrt{2}a}$.

If $\beta(0) = \infty$, we choose $P_0 = (1, h)$, $L_0 = \{(x, y) : y = hx\}$, $L_1 = \mathbb{R} \times \{0\}$ for some small $h > 0$. The algorithm's cost is $h\sqrt{1 + \beta(h)^2}$, whereas the optimal cost is h. Choosing h arbitrarily small shows that the competitive ratio is unbounded.

If $\beta(0) \leq 0$, fix some $\epsilon \in (0, \frac{1}{2}]$ and choose $a \in (0, \epsilon]$ with $\beta(a) \leq \epsilon$. Let $P_0 = (1, 0)$, $L_0 = \mathbb{R} \times \{0\}$ and define L_t as the clockwise rotation of L_{t-1} around the origin $O := (0, 0)$ by angle $\arctan(a)$. Thus, we have $\frac{h_t}{s_t} = a$ for each t. Notice that $s_t\sqrt{1 + a^2} = \ell(O, P_{t-1}) = s_{t-1}(1 - a\beta(a))$, and therefore

$$\frac{s_t}{s_{t-1}} = \frac{1 - a\beta(a)}{\sqrt{1 + a^2}} \geq 1 - \frac{a^2 + a\beta(a)}{1 + a^2} \geq 1 - 2\epsilon a,$$

where the first inequality uses $\sqrt{1 + a^2} \leq 1 + a^2$ and the second inequality uses $0 < a \leq \epsilon$ and $\beta(a) \leq \epsilon$. Hence,

$$s_t \geq (1 - 2\epsilon a)^{t-1}$$

Since $\ell(P_{t-1}, P_t) \geq h_t = as_t$, the total cost of the algorithm is

$$\sum_{t=1}^{m} \ell(P_{t-1}, P_t) \geq a \sum_{t=0}^{m-1} (1 - 2\epsilon a)^t \xrightarrow{m\to\infty} \frac{1}{2\epsilon}.$$

Meanwhile, an optimal algorithm pays total cost 1 by moving to O immediately. Letting $\epsilon \to 0$, we find again that the competitive ratio is unbounded.

It remains to consider the case $0 < \beta(0) < \infty$. Then we can choose arbitrarily small $a > 0$ such that $0 < a\beta(a) < 1$. We choose the initial point $P_0 = (\frac{1}{a}, 1)$, and the request sequence starts with $L_0 = \{(x, y) : y = ax\}$ and $L_1 = \mathbb{R} \times \{0\}$. For $t \geq 2$, we define L_t as the clockwise rotation of L_{t-1} around $S_t = S_2 = \left(\frac{1}{a} - \beta(a) + \sqrt{1 + a^2}\left(\beta(a) + \frac{1}{2\beta(a)}\right), 0\right)$ by angle $\arctan(a)$. The idea is that in response to L_1, the algorithm drifts to the left (towards intersection $S_1 = (0, 0)$), but the subsequent requests are such that it would have been cheaper to drift to the right (away from S_1) instead.

We have $s_1 = \frac{1}{a}$ and $s_2 = \frac{\ell(P_1,S_2)}{\sqrt{1+a^2}} = \beta(a) + \frac{1}{2\beta(a)}$. For $t \geq 3$, similarly to the previous case we get

$$\frac{s_t}{s_{t-1}} \geq 1 - \frac{a^2 + a\beta(a)}{1 + a^2} \geq 1 - a^2 - a\beta(a)$$

and therefore

$$s_t \geq \left(\beta(a) + \frac{1}{2\beta(a)}\right)\left(1 - a^2 - a\beta(a)\right)^{t-2} \qquad \text{if } t \geq 2.$$

As $m \to \infty$, the cost of the algorithm is

$$\begin{aligned}\sum_{t=1}^{\infty} \ell(P_{t-1}, P_t) &= \sum_{t=1}^{\infty} h_t\sqrt{1 + \beta(a)^2} = \sqrt{1 + \beta(a)^2}a\sum_{t=1}^{\infty} s_t \\ &\geq \sqrt{1 + \beta(a)^2}\left(1 + \left(\beta(a) + \frac{1}{2\beta(a)}\right)\frac{1}{a + \beta(a)}\right) \\ &\xrightarrow{a\to 0} \sqrt{1 + \beta(0)^2}\left(2 + \frac{1}{2\beta(0)^2}\right),\end{aligned}$$

where the limit $a \to 0$ is taken along a sequence where $\beta(a) \to \beta(0)$. In contrast, an offline algorithm can move immediately from P_0 to S_2, paying cost $\sqrt{1 + \frac{1}{4\beta(0)^2}}$ as $a \to 0$ and

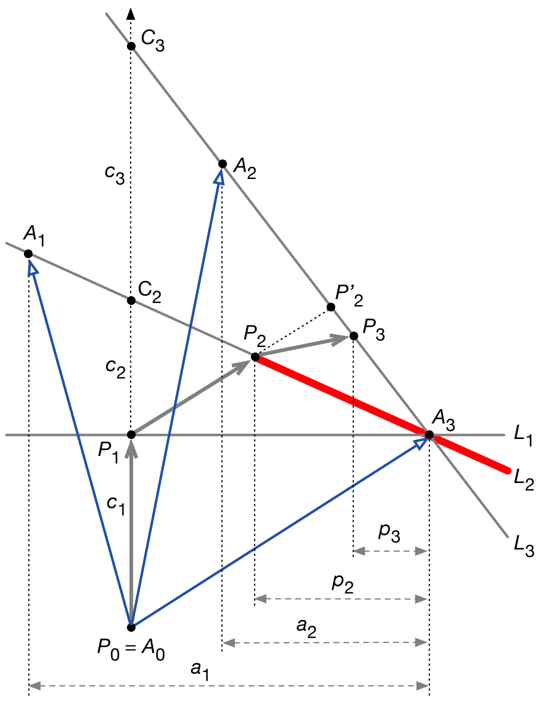

Figure 3 Visual description of our lower bound for arbitrary algorithms. Lines L_1, L_2 and L_3 are presented to an online algorithm. Blue arrows describe possible movements of OPT, while gray thick arrows describe a path of an algorithm that minimizes the competitive ratio for this adversarial construction. Red thick half-line denotes the forbidden region.

$\beta(a) \to \beta(0)$. By dividing, we see that the competitive ratio is at least

$$\sqrt{(1+\beta(0)^2)\left(4+\frac{1}{\beta(0)^2}\right)} = \sqrt{4\beta(0)^2+\frac{1}{\beta(0)^2}+5},$$

which is minimized for $\beta(0) = \frac{1}{\sqrt{2}}$, taking value 3. ◀

5 Lower Bound for Arbitrary Algorithms

Finally, in this section, we show how to improve an existing lower bound of $\sqrt{2} \approx 1.41$ for arbitrary algorithms to 1.5358. Our bound holds even in two dimensions, and improves also the lower bound for the more general convex body chasing in two dimensions.

▶ **Theorem 5.** *The competitive ratio of any deterministic online algorithm $\mathcal{A}$ for the line chasing problem is at least* 1.5358*.*

Proof. We describe our adversarial strategy below. On the created input, we will compare the cost of $\mathcal{A}$ to the cost of an offline optimum OPT. We assume that both $\mathcal{A}$ and OPT start at origin point $P_0 = A_0 = (0,0)$.

Our construction is parameterized with real positive numbers $c_1 = 0.5535$, $c_2 = 0.4965$, $c_3 = 0.8743$, $a_1 = 1.3012$, $a_2 = 0.6663$, $p_2 = 0.5612$, and $p_3 = 0.1696$.

We fix points $P_1 = (0, c_1)$, $C_2 = (0, c_1 + c_2)$, $C_3 = (0, c_1 + c_2 + c_3)$ and $A_3 = (1, c_1)$, see Figure 3 for illustration. For succinctness, we use notation $\triangle(x, y) = \sqrt{x^2 + y^2}$.

Initial part: Line L_1

The first request line is the line P_1A_3, denoted L_1. Without loss of generality, we can assume that $\mathcal{A}$ moves to point P_1. This is because the adversary can either play the strategy described below or its mirror image (flipped against the line P_0P_1), so any deviation from P_1, either to the left or right, can only increase the cost of $\mathcal{A}$.

From now on, for any point Q we denote its projection on line L_1 by Q^x.

Middle part: Line L_2

Next, the adversary issues the request line C_2A_3, denoted L_2. Let $P_2 \in L_2$ and $A_1 \in L_2$ be the points to the left of A_3, such that $\ell_{P_2^x A_3} = p_2$ and $\ell_{A_1^x A_3} = a_1$.

Let $\bar{P}_2$ be the point on L_2 chosen by $\mathcal{A}$. If $\bar{P}_2$ lies to the right of point P_2, then the adversary forces $\mathcal{A}$ to move to A_1 (by giving sufficiently many different lines that go through A_1 at different angles). Opt may then serve the whole sequence by going from A_0 to A_1 at cost

$$\ell_{A_0A_1} = \triangle(c_1 + c_2 \cdot a_1,\, a_1 - 1) \le 1.23679$$

while the cost of $\mathcal{A}$ is then at least

$$\begin{aligned}\ell_{P_0P_1} + \ell_{P_1P_2} + \ell_{P_2A_1} &= \ell_{P_0P_1} + \ell_{P_1P_2} + \ell_{A_3A_1} - \ell_{A_3P_2}\\ &= c_1 + \triangle(1-p_2,\, c_2\cdot p_2) + \triangle(a_1,\, c_2\cdot a_1) - \triangle(p_2,\, c_2\cdot p_2)\\ &\ge 1.89948\end{aligned}$$

Hence, the competitive ratio in this case is at least 1.5358.

We call the half-line of L_2 to the right of point P_2 *forbidden region*. From now on, we assume that the point chosen by $\mathcal{A}$ in L_2 does not lie in this region.

Final part: Line L_3

Finally, the adversary issues the request line C_3A_3, denoted L_3. Let P_2' be the intersection of line P_1P_2 with line L_3. Next, let A_2 and P_3 be the points on the line L_3 to the left of A_3, such that $\ell_{A_2^x A_3} = a_2$ and $\ell_{P_3^x A_3} = p_3$. Note that P_3 belongs to the interval $P_2'A_3$.

Let $\bar{P}_3$ be the point on L_3 chosen by $\mathcal{A}$. We consider two cases.

Case 1. $\bar{P}_3$ lies at point P_3 or to its left. In this case, the adversary forces $\mathcal{A}$ to move to A_3. Opt may serve the whole sequence by going from A_0 to A_3 paying

$$\ell_{A_0A_3} = \triangle(1,\, c_1) \le 1.142963.$$

We may now argue that the cost of $\mathcal{A}$ is minimized if $\bar{P}_3$ is equal to P_3: If $\bar{P}_3$ is to the left of point P_2', then the cost of $\mathcal{A}$ is at least $\ell_{P_0P_1} + \ell_{P_1\bar{P}_3} + \ell_{\bar{P}_3A_3}$. Both the second and the third summand decrease when we move $\bar{P}_3$ towards P_2'. Hence, now we may assume that $\bar{P}_3$ belongs to the interval $P_2'P_3$. As the path of $\mathcal{A}$ must avoid forbidden region, its cost is at least $\ell_{P_0P_1} + \ell_{P_1P_2} + \ell_{P_2\bar{P}_3} + \ell_{\bar{P}_3A_3}$. The sum of the last two summands decreases when we move $\bar{P}_3$ towards P_3. Therefore, we obtain that the cost of $\mathcal{A}$ is at least

$$\begin{aligned}\ell_{P_0P_1} + &\ell_{P_1P_2} + \ell_{P_2P_3} + \ell_{P_3A_3}\\ &= c_1 + \triangle(1-p_2,\, c_2\cdot p_2) + \triangle((c_2+c_3)\cdot p_3 - c_2\cdot p_2,\, p_2 - p_3)\\ &\qquad + \triangle(p_3,\, (c_2+c_3)\cdot p_3) \ge 1.75537.\end{aligned}$$

Thus, in this case the competitive ratio is at least 1.5358.

Case 2. If $\bar{P}_3$ lies to the right of point P_3, then the adversary forces $\mathcal{A}$ to move to A_2. Opt may serve the whole sequence by going from A_0 to A_2 at cost

$$\ell_{A_0A_2} = \triangle(c_1 + (c_2 + c_3) \cdot a_2,\, 1 - a_2) \le 1.50435.$$

To go from P_1 to $\bar{P}_3$ and avoid the forbidden region, $\mathcal{A}$ has to pay at least $\ell_{P_1P_2} + \ell_{P_2\bar{P}_3}$. Therefore, its cost is at least

$$\begin{aligned}\ell_{P_0P_1} + &\ell_{P_1P_2} + \ell_{P_2\bar{P}_3} + \ell_{\bar{P}_3A_2}\\ &\ge \ell_{P_0P_1} + \ell_{P_1P_2} + \ell_{P_2P_3} + \ell_{P_3A_2}\\ &\ge \ell_{P_0P_1} + \ell_{P_1P_2} + \ell_{P_2P_3} + \ell_{A_2A_3} - \ell_{P_3A_3}\\ &= c_1 + \triangle(1 - p_2,\, c_2 \cdot p_2) + \triangle((c_2 + c_3) \cdot p_3 - c_2 \cdot p_2,\, p_2 - p_3)\\ &\qquad + \triangle(a_2,\, (c_2 + c_3) \cdot a_2) - \triangle(p_3,\, (c_2 + c_3) \cdot p_3) \ge 2.31039.\end{aligned}$$

Thus, in this case the ratio is also at least 1.5358. ◀

6 Final Comments

Establishing the optimal competitive ratio for line chasing with memory remains an open problem. We believe that with memory, a competitive ratio better than 3 is achievable.

The intuition is that in the first move, if L and P are the initial line and position and L' is the new request line, then the algorithm should move to the nearest point $\bar{P}$ on L'. More generally, if the requests on L and L' alternate (and their angle is small), the algorithm should initially drift slowly towards $S = L \cap L'$ and only gradually accelerate as it becomes more credible that the adversary is located at S. To gauge this credibility for general request sequences, an algorithm might store the current work function at each step.

It appears also that our lower bound of 1.5358 can be improved by introducing additional steps, although this gives only very small improvements and leads to a very involved analysis. It is possible that an approach fundamentally different from ours may give a better bound with simpler analysis.

References

1 Antonios Antoniadis, Neal Barcelo, Michael Nugent, Kirk Pruhs, Kevin Schewior, and Michele Scquizzato. Chasing Convex Bodies and Functions. In *Proc. 12th Latin American Theoretical Informatics Symposium (LATIN)*, pages 68–81, 2016. doi:10.1007/978-3-662-49529-2_6.

2 C. J. Argue, Sébastien Bubeck, Michael B. Cohen, Anupam Gupta, and Yin Tat Lee. A Nearly-Linear Bound for Chasing Nested Convex Bodies. In *Proc. 30th ACM-SIAM Symp. on Discrete Algorithms (SODA)*, pages 117–122, 2019.

3 Nikhil Bansal, Martin Böhm, Marek Eliás, Grigorios Koumoutsos, and Seeun William Umboh. Nested Convex Bodies are Chaseable. In *Proc. 29th ACM-SIAM Symp. on Discrete Algorithms (SODA)*, pages 1253–1260, 2018. doi:10.1137/1.9781611975031.81.

4 Nikhil Bansal, Anupam Gupta, Ravishankar Krishnaswamy, Kirk Pruhs, Kevin Schewior, and Clifford Stein. A 2-Competitive Algorithm For Online Convex Optimization With Switching Costs. In *Proc. Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM)*, pages 96–109, 2015. doi:10.4230/LIPIcs.APPROX-RANDOM.2015.96.

5 Allan Borodin, Nathan Linial, and Michael E. Saks. An Optimal On-Line Algorithm for Metrical Task System. *J. ACM*, 39(4):745–763, 1992. doi:10.1145/146585.146588.

6 Sébastien Bubeck, Yin Tat Lee, Yuanzhi Li, and Mark Sellke. Chasing Nested Convex Bodies Nearly Optimally. *CoRR*, abs/1811.00999, 2018. URL: http://arxiv.org/abs/1811.00999.

7 Sébastien Bubeck, Yin Tat Lee, Yuanzhi Li, and Mark Sellke. Competitively chasing convex bodies. In *Proc. 51st ACM Symp. on Theory of Computing (STOC)*, pages 861–868, 2019. doi:10.1145/3313276.3316314.

8 William R. Burley. Traversing Layered Graphs Using the Work Function Algorithm. *J. Algorithms*, 20(3):479–511, 1996. doi:10.1006/jagm.1996.0024.

9 Marek Chrobak and Lawrence L. Larmore. Metrical Task Systems, the Server Problem and the Work Function Algorithm. In *Online Algorithms, The State of the Art (Proc. Dagstuhl Seminar, June 1996)*, pages 74–96, 1996. doi:10.1007/BFb0029565.

10 Amos Fiat, Dean P. Foster, Howard J. Karloff, Yuval Rabani, Yiftach Ravid, and Sundar Vishwanathan. Competitive Algorithms for Layered Graph Traversal. *SIAM J. Comput.*, 28(2):447–462, 1998. doi:10.1137/S0097539795279943.

11 Joel Friedman and Nathan Linial. On Convex Body Chasing. *Discrete & Computational Geometry*, 9:293–321, 1993. doi:10.1007/BF02189324.

12 Elias Koutsoupias and Christos H. Papadimitriou. On the k-Server Conjecture. *J. ACM*, 42(5):971–983, 1995. doi:10.1145/210118.210128.

13 Minghong Lin, Adam Wierman, Lachlan L. H. Andrew, and Eno Thereska. Dynamic Right-Sizing for Power-Proportional Data Centers. *IEEE/ACM Trans. Netw.*, 21(5):1378–1391, 2013. doi:10.1109/TNET.2012.2226216.

14 Mark S. Manasse, Lyle A. McGeoch, and Daniel Dominic Sleator. Competitive Algorithms for Server Problems. *J. Algorithms*, 11(2):208–230, 1990. doi:10.1016/0196-6774(90)90003-W.

15 H. Ramesh. On Traversing Layered Graphs On-Line. *J. Algorithms*, 18(3):480–512, 1995. doi:10.1006/jagm.1995.1019.

16 René Sitters. The Generalized Work Function Algorithm Is Competitive for the Generalized 2-Server Problem. *SIAM J. Comput.*, 43(1):96–125, 2014. doi:10.1137/120885309.