Nested Convex Bodies are Chaseable*

Nikhil Bansal[†‡] Martin Böhm[§] Marek Eliáš[†] Grigorios Koumoutsos[†]
Seeun William Umboh[†]

Abstract

In the Convex Body Chasing problem, we are given an initial point $v_0 \in \mathbb{R}^d$ and an online sequence of n convex bodies $F_1, \ldots, F_n$. When we receive F_i, we are required to move inside F_i. Our goal is to minimize the total distance traveled. This fundamental online problem was first studied by Friedman and Linial (DCG 1993). They proved an $\Omega(\sqrt{d})$ lower bound on the competitive ratio, and conjectured that a competitive ratio depending only on d is possible. However, despite much interest in the problem, the conjecture remains wide open.

We consider the setting in which the convex bodies are nested: $F_1 \supset \ldots \supset F_n$. The nested setting is closely related to extending the online LP framework of Buchbinder and Naor (ESA 2005) to arbitrary linear constraints. Moreover, this setting retains much of the difficulty of the general setting and captures an essential obstacle in resolving Friedman and Linial's conjecture. In this work, we give a $f(d)$-competitive algorithm for chasing nested convex bodies in $\mathbb{R}^d$.

1 Introduction

In the *convex body chasing* problem, introduced by Friedman and Linial [FL93], we are given an initial position $v_0 \in \mathbb{R}^d$. At each time step i, we receive a convex set $F_i \subset \mathbb{R}^d$ as a request, and to serve the request, we must move to some point $v_i \in F_i$. The goal is to minimize the total distance traveled to serve the requests. The distance can be measured using an arbitrary norm, but unless stated otherwise, it is measured using the Euclidean norm. As any convex body can be approximated arbitrarily well by intersection of halfspaces, one can assume that F_i are halfspaces[1] and hence this problem is also known as *halfspace chasing*.

This problem belongs to a very rich class of problems called *Metrical Service Systems* (MSS) [BLS92]. In an MSS, we are given an arbitrary metric space (V, ρ) and an initial position $v_0 \in V$. At each time i, a request set $F_i \subset V$ arrives and we must serve it by moving to some $v_i \in F_i$. MSS captures several interesting online problems such as the k-server problem. While almost tight bounds are known for general MSS [BLS92, FM00, BBM01], these bounds are not so interesting as typical online problems correspond to MSS with highly structured requests F_i and metric space (V, ρ). There has been a lot of interesting work on particular cases of MSS, e.g. [KP95, CL96, Bur96, BK04, SS06, Sit14], but understanding the role of structure in MSS instances is a major long-term goal in online computation with far-reaching consequences.

Indeed, the main motivation of [FL93] for considering the convex body chasing problem was to express the competitive ratio of MSS in terms of geometric properties of the request sets F_i. For the convex body chasing problem, they obtained an $O(1)$-competitive algorithm for $d = 2$; for $d > 2$, they gave an $\Omega(\sqrt{d})$ lower bound and conjectured that a competitive ratio depending only on d is possible. However, despite much interest, the conjecture remains open.

Nested Convex Body Chasing. In this paper, we consider the *nested convex body chasing* problem where the requested convex bodies are nested, i.e., $F_i \subset F_{i-1}$ for each $i \geq 1$. This natural special case is closely related to many fundamental questions in online algorithms and online learning, and has been of interest in recent years. However, prior to our work, nothing was known for it beyond the results of Friedman and Linial [FL93] for the general case.

1.1 Connections and Related Work

A useful equivalent formulation of the nested problem is the following: Given an initial position v_0, at each time step i, we receive some arbitrary convex body F_i (not necessarily nested), and we must move to some point v_i that is contained in *every* convex body seen so far, i.e. $v_i \in F_1 \cap \ldots \cap F_i$. The goal is to minimize the total distance traveled. Indeed, this is equivalent to convex body chasing with requests $F'_i = F_1 \cap \ldots \cap F_i$, which form a nested sequence.

---

*This work was supported by NWO grant 639.022.211, ERC consolidator grant 617951, and GAČR grant P202/12/G061.

[†]TU Eindhoven, Netherlands.
{n.bansal,m.elias,g.koumoutsos}@tue.nl,
seeun.umboh@gmail.com

[‡]Centrum Wiskunde & Informatica, Netherlands.

[§]Charles University, Czech Republic. bohm@iuuk.mff.cuni.cz

[1]If F is the intersection of halfspaces $H_1, \ldots, H_s$, to simulate the request for F, the adversary can give $H_1, \ldots, H_s$ several times in a round-robin manner until the online algorithm moves inside F. Not revealing F directly can only hurt the online algorithm and does not affect the offline solution.

Online Covering LP. The influential primal-dual framework of Buchbinder and Naor for online covering LPs [BN09] can now be seen as a special case of nested convex body chasing with the ℓ_1-norm. In the former problem, the algorithm starts at the origin $v_0 = \mathbf{0}$, and at each time i, a linear constraint $a_i^\top x \geq b_i$ with non-negative a_i and b_i arrives. The goal is to maintain a feasible point x_i that satisfies all previous requests while the coordinates of x_i are only allowed to increase over time. The objective function $c^\top x$ (where c is non-negative) can be assumed to be $\|x\|_1$ by scaling. Finally, note that in nested convex body chasing with covering constraints, it never helps to decrease any variable and hence online covering LP is indeed a special case of nested convex body chasing.

While the online primal-dual framework [BN09] has been applied successfully to many online problems, so far it is limited to LPs with covering and packing constraints, and minor tweaks thereof. An important goal is to extend the online LP framework more broadly beyond packing and covering LPs. For example, it is unclear how to do this even for seemingly simple formulations such as k-server on depth-2 HSTs or Metrical Task Systems on a line. Since the nested convex body chasing problem corresponds to solving online LPs with arbitrary constraints (with both positive and negative entries) and a specific type of objective, understanding the nested convex body chasing problem is an essential step towards this goal. Indeed, this is one of our main motivations to consider this problem.

General Convex Body Chasing. Another motivation for studying the nested case is that it captures much of the inherent hardness of the general convex body chasing problem. For example, the $\Omega(\sqrt{d})$ lower bound [FL93] for the general problem also holds in the nested setting. Moreover, several natural algorithms also fail for the nested case.

Other Special Cases. The only known algorithms for chasing convex bodies in $\mathbb{R}^d$ with $d > 2$ are for certain restricted families of convex bodies F_i such as lines and affine subspaces. For chasing lines, Friedman and Linial [FL93] gave an $O(1)$-competitive algorithm. For chasing lines and half-line, Sitters [Sit14] showed that the generalized work function algorithm (WFA) is also $O(1)$-competitive; this is interesting as the WFA is a generic algorithm that attains nearly-optimal competitive ratios for many MSS and is a natural candidate to be $f(d)$-competitive for convex body chasing. Recently, Antoniadis et al. [ABN+16] gave an elegant and simple $O(1)$-competitive algorithm for chasing lines, and a $2^{O(d)}$-competitive algorithm for chasing affine subspaces. However, all these results crucially rely on the fact that the requests F_i have a lower dimension and do not seem to apply to our problem.

Connections to Online Learning. The convex body chasing problem is also closely related to recent work combining aspects of competitive analysis and online learning. One such work is the *Smoothed Online Convex Optimization* setting of Andrew et al. [ABL+13, ABL+15] which incorporates movement cost into the well-studied online learning setting of online convex optimization. The problem is well-understood for $d = 1$ [BGK+15, AS17], but nothing is known for larger d. Another related work is that of Buchbinder, Chen and Naor [BCN14] which combines online covering LPs with movement cost.

1.2 Our Results

Our main result is the following.

THEOREM 1.1. *There is an algorithm for chasing nested convex bodies in $\mathbb{R}^d$ with competitive ratio that only depends on d. In particular, it has competitive ratio $O(6^d(d!)^2)$.*

The algorithm is described in Section 3 and is based on two ideas. First we show that to design an $O_d(1)$-competitive algorithm for chasing nested convex bodies, it suffices to make an algorithm for r-bounded instances, where all the bodies F_i are completely contained in some ball $B(v, r)$ with radius r and center v. Moreover, even though competitive ratio is a relative guarantee, it suffices to bound the total movement of the algorithm on any r-bounded instance by $O_d(r)$. Proving an absolute bound on the distance moved makes the algorithmic task easier and we design such a bounded chasing algorithm in Section 3.1.

Surprisingly, the natural approaches for r-bounded instances based on the Ellipsoid Method or the centroid approach do not work. In particular, consider a 1-bounded instance where the initial body is $F_1 = B(\mathbf{0}, 1)$, and the algorithm starts at the origin. As nested convex bodies arrive, if the current point v_{i-1} is infeasible for the request F_i, a natural approach might be to move to the centroid of F_i or to the center of the minimum volume ellipsoid enclosing F_i (see Figure 1). In Section 4, we describe a simple 1-bounded instance in $\mathbb{R}^2$ on which the above algorithms travel an unbounded distance.

We design our d-dimensional bounded chasing algorithm in Section 3 based on a recursive approach together with some simple geometric properties. It iteratively invokes the $(d-1)$-dimensional algorithm on at most d bounded instances defined on some suitably chosen hyperplanes. When these instances end, we can argue that the future requests must lie in some smaller ball $B(v', \gamma r)$, for some fixed $\gamma < 1$. Roughly, this al-

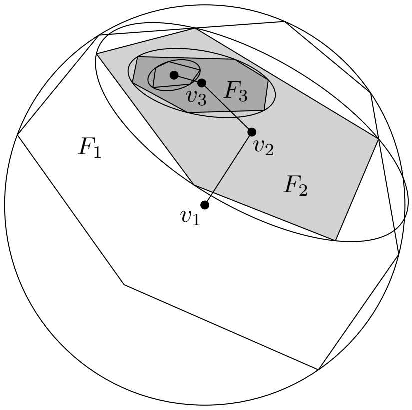

Figure 1: A few steps of the ellipsoid algorithm.

lows us to bound the competitive ratio by $g(d)$, that satisfies the recurrence $g(d) \le dg(d-1)/(1-\gamma)$.

2 Preliminaries

We define some notation and recall some basic facts from geometry.

Definition 2.1 (Nested Convex Body Chasing) *In the* nested convex body chasing problem *in $\mathbb{R}^d$, the algorithm starts at some position v_0, and an online sequence of n nested convex bodies $F_1 \supset \ldots \supset F_n$ arrive one by one. When convex body F_i arrives, the algorithm must move to a point v_i that lies in F_i. The goal is to minimize the total distance traveled $\sum_{i=1}^n \|v_i - v_{i-1}\|_2$.*

Note that the choice of measuring distance using the ℓ_2-norm—as opposed to some other symmetric norm, say the ℓ_1-norm—has a negligible effect on the competitive ratio that we obtain because all symmetric norms on $\mathbb{R}^d$ are within a $d^{1/2}$ factor of each other.

Let $B(v, r)$ denote the ball of radius r centered at v. The following useful fact is a variant of John's theorem for balls instead of ellipsoids (for a proof, see e.g. [MG07, Lemma 8.7.3]):

Proposition 2.1 (Minimum-volume enclosing ball) *Let F be a bounded convex body and suppose $B(v, r)$ is the minimum-volume ball enclosing F. Then, the center v of the ball $B(v, r)$ is contained in F.*

Next we need the following standard fact that we prove here for completeness. We will use it to show that either we can reduce to a $(d-1)$-dimensional instance or a d-dimensional instance that is contained in a ball with smaller radius. We use $\mathbf{0}$ to denote the origin.

Proposition 2.2 (Dimension reduction or radius reduction) *Let $d \ge 2$ and F be a bounded convex body in $\mathbb{R}^d$ contained in $B(\mathbf{0}, r)$. Then, either F intersects some axis-aligned hyperplane, or it is contained in some orthant of $B(\mathbf{0}, r)$. Moreover, in the latter case, the smallest ball $B(s, r')$ enclosing F has radius $r' \le r(1 - 1/d)^{1/2}$.*

Proof. The first part immediately follows by convexity. For the second part, we assume, without loss of generality, that $r = 1$ and that F is contained in the positive orthant of $B(\mathbf{0}, 1)$. We now show that every point x in the positive orthant of $B(\mathbf{0}, 1)$ is within distance at most $r' := (1 - 1/d)^{1/2}$ from the point $s = (1/d, \ldots, 1/d)$. There are two cases: (1) $\|x\|_1 \le 1$; (2) $\|x\|_1 > 1$.

In case (1), x is in the convex hull of $e_1, \ldots, e_d$, and $\mathbf{0}$, where e_k denotes the k-th vector of the standard basis with 1 in the k-th coordinate and 0 elsewhere. Therefore, it suffices to prove that $\mathbf{0}$ and $e_1, \ldots, e_d$ are within distance r' from s. Indeed, $\|s - \mathbf{0}\|_2 = (1/d)^{1/2} \le r'$ (as $d \ge 2$) and $\|s - e_k\|_2 = r'$ for each $k \in [d]$.

In case (2), we have

$$\|s - x\|_2^2 = \sum_{k=1}^d \left(x_k - \frac{1}{d}\right)^2 = \sum_{k=1}^d \left(x_k^2 - \frac{2x_k}{d} + \frac{1}{d^2}\right)$$
$$= \|x\|_2^2 - \frac{2\|x\|_1}{d} + \frac{1}{d} \le 1 - 1/d,$$

where the inequality uses that $x \in B(\mathbf{0}, 1)$ and hence $\|x\|_2 \le 1$ and $\|x\|_1 > 1$. Thus, the positive orthant of $B(\mathbf{0}, 1)$ is contained in $B(s, r(1 - 1/d)^{1/2})$. □

3 Algorithm

We now describe our algorithm and prove Theorem 1.1.

We first show, using a guess-and-double approach, that finding a good algorithm for the general nested convex body chasing problem can be reduced to an easier problem of designing an algorithm for which we can upper bound the *absolute* distance traveled, on bounded instances of the following type.

Definition 3.1 (r-Bounded Instances) *An instance with starting point v_0 and requests $F_1 \supset \ldots \supset F_n$ is said to be* r-bounded *if every request F_i is contained in $B(v_0, r)$.*

Note that a general instance may not be r-bounded for any finite r, e.g. in a covering LP where all the F_i are halfspaces of the type $a_i^\top x \ge 1$ where a_i has all entries non-negative.

As we shall see, the task of showing an absolute bound on the distance traveled (instead of a relative bound needed for competitive ratio) makes the problem cleaner. We now describe the reduction.

Lemma 3.1 (Reduction to Bounded Chasing) *For any fixed $r > 0$, if there exists an algorithm* Chase_d

for r-bounded instances that travels a total distance of at most $g(d) \cdot r$, then there exists an $f(d)$-competitive algorithm for general instances with $f(d) = 8g(d)$.

Proof. Consider a general instance with starting point v_0. Let δ_i be the distance between v_0 and the closest point in F_i; note that δ_i is non-decreasing in i because F_i's are nested. Wlog, we can assume that $v_0 \notin F_1$ and $\delta_1 = 1$ (by scaling).

The algorithm for the general instance proceeds in *stages*. For $j = 1, 2, \ldots$, stage j consists of all requests F_i for which $\delta_i \in [2^{j-1}, 2^j)$, i.e., stage j begins with the first request F_i that intersects with $B(v_0, 2^j)$ but not with $B(v_0, 2^{j-1})$, and ends with the last request $F_{i'}$ that intersects with $B(v_0, 2^j)$.

The algorithm will run a new instance of Chase_d at each stage j. Let $F_{s(j)}$ be the first request of stage j and $F_{\ell(j)}$ be the last. At the start of stage j, the algorithm starts at the point v_0, and begins an instantiation $\text{Chase}_d(v_0, 2^j)$ of Chase_d that it runs over the course of the stage with requests

$$F_{s(j)} \cap B(v_0, 2^j), F_{s(j)+1} \cap B(v_0, 2^j), \ldots, F_{\ell(j)} \cap B(v_0, 2^j).$$

Note that these requests form a 2^j-bounded instance.

We now bound the performance of the algorithm. Clearly, $\text{OPT} = \delta_n$. Let j^* denote the index of the final stage, and hence $\text{OPT} \geq 2^{j^*-1}$. For each $1 \leq j \leq j^*$, the movement cost during stage j has two parts and can be bounded as follows:

- The movement of $\text{Chase}_d(v_0, 2^j)$. This is at most $g(d) \cdot 2^j$ by the assumed guarantee on Chase_d.
- Returning to v_0 from its previous location, just before stage j begins. This cost is at most $2^{j'} \leq 2^{j-1}$ where $j' < j$ is the stage that ends just before stage j. This is because the algorithm was following $\text{Chase}_d(v_0, 2^{j'})$ which always stays within $B(v_0, 2^{j'})$ (as all $\text{Chase}_d(v_0, r)$ requests lie in $B(v_0, r)$).

So the total distance traveled by our algorithm is at most

$$\sum_{j \leq j^*} 2^j (g(d) + 1) \leq 2^{j^*+1}(g(d) + 1) \leq 4(g(d) + 1)\,\text{OPT}\,.$$

The lemma now follows from the fact that $g(d) \geq 1$ as the algorithm might need to travel a distance of r. ☐

3.1 Bounded Chasing Algorithm We now focus on designing an algorithm for the Bounded Chasing problem. The following theorem is our main technical result.

Theorem 3.1 (Bounded Chasing Theorem) *There exists an algorithm* Chase_d *that travels at most $g(d) \cdot r$ distance on r-bounded instances where $g(d) = 6^d (d!)^2$.*

Before we prove Theorem 3.1, let us note that Theorem 1.1 immediately follows by combining Theorem 3.1 and Lemma 3.1.

We now construct the algorithm Chase_d and prove Theorem 3.1. The proof is by induction on d. The base case ($d = 1$) is trivial: the requests form nested intervals and the greedy algorithm that always moves to the closest feasible point is 1-competitive, so $g(d) = 1$. In the remainder of this section, we focus on the $d \geq 2$ case and assume that there exists a $(d-1)$-dimensional algorithm Chase_{d-1} with the required properties.

Algorithm. Consider an r_0-bounded instance with starting point s_0. The high level idea of the algorithm is to reduce the instance into a sequence of $(d-1)$-dimensional instances and run Chase_{d-1} on these instances.

The algorithm runs in phases. Each phase starts at some center s with radius parameter $r \leq r_0$. The first phase starts at $s = s_0$ with radius $r = r_0$. In each phase, we run Chase_{d-1} with center s and radius r on the $(d-1)$-dimensional instances induced by the d axis-aligned hyperplanes $H_1, \ldots, H_d$ containing s. These are called *hyperplane steps*. When some request F_i arrives that does not intersect with any of these hyperplanes $H_1, \ldots, H_d$, we perform a *recentering step* by computing the smallest ball $B(s', r')$ enclosing F_i and moving to s'; the current phase then ends, and a new phase starts with center s' and radius r'. A key property we will use in the analysis (based on Proposition 2.2) is that $r' \leq (1 - 1/d)^{1/2} r$, which will allow us to argue that the algorithm makes progress.

Description of a phase. We now describe how a phase works. The reader may find it helpful to refer to Figure 2 while reading the description below.

Consider a phase that starts at center s and radius r. For notational convenience, we reindex the requests so that the first request of the phase is F_1. Let $H_1, \ldots, H_d$ denote the axis-aligned hyperplanes passing through s.

Hyperplane steps. Initially at request F_1, we choose the axis-aligned hyperplane H_k with the smallest index $k \in [d]$ that intersects F_1 (if no such hyperplane exists, we move to the Recentering step below), and run Chase_{d-1} on the $(d-1)$-dimensional instance induced by H_k and follow it for as long as we can. More specifically, we run Chase_{d-1} on the $(d-1)$-dimensional instance with starting point s and radius r, and requests

$$F_1 \cap H_k, \ldots, F_{\ell(k)} \cap H_k,$$

where $F_{\ell(k)}$ is the last request in the current phase that intersects H_k; for $i \leq \ell(k)$, we serve request F_i by

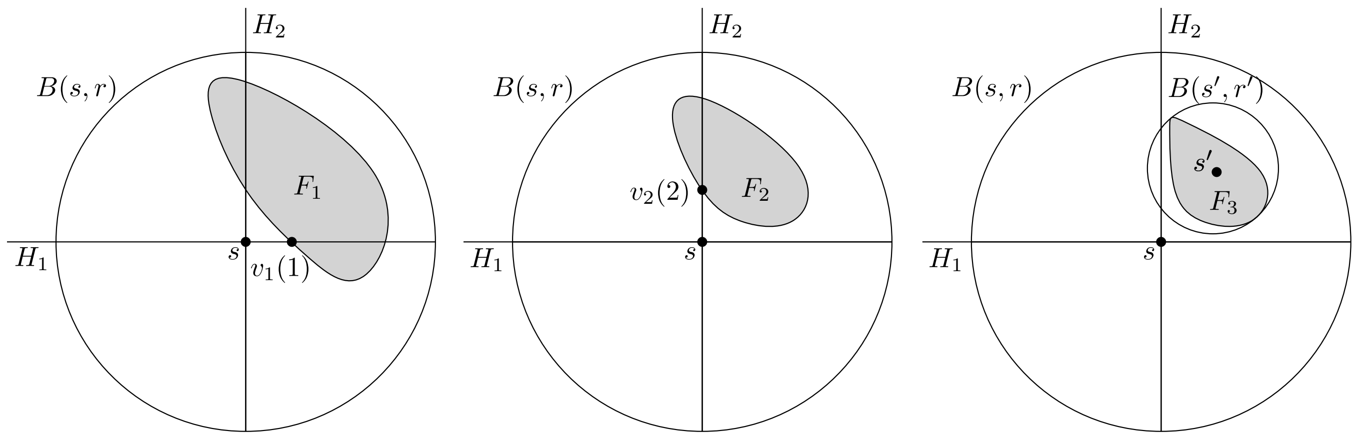

Figure 2: A phase of Chase_2, that starts at s. The first request F_1 is served using Chase_1 in H_1. The second request F_2 does not intersect H_1 so it is served using Chase_1 in H_2. Finally, F_3 does not intersect H_2 either and a recentering step is performed.

moving to $v_k(i)$ where $v_k(i)$ is the location of Chase_{d-1} on request $F_i \cap H_k$.

When the first request F_i arrives that does not intersect the current H_k, i.e., $F_i = F_{\ell(k)+1}$, then we change the hyperplane and repeat the above process. That is, we pick $H_{k'}$ that intersects F_i, with the smallest index k' (if it exists), and run Chase_{d-1} on $H_{k'}$ starting at position s with radius r and requests $F_{\ell(k)+1} \cap H_{k'}, \ldots$ and follow it for as long as we can.

Recentering step. If a request F_i arrives that does not intersect any H_k for $k \in [d]$, we compute the smallest ball $B(s', r')$ containing F_i, move to s' and serve F_i (note that $s' \in F_i$ by Proposition 2.1). The current phase ends, and a new phase with center s' and radius r' starts.

This completes the description of a phase and we now turn to analyzing the algorithm.

Analysis. We need to show that Chase_d is always feasible (Claim 3.1) and bound the distance it travels (Claim 3.2). These claims give us Theorem 3.1.

Claim 3.1 (Feasibility) *The algorithm* Chase_d *is always feasible, i.e.* $v_i \in F_i$ *for all* i.

Proof. We prove the claim by induction on d. For $d = 1$, the algorithm is trivial and it is always feasible. Assume that the claim is true for $d-1$. Consider some request F_i. Observe that Chase_d either performs a hyperplane step or a recentering step. In the former, since we follow Chase_{d-1} on some hyperplane H_k and Chase_{d-1} stays feasible (by induction), so we have that $v_i \in F_i \cap H_k$. In the latter, v_i is the center of the smallest ball containing F_i so $v_i \in F_i$ by Proposition 2.1. Thus, in both cases, $v_i \in F_i$. □

Next, we bound the distance traveled by Chase_d.

Claim 3.2 (Cost) *The total distance traveled by* Chase_d *on an* r*-bounded instance is at most* $g(d) \cdot r$, *where* $g(d) = 6^d (d!)^2$.

Proof. We now bound the distance traveled during each phase. Consider phase j. Let B_j denote Chase_d's enclosing ball during the phase and r_j be its radius. Note that during the phase, the algorithm stays within B_j and all requests F_i in the phase are contained in B_j. The movement in phase j consists of:

- Movement due to hyperplane steps. On each hyperplane H_k, we move at most $g(d-1) \cdot r_j$ by following Chase_{d-1}. Thus, the total movement due to hyperplane steps is at most $d \cdot g(d-1) \cdot r_j$
- Movement due to switching hyperplanes. We switch hyperplanes at most $d-1$ times, so this is at most $(d-1) \cdot 2r_j$.
- Movement due to recentering. This is at most $2r_j$.

Thus, the total distance traveled in phase j is at most

$$d \cdot (g(d-1) + 2) \cdot r_j \leq 3d \cdot g(d-1) \cdot r_j,$$

since $g(d-1) \geq 1$.

By Proposition 2.1, the radii of the enclosing balls decrease geometrically across phases: $r_j \leq r_{j-1}(1 - 1/d)^{1/2}$ for $j > 1$. As $r_1 = r$, the total distance traveled by Chase_d over all phases is at most

$$3d \cdot g(d-1) \cdot \frac{r}{1-(1-1/d)^{1/2}} \leq 3d \cdot g(d-1) \cdot 2dr = 6d^2 g(d-1) r,$$

where we use that $1-(1-1/d)^{1/2} \geq 1/(2d)$, as $(1-x)^\alpha \leq 1-\alpha x$ for any $x \in [0,1]$ and $\alpha < 1$.

Thus, we conclude that Chase_d travels at most $g(d) \cdot r$, where $g(d) = 6^d (d!)^2$ is the solution to the recurrence relation $g(d) = 6d^2 g(d-1)$ with base case $g(1) = 1$. □

4 Lower Bounds for Ellipsoid and Centroid

In this section, we consider some natural ellipsoid-based and centroid-based algorithms for chasing nested bodies in the r-bounded setting, and show that they are not competitive. The main reason these algorithms fail is that for (relatively) flat convex bodies, the center of the bounding ellipsoid, or the centroid, can move by a large distance in directions that do not matter.

Henceforth, for a set $S \subset \mathbb{R}^d$, let $E(S)$ denote the smallest-volume ellipsoid containing S.

Algorithm 1 An ellipsoid-based algorithm

Let F_t be the current bounded convex body on input.
Whenever the current position becomes infeasible:
Move to the center of $E(F_t)$.

We now construct an $\mathbb{R}^2$ instance in which Algorithm 1 travels an arbitrarily large distance while the optimal offline cost is constant. In the following, we will use the notation (x, y) for a point in $\mathbb{R}^2$.

The starting point of the instance is $(0, 1)$. Each request F_t is an intersection of four halfspaces A, B, C, H_t. The first three halfspaces A, B, C are $y \geq 0, x \geq -1$, and $x \leq 1$, respectively. The last halfspace H_t will be different for each F_t.

For the first request F_1, we set

$$H_1 = \{(x, y) \mid 2y \leq (1-\alpha)x + (1+\alpha)\},$$

for some parameter α. Note that the boundary of H_1 passes through the points $(-1, \alpha)$ and $(1, 1)$, as seen in Figure 3. The parameter α is chosen so that the center of $E(F_1)$ is strictly to the right of the y axis, as guaranteed by the following lemma:

LEMMA 4.1. *There exists $0 < \alpha < 1$ such that the center of the smallest ellipsoid containing F_1 has a strictly positive x-coordinate. More precisely, its center is (c, b) with $c, b > 0$.*

We postpone the proof of Lemma 4.1 and continue with the description of the request sequence. The remaining nested bodies $\{F_t \mid t \geq 2\}$ are created so that the x-coordinate of the center of $E(F_t)$ oscillates between c and $-c$. To this end, we construct two infinite families of halfspaces R_i and L_i: for $i \geq 0$, we define

$$R_i := \{(x, y) \mid 2y \leq (\alpha^{2i} - \alpha^{2i+1})x + (\alpha^{2i} + \alpha^{2i+1})\}$$
$$L_i := \{(x, y) \mid 2y \leq (\alpha^{2i+2} - \alpha^{2i+1})x + (\alpha^{2i+2} + \alpha^{2i+1})\}.$$

Observe that the boundary of R_i passes through the points $(1, \alpha^{2i})$ and $(-1, \alpha^{2i+1})$, and the boundary of L_i passes through $(-1, \alpha^{2i+1})$ and $(1, \alpha^{2i+2})$. See Figure 4 for an illustration.

We now describe the requests F_t for $t \geq 2$. For even t, we set H_t to be L_i, where i is the smallest index such that L_i does not contain the current position of the algorithm. For odd t, we select H_t to be R_i in a similar fashion. This completes our description of the requests F_t. Looking at Figure 4, one can easily observe that our requests F_t are indeed nested.

The following lemma describes the position of the center of each ellipsoid $E(F_t)$. Note that c and b are the strictly positive constants from Lemma 4.1.

LEMMA 4.2. *When t is odd, the center of $E(F_t)$ is $(c, b\alpha^{2i})$ for some i. When t is even, the center of $E(F_t)$ is $(-c, b\alpha^{2i+1})$ for some i.*

Proof. Recall that $F_t = A \cap B \cap C \cap H_t$. First, consider odd t, where $H_t = R_i$. We define a map $f\colon (x, y) \to (x, y/\alpha^{2i})$, which rescales the y-coordinate. Note that it maps F_t to F_1. Moreover, f preserves ratios between volumes, and therefore the map of the smallest ellipsoid containing F_t is the smallest ellipsoid containing F_1. We know that its center is at (c, b), and therefore the center of $E(F_t)$ is at $(c, b\alpha^{2i})$.

For t even, we have $H_t = L_i$ for some i. We define $g\colon (x, y) \to (-x, y/\alpha^{2i+1})$, which first mirrors F_t with respect to the y axis, and then rescales the y-coordinate, so that $g(F_t) = F_1$. Clearly, mirroring preserves the volumes, while rescaling preserves their ratios. Therefore, f maps $E(F_t)$ to $E(F_1)$ whose center is at (c, b) and the center of $E(F_t)$ is at $(-c, b\alpha^{2i+1})$. □

Let us now estimate the competitive ratio of Algorithm 1. At each time step, it incurs cost at least $2c$, since it moves between two points with x-coordinates c and $-c$ respectively. Therefore, if N is the total number of requests, the total cost incurred by Algorithm 1 is at least $N \cdot 2c$, which can be arbitrary large. On the other hand, the point $(0, 0)$ is contained in every F_t, since it belongs to F_1 and also to every halfspace R_i and L_i. Therefore, the cost of OPT is at most 1 and the competitive ratio of Algorithm 1 is unbounded.

Proof. [Proof of Lemma 4.1] Using a computer algebra system, we computed that for $\alpha = 1/2$ the center of $E(F_1)$ is in $(0.24568, 0.40571)$. This can be calculated, e.g., using the function `ellipsoidhull` in R, but similar functions are also available for Matlab. This shows that $\alpha = 1/2$ satisfies the requirements of the lemma.

The manual computation of $E(F_1)$ for $\alpha = 1/2$ is laborious, but we can still give a formal proof of the

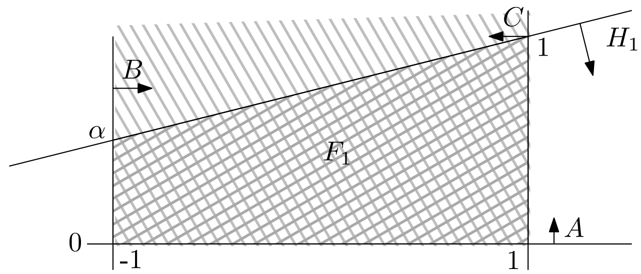

Figure 3: The first request F_1.

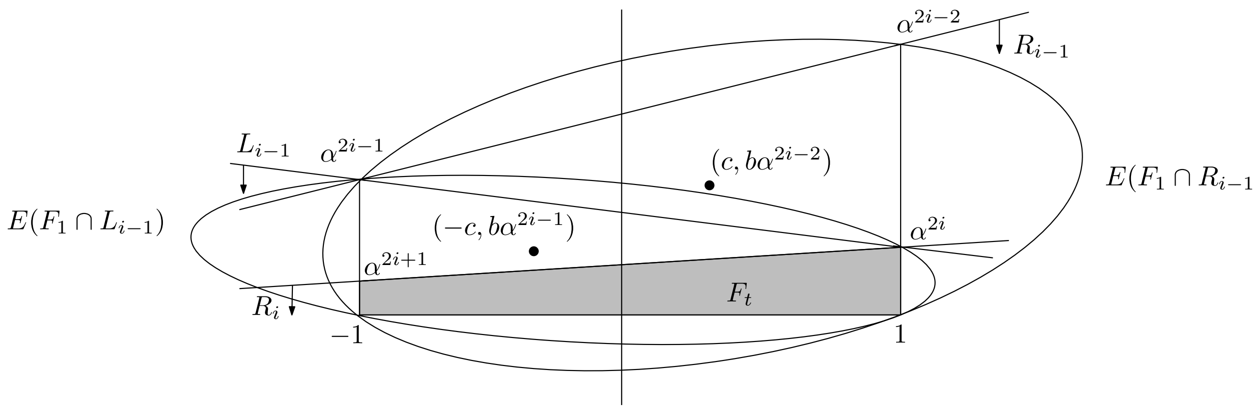

Figure 4: Halfspaces L_i and R_i

existence of a suitable α. Let $F_1(\alpha)$ denote F_1 with parameter α. Observe that $F_1(0)$ is a triangle with vertices $(1, 0), (1, 1)$, and $(-1, 0)$; and $F_1(1)$ is a square with vertices $(1, 0), (1, 1)$, $(-1, 0)$ and $(-1, 1)$. Since the center of $E(F_1(\alpha))$ evolves continuously with α, it suffices to show that the center of the smallest ellipsoid containing the triangle $F_1(0)$ lies strictly to the right of the y-axis. By continuity, this implies that there exists $\alpha > 0$ such that the x-coordinate of the center of $E(F_1(\alpha))$ is still strictly positive.

We define the affine map $f\colon (x, y) \to \begin{pmatrix} 1 & -1 \\ 0 & \sqrt{3} \end{pmatrix} (x, y)^\top$. Let $T = F_1(\alpha)$. This transformation makes T equilateral by first shearing it to the left to be symmetric with respect to the y-axis and then shrinking the y-coordinate. The smallest ellipsoid containing an equilateral triangle is its circumcircle, whose center lies in the intersection of its altitudes. Since one of the altitudes lies on the y-axis, the x-coordinate of the center of $E(f(T))$ is 0, and its y-coordinate is strictly positive. Since f preserves ratios between volumes, we have $f(E(T)) = E(f(T))$. Therefore, applying f^{-1} to the center of $E(f(T))$, we know that the center of $E(T)$ has both coordinates strictly positive. $\square$

Lower bound for the centroid algorithm. Similar to the Ellipsoid-based algorithm, one can propose an algorithm that moves to the centroid (center of mass) instead:

Algorithm 2 A centroid-based algorithm

Whenever the current position becomes infeasible:
    Move to the centroid of F_t.

The same requests F_t as above also shows that this algorithm is not competitive either. In fact, the analysis here is much easier, as we can compute the centroids using simple geometry (the input convex bodies can be partitioned into a right triangle and a rectangle, as seen e.g. in Figure 3).

A simple calculation shows that for $\alpha = 1/2$, the centroid of F_1 is $(1/9, 7/9)$. For the convex bodies requested later, the x-coordinate of the centroid will oscillate between $-1/9$ and $1/9$, again showing that the total distance traveled by the algorithm can be made arbitrarily large.

5 Concluding Remarks

After the initial announcement of this work, Sébastien Bubeck pointed out to us that the greedy algorithm (which moves at each time to the closest feasible point from the current location) is also $d^{O(d)}$-competitive, and that this can be shown using the results about gradient flows due to Manselli and Pucci [MP91].

Acknowledgments

We would like to thank Sébastien Bubeck, Niv Buchbinder, Anupam Gupta, Guru Guruganesh, Cristóbal Guzmán, and René Sitters for several interesting discussions.

References

[ABL$^+$13] Lachlan L. H. Andrew, Siddharth Barman, Katrina Ligett, Minghong Lin, Adam Meyerson, Alan Roytman, and Adam Wierman. A tale of two metrics: Simultaneous bounds on competitiveness and regret. In *COLT*, pages 741–763, 2013.

[ABL$^+$15] Lachlan L. H. Andrew, Siddharth Barman, Katrina Ligett, Minghong Lin, Adam Meyerson, Alan Roytman, and Adam Wierman. A tale of two metrics: Simultaneous bounds on competitiveness and regret. *CoRR*, arXiv:1508.03769, 2015.

[ABN$^+$16] Antonios Antoniadis, Neal Barcelo, Michael Nugent, Kirk Pruhs, Kevin Schewior, and Michele Scquizzato. Chasing convex bodies and functions. In *LATIN*, pages 68–81, 2016.

[AS17] Antonios Antoniadis and Kevin Schewior. A tight lower bound for online convex optimization with switching costs. In *Submitted, June 2017*, 2017.

[BBM01] Yair Bartal, Béla Bollobás, and Manor Mendel. A Ramsey-type theorem for metric spaces and its applications for metrical task systems and related problems. In *Foundations of Computer Science, FOCS*, pages 396–405, 2001.

[BCN14] Niv Buchbinder, Shahar Chen, and Joseph Naor. Competitive analysis via regularization. In *SODA*, pages 436–444, 2014.

[BGK$^+$15] 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 *APPROX/RANDOM*, pages 96–109, 2015.

[BK04] Yair Bartal and Elias Koutsoupias. On the competitive ratio of the work function algorithm for the k-server problem. *Theoretical Computer Science*, 324(2):337 – 345, 2004.

[BLS92] Allan Borodin, Nathan Linial, and Michael E. Saks. An optimal on-line algorithm for metrical task system. *J. ACM*, 39(4):745–763, 1992.

[BN09] Niv Buchbinder and Joseph Naor. Online primal-dual algorithms for covering and packing. *Math. Oper. Res.*, 34(2):270–286, 2009.

[Bur96] William R. Burley. Traversing layered graphs using the work function algorithm. *J. Algorithms*, 20(3):479–511, 1996.

[CL96] 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*, pages 74–96, 1996.

[FL93] Joel Friedman and Nathan Linial. On convex body chasing. *Discrete & Computational Geometry*, 9:293–321, 1993.

[FM00] Amos Fiat and Manor Mendel. Better algorithms for unfair metrical task systems and applications. In *Symposium on Theory of Computing*, pages 725–734, 2000.

[KP95] Elias Koutsoupias and Christos H. Papadimitriou. On the k-server conjecture. *Journal of the ACM*, 42(5):971–983, 1995.

[MG07] Jiří Matoušek and Bernd Gärtner. *Understanding and using linear programming*. Springer-Verlag Berlin Heidelberg, 2007.

[MP91] Paolo Manselli and Carlo Pucci. Maximum length of steepest descent curves for quasi-convex functions. *Geometriae Dedicata*, 38(2):211–227, May 1991.

[Sit14] René Sitters. The generalized work function algorithm is competitive for the generalized 2-server problem. *SIAM J. Comput.*, 43(1):96–125, 2014.

[SS06] René A. Sitters and Leen Stougie. The generalized two-server problem. *Journal of ACM*, 53(3):437–458, 2006.