Chasing Convex Bodies and Functions

Antonios Antoniadis[1], Neal Barcelo[2], Michael Nugent[2], Kirk Pruhs[2,⋆], Kevin Schewior[3,⋆⋆], and Michele Scquizzato[4]

[1] Max-Planck-Institut für Informatik, Saarbrücken, Germany
[2] Department of Computer Science, University of Pittsburgh, Pittsburgh, USA
[3] Technische Universität Berlin, Institut für Mathematik, Berlin, Germany
[4] Department of Computer Science, University of Houston, Houston, USA

Abstract. We consider three related online problems: Online Convex Optimization, Convex Body Chasing, and Lazy Convex Body Chasing. In Online Convex Optimization the input is an online sequence of convex functions over some Euclidean space. In response to a function, the online algorithm can move to any destination point in the Euclidean space. The cost is the total distance moved plus the sum of the function costs at the destination points. Lazy Convex Body Chasing is a special case of Online Convex Optimization where the function is zero in some convex region, and grows linearly with the distance from this region. And Convex Body Chasing is a special case of Lazy Convex Body Chasing where the destination point has to be in the convex region. We show that these problems are equivalent in the sense that if any of these problems have an $O(1)$-competitive algorithm then all of the problems have an $O(1)$-competitive algorithm. By leveraging these results we then obtain the first $O(1)$-competitive algorithm for Online Convex Optimization in two dimensions, and give the first $O(1)$-competitive algorithm for chasing linear subspaces. We also give a simple algorithm and $O(1)$-competitiveness analysis for chasing lines.

1 Introduction

We consider the following three related online problems, all set in a d-dimensional Euclidean space S, with some distance function ρ.

Convex Body Chasing: The input consists of an online sequence $\mathcal{F}_1, \mathcal{F}_2, \ldots, \mathcal{F}_n$ of convex bodies in S. In response to the convex body $\mathcal{F}_i$, the online algorithm has to move to any destination/point $p_i \in \mathcal{F}_i$. The cost of such a feasible solution is the total distance traveled by the online algorithm, namely $\sum_{i=1}^{n} \rho(p_{i-1}, p_i)$. The objective is to minimize the cost. If the convex bodies are restricted to be of a particular type $\mathcal{T}$, then we refer to the problem as $\mathcal{T}$ Chasing. So for example, Line Chasing means that the convex bodies are restricted to being lines.

⋆ Supported, in part, by NSF grants CCF-1115575, CNS-1253218, CCF-1421508, and an IBM Faculty Award.
⋆⋆ Supported by the DFG within the research training group 'Methods for Discrete Structures' (GRK 1408).

Lazy Convex Body Chasing: The input consists of an online sequence of lazy convex bodies $(\mathcal{F}_1, \epsilon_1), (\mathcal{F}_2, \epsilon_2), \ldots, (\mathcal{F}_n, \epsilon_n)$, where each $\mathcal{F}_i$ is a convex body in S, and each slope ϵ_i is a nonnegative real number. In response to the pair $(\mathcal{F}_i, \epsilon_i)$, the online algorithm can move to any destination/point in the metric space S. The cost of such a feasible solution is

$$\sum_{i=1}^{n} \left(\rho(p_{i-1}, p_i) + \epsilon_i \rho(p_i, \mathcal{F}_i)\right),$$

where $\rho(p_i, \mathcal{F}_i)$ is the minimal distance of a point in $\mathcal{F}_i$ to p_i. So the online algorithm need not move inside each convex body, but if it is outside the convex body, in addition to paying the distance traveled, the online algorithm pays an additional cost that is linear in the distance to the convex body. The objective is to minimize the cost. Again if the convex bodies are restricted to be of a particular type $\mathcal{T}$, then we refer to the problem as Lazy $\mathcal{T}$ Chasing.

Online Convex Optimization: The input is an online sequence $F_1, F_2, \ldots, F_n$ of convex functions from S to $\mathbb{R}^+$. In response to the function F_i, the online algorithm can move to any destination/point in the metric space S. The cost of such a feasible solution is

$$\sum_{i=1}^{n} \left(\rho(p_{i-1}, p_i) + F_i(p_i)\right).$$

So the algorithm pays the distance traveled plus the value of the convex functions at the destinations points. The objective is to minimize the cost.

It is easy to see that a c-competitive algorithm for Online Convex Optimization implies a c-competitive algorithm for Lazy Convex Body Chasing, and a c-competitive algorithm for Lazy Convex Body Chasing implies a c-competitive algorithm for Convex Body Chasing. To see this, note that Convex Body Chasing is a special case of Lazy Convex Body Chasing where each ϵ_i is infinite (or, more formally, so large that any competitive algorithm would essentially have to move inside each convex body). Similarly, Lazy Convex Body Chasing is a special case of Online Convex Optimization in which the convex functions are zero on some convex set, and that grow linearly as one moves away from this convex set.

1.1 The History

Our initial interest in Online Convex Optimization arose from applications involving right-sizing data centers [1, 14–18, 21]. In these applications, there is a collection of d centrally-managed data centers, where each data center consists of a homogeneous collection of servers/processors which may be powered down. We represent the state of the data centers by a point in a d-dimensional space where coordinate i represents how many servers are currently powered-on in data center i (the assumption is that there are enough servers in each data center so

that one may reasonably treat the number of servers as a real number instead of an integer). In response to a change in load, the number of servers powered-on in various data centers can be changed. Under the standard assumption that there is some fixed cost for powering a server on, or powering the server off, the Manhattan-distance between states represents the costs for powering on/off servers. The function costs represent the cost for operating the data-centers with the specified number of servers in each data center. The standard models of operating costs, such as those based on either queuing theoretic costs, and those based on energy costs for speed-scalable processors, are convex functions of the state.

Online Convex Optimization for $d = 1$: Essentially all the results in the literature for Online Convex Optimization are restricted to the case that the dimension is $d = 1$. [16] observed that the offline problem can be modeled as a convex program, which is solvable in polynomial time, and that if the line/states are discretized, then the offline problem can be solved by a straight-forward dynamic program. [16] also gave a 3-competitive deterministic algorithm that solves a (progressively larger) convex program at each time. [1] shows that there is an algorithm with sublinear regret, but that $O(1)$-competitiveness and sublinear regret cannot be simultaneously achieved. [1] gave a randomized online algorithm, RBG, and a 2-competitiveness analysis, but there is a bug in the analysis [22]. A revised 2-competitiveness analysis can be found in [2]. Independently, [4] gave a randomized algorithm and showed that it is 2-competitive. [4] also observed that any randomized algorithm can be derandomized, without any loss in the competitive ratio. [4] also gave a simple 3-competitive memoryless algorithm, and showed that this is optimally competitive for memoryless algorithms.

Convex Body Chasing: Convex Body Chasing and Lazy Convex Body Chasing were introduced in [10]. [10] assumed the standard Euclidean distance function, and observed that the optimal competitive ratio is $\Omega(\sqrt{d})$, where d is the dimension of the space. [10] gave a somewhat complicated algorithm and $O(1)$-competitiveness analysis for chasing lines in two dimensions, and observe that any $O(1)$-competitive line chasing algorithm for two dimensions can be extended to an $O(1)$-competitive line chasing algorithm for an arbitrary number of dimensions. [10] gave an even more complicated algorithm and $O(1)$-competitiveness analysis for chasing arbitrary convex bodies in two dimensions. [10] also observed that plane chasing in three dimensions is equivalent to lazy line chasing in two dimensions in the sense that one of these problems has an $O(1)$-competitive algorithm if and only if the other one does. [20] showed in a complicated analysis that the work function algorithm is $O(1)$-competitive for chasing lines and line segments in any dimension. [11] showed that the greedy algorithm is $O(1)$-competitive if $d = 2$ and the convex bodies are regular polygons with a constant number of sides.

Classic Online Problems: Online Convex Optimization is also related to several classic online optimization problems. It is a special case of the *metrical task*

system problem in which the metric space is restricted to be a d-dimensional Euclidean space and the costs are restricted to be convex functions on that space. The optimal deterministic competitive ratio for a general metrical task system is $2n-1$, where n is the number of points in the metric [7], and the optimal randomized competitive ratio is $\Omega(\log n/\log\log n)$ [5,6] and $O(\log^2 n \log\log n)$ [9]. Online Convex Optimization is related to the *allocation problem* defined in [3], which arises when developing a randomized algorithm for the classic k-server problem using tree embeddings of the underlying metric space [3,8]. In fact, the algorithm RBG in [1] is derived from a similar algorithm in [8] for this allocation problem. The classic *ski rental* problem, where randomized algorithms are allowed, is a special case of Online Convex Optimization. The optimal competitive ratio for randomized algorithms for the ski rental problem is $e/(e-1)$ [12]. [4] showed that the optimal competitive ratio for Online Convex Optimization for $d = 1$ is strictly greater than the one for online ski rental. The k-server and CNN problems [13] can be viewed as chasing nonconvex sets.

1.2 Our Results

In Section 2 we show that all three of the problems that we consider are equivalent in the sense that if one of the problems has an $O(1)$-competitive algorithm, then they all have $O(1)$-competitive algorithms. More specifically, we show that if there is an $O(1)$-competitive algorithm for Lazy Convex Body Chasing in d dimensions then there is an $O(1)$-competitive algorithm for Online Convex Optimization in d dimensions. The crux of this reduction is to show that any convex function can be approximated to within a constant factor by a finite collection of lazy convex bodies. We then show that if there is an $O(1)$-competitive algorithm for Convex Body Chasing in d dimensions, then there is an $O(1)$-competitive algorithm for Lazy Convex Body Chasing (objects of the same type) in d dimensions. Intuitively in this reduction, each lazy convex body $(\mathcal{F}_i, \epsilon_i)$ is fed to the Convex Body Chasing algorithm with probability ϵ_i. As in [4], this algorithm can be derandomized by deterministically moving to the expected location of the randomized algorithm. The equivalence of these problems follows by combining these reductions with the obvious reductions in the other direction. Combining these reductions with the results in [10], most notably the $O(1)$-competitive algorithm for chasing halfspaces in two dimensions, we obtain an $O(1)$-competitive algorithm for Online Convex Optimization in two dimensions.

In Section 3 we give an online algorithm for Convex Body Chasing when the convex bodies are subspaces, in any dimension, and an $O(1)$-competitiveness analysis. In this context, subspace means a linear subspace closed under vector addition and scalar multiplication; So a point, a line, a plane, etc. The two main components of the algorithm are (1) A reduction from hyperplane chasing in d dimensions to lazy hyperplane chasing in $d-1$ dimensions, and (2) our reduction from Lazy Convex Body Chasing to Convex Body Chasing. The first reduction is the natural generalization of the continuous reduction from plane chasing in three dimesions to lazy line chasing in two dimensions given in [10]. Combining these two components gives an $O(1)$-approximation reduction from subspace chasing

in d dimensions to subspace chasing in $d-1$ dimensions. One then obtains a $2^{O(d)}$-competitive algorithm by repeated applications of these reductions, and the use of any of the $O(1)$-competitive algorithms for Online Convex Optimization in one dimension. Within the context of right-sizing data centers, it is reasonable to assume that the number of data centers is a smallish constant, and thus this algorithm would be $O(1)$-competitive under this assumption.

In Section 4 we give an online algorithm for chasing lines and line segments in any dimension, and show that it is $O(1)$-competitive. The underlying insight of our online algorithm is the same as in [10], to be greedy with occasional adjustments toward the area where the adversary might have cheaply handled recent requests. However, our algorithm is cleaner/simpler than the algorithm in [10]. In particular our algorithm is essentially memoryless as the movement is based solely on the last two lines, instead of an unbounded number of lines as in [10]. Our analysis is based on a simple potential function: the distance between the location for the online algorithm and for the adversary, and is arguably cleaner than the analysis in [10], and is certainly cleaner than the analysis of the work function algorithm in [20].

While our results are not that technically deep, they do provide a much clearer picture of the algorithmic relationship of the various online problems in this area. Our results also suggest that the "right" problem to attack in this area is finding (if it exists) an $O(1)$-competitive algorithm for half-space chasing, as this is the simplest problem that would give an $O(1)$-competitive algorithm for all of these problems.

For concreteness we will assume ρ is the standard Euclidean distance function. Although as our focus is on $O(1)$-approximation, without being too concerned about the exact constant, our results will also hold for the Manhattan distance, and other standard normed distances.

2 Reductions

In this section we show in Lemma 1 that Lazy Convex Body Chasing is reducible to Convex Body Chasing, in Lemma 2 that Online Convex Optimization is reducible to Lazy Convex Body Chasing, and in Corollary 1 that these reductions give an $O(1)$-competitive algorithm for Online Convex Optimization in two dimensions.

Lemma 1. *If there is an $O(1)$-competitive algorithm A_C for Convex Body Chasing in d dimensions, then there is an $O(1)$-competitive algorithm A_L for Lazy Convex Body Chasing in d dimensions. The same result holds if the convex bodies in both problems are restricted to be of a particular type.*

Proof. We build a randomized algorithm A_L from A_C and then explain how to derandomize it. We first modify the input instance by replacing each lazy convex body $(\mathcal{F}_i, \epsilon_i)$ whose slope ϵ_i is greater than 1 by $\lceil \epsilon_i \rceil$ lazy convex bodies, each having $\mathcal{F}_i$ as the convex body. The first $\lfloor \epsilon_i \rfloor$ of these lazy convex bodies will have slope 1, and the potentially remaining convex body will have slope $\epsilon_i - \lfloor \epsilon_i \rfloor$. This

modification does not affect the optimal cost, and will not decrease the online cost. From now on, we will assume that any input instance for Lazy Convex Body Chasing is of this modified form. It is easy to see how one can go back from a solution to the modified input to one to the original input without increasing the cost, since our algorithm will never "move away" from the line that just arrived.

Algorithm A_L: Upon the arrival of a new lazy convex body $(\mathcal{F}_i, \epsilon_i)$, the algorithm with (independent) probability $1 - \epsilon_i$ does not move, and with probability ϵ_i passes $\mathcal{F}_i$ to A_C and moves to the location to which A_C moves.

Notice that in the modified input instance every slope is a real number in $[0, 1]$, and thus probabilities ϵ_i and $1 - \epsilon_i$ are all well defined.

Analysis: Consider a particular input instance I_L of Lazy Convex Body Chasing, as defined before. Let I_C denote the random variable representing the sequence of convex bodies passed to A_C. Let $\textsc{Opt}_L$ be the optimal solution for Lazy Convex Body Chasing on I_L. Let $\textsc{Opt}_C$ be a random variable equal to the optimal solution for Convex Body Chasing on I_C. Let $\textsc{Opt}_T$ be a random variable equal to the optimal solution for the Lazy Convex Body Chasing instance I_T derived from I_C by replacing each $\mathcal{F}_i \in I_C$ by the lazy convex body $(\mathcal{F}_i, 1)$. We will use absolute value signs to denote the cost of a solution.

The $O(1)$-competitiveness of A_L then follows from the following sequence of inequalities:

$$
\begin{aligned}
\mathbb{E}[|A_L(I_L)|] &= \sum_i \mathbb{P}[\mathcal{F}_i \in I_C]\ \mathbb{E}[\text{Cost of } A_C \text{ on } \mathcal{F}_i \mid \mathcal{F}_i \in I_C] \\
&\quad + \sum_i \mathbb{P}[\mathcal{F}_i \notin I_C]\ \mathbb{E}[\text{Cost of } A_L \text{ on } \mathcal{F}_i \mid \mathcal{F}_i \notin I_C] && (1)\\
&= \sum_i \epsilon_i\, \mathbb{E}[\text{Cost of } A_C \text{ on } \mathcal{F}_i \mid \mathcal{F}_i \in I_C] \\
&\quad + \sum_i (1-\epsilon_i)\ \mathbb{E}[\epsilon_i\, \rho(p_{i-1}, \mathcal{F}_i)] && (2)\\
&\leq \sum_i \epsilon_i\, \mathbb{E}[\text{Cost of } A_C \text{ on } \mathcal{F}_i \mid \mathcal{F}_i \in I_C] \\
&\quad + \sum_i \epsilon_i\, \mathbb{E}[\rho(p_{i-1}, \mathcal{F}_i)] && (3)\\
&\leq 2 \sum_i \epsilon_i\, \mathbb{E}[\text{Cost of } A_C \text{ on } \mathcal{F}_i \mid \mathcal{F}_i \in I_C] && (4)\\
&= 2 \sum_i \mathbb{P}[\mathcal{F}_i \in I_C]\ \mathbb{E}[\text{Cost of } A_C \text{ on } \mathcal{F}_i \mid \mathcal{F}_i \in I_C] && (5)\\
&= 2\, \mathbb{E}[|A_C(I_C)|] && (6)\\
&= O(\mathbb{E}[|\textsc{Opt}_C|]) && (7)\\
&= O(\mathbb{E}[|\textsc{Opt}_T|]) && (8)\\
&= O(|\textsc{Opt}_L|). && (9)
\end{aligned}
$$

Equality (1) follows from the definitions of expectation and conditional expectation, and linearity of expectation. Notice that all expectations involving $\mathcal{F}_i$ only depend upon the history up until $\mathcal{F}_i$ arrives. Equality (2) follows from the fact that $\mathcal{F}_i$ is added to I_C with probability ϵ_i, and if $\mathcal{F}_i$ is not added then A_L pays ϵ_i times the distance to $\mathcal{F}_i$. Inequality (3) follows from the linearity of expectation and since $1-\epsilon_i \leq 1$. Inequality (4) holds since A_C has to move to each $\mathcal{F}_i \in I_C$ and thus in expectation has to pay at least $\mathbb{E}[\rho(p_{i-1}, \mathcal{F}_i)]$ (note that only by independence of the coin flips the expected position of A_C in case $\mathcal{F}_i \in I_C$ is identical to the expected position of A_L). Equality (5) holds since $\mathcal{F}_i \in I_C$ with probability ϵ_i. Equality (6) follows by linearity of expectation and the definition of conditional expectation. Inequality (7) follows by the assumption that A_C is $O(1)$-competitive. To prove Inequality (8) it is sufficient to construct a solution $\mathcal{S}$ for each possible instantiation of I_C that is at most a constant times more expensive than $\textsc{Opt}_T$. In response to a convex body $\mathcal{F}_i \in I_C$, $\mathcal{S}$ moves to the same destination point p_i as $\textsc{Opt}_T$, then moves to the closest point on $\mathcal{F}_i$, and then back to p_i. Thus the movement cost for $\mathcal{S}$ is at most the movement cost for $\textsc{Opt}_T$ plus twice the function costs for $\textsc{Opt}_T$. To prove Inequality (9) it is sufficient to construct an algorithm B to solve Lazy Convex Body Chasing on I_T with expected cost $O(|\textsc{Opt}_L|)$. For each convex body in $\mathcal{F}_i \in I_C$, Algorithm B first moves to the destination point p_i that $\textsc{Opt}_L$ moves to after $\mathcal{F}_i$. Call this a basic move. Then algorithm B moves to the closest point in $\mathcal{F}_i$, and then back to p_i. Call this a detour move. Then by the triangle inequality the expected total cost of the basic moves for algorithm B is at most the movement cost of $\textsc{Opt}_L$. The probability that algorithm B incurs a detour cost for convex body $\mathcal{F}_i$ is ϵ_i, and when it incurs a detour cost, this detour cost is $2/\epsilon_i$ times the function cost incurred by $\textsc{Opt}_L$. Thus the expected cost for algorithm B on I_T is at most $3|\textsc{Opt}_L|$.

Derandomization: As in [4], we can derandomize A_L to get a deterministic algorithm A_D with the same competitive ratio as A_L. A_D always resides in the expected position of A_L. More specifically, let x_i be a random variable denoting the position of A_L directly after the arrival of $\mathcal{F}_i$. Then A_D sets its position to $\mu_i := \mathbb{E}[x_i]$.

Then, we have that for each step i, A_L's expected cost is $\mathbb{E}[\rho(x_{i-1}, x_i)] + \epsilon_i \mathbb{E}[\rho(x_i, \mathcal{F}_i)]$. On the other hand, A_D's cost is $\rho(\mathbb{E}[x_i], \mathbb{E}[x_{i-1}]) + \epsilon_i \rho(\mathbb{E}[x_i], \mathcal{F}_i)$. By a generalization of Jensen's inequality (see for example Proposition B.1, page 343 in the book by Marshall and Olkin [19]), and by the convexity of our distance function ρ (the distance function is a norm, and therefore convexity follows by triangle inequality and absolute homogeneity), we have, for each i,

$$\mathbb{E}[\rho(x_i, x_{i-1})] \geq \rho(\mathbb{E}[x_i], \mathbb{E}[x_{i-1}])$$

and

$$\mathbb{E}[\rho(x_i, \mathcal{F}_i)] \geq \rho(\mathbb{E}[x_i], \mathcal{F}_i).$$

Summing over all i completes the analysis. □

Lemma 2. *If there is an $O(1)$-competitive algorithm A_L for Lazy Convex Body Chasing in d dimensions, then there is an $O(1)$-competitive algorithm A_O for Online Convex Optimization in d dimensions.*

Proof. Consider an arbitrary instance I_O of the convex optimization problem. We can without loss of generality ignore the prefix of the sequence of functions that can be handled with zero cost. So let $L > 0$ be the optimal cost for chasing function F_1. The algorithm will use L as a lower bound for the optimal cost.

For each function F_i that it sees, the algorithm A_O feeds the algorithm A_L a finite collection $\mathcal{C}_i$ of lazy convex bodies, and then moves to the final destination point that A_L moved to. Let I_L be the resulting instance of Lazy Convex Body Chasing. To define $\mathcal{C}_i$ assume without loss of generality that the minimum of F_i occurs at the origin. We can also assume without loss of generality that the minimum of F_i is zero.

Define $F_i'(x)$ to be the partial derivative of F_i at the point $x \in S = \mathbb{R}^d$ in the direction away from the origin. Now let C_j be the curve in $\mathbb{R}^{d+1}$ corresponding to the points $(x, F_i'(x))$ where $F_i'(x) = 2^j$ for integer $j \in (-\infty, +\infty)$. (Or more technically where the $F_i'(x)$ transitions from being less than 2^j to more than 2^j.) Let D_j be the projection of C_j onto S. Note that D_j is convex.

Let u be the minimum integer such that the location of A_O just before F_i is inside of D_u. Let ℓ be the maximum integer such that:

- the diameter of D_ℓ is less than $L/8^i$,
- the maximum value of $F_i(x)$ for an $x \in D_\ell$ is less than $L/8^i$, and
- $\ell < u - 10$.

Then $\mathcal{C}_i$ consists of the lazy convex bodies $(D_j, 2^j)$ for $j \in [\ell, u]$.

Let $G_j(x)$ be the function that is zero within D_j and grows linearly at a rate 2^j as one moves away from D_j. Now what we want to prove is that $\sum_j G_j(x) = \Theta(F_i(x))$ for all x outside of $D_{\ell+2}$. To do this consider moving toward x from the origin. Consider the region between D_j and D_{j+1} for $j \geq \ell + 2$, and a point y in this region that lies on the line segment between the origin and x. Then we know that $2^j \leq F_i'(y) \leq 2^{j+1}$. Thus as we are moving toward x, the rate of increase of $\sum_j G_j(x)$ is within a constant factor of the rate of increase of $F_i(x)$, and thus $\sum_j G_j(x) = \Theta(F_i(x))$.

Let $\textsc{Opt}_L$ be the optimal solution for the Lazy Line Chasing instance I_L and Let $\textsc{Opt}_O$ be the optimal solution for the Online Convex Optimization instance I_O. Now the claim follows via the following inequalities:

$$|A_O(I_O)| = O(|A_L(I_L)|) \tag{10}$$
$$= O(|\textsc{Opt}_L|) \tag{11}$$
$$= O(|\textsc{Opt}_O|). \tag{12}$$

Inequality (10) holds since the movement cost for A_L and A_O are identical, and the function costs are within a constant of each other by the observation that $\sum_j G_j(x) = \Theta(F_i(x))$. Inequality (11) holds by the assumption that A_L is $O(1)$-competitive. Inequality (12) holds by the observation that $\sum_j G_j(x) = \Theta(F_i(x))$

and the observation that the maximum savings that OPT_L can obtain from being inside of each D_ℓ is at most $L/2$. □

Corollary 1. *There is an $O(1)$-competitive algorithm for Online Convex Optimization in two dimensions.*

Proof. This follows from the reduction from Online Convex Optimization to Lazy Convex Body Chasing in Lemma 2, the reduction from Lazy Convex Body Chasing to Convex Body Chasing in Lemma 1, and the $O(1)$-competitive algorithm for Convex Body Chasing in two dimensions in [10]. □

3 Subspace Chasing

In this section we describe a $2^{O(d)}$-competitive algorithm for chasing subspaces in any dimension d. As noticed in [10], it suffices to give such an algorithm for chasing $(d-1)$-dimensional subspaces (hyperplanes). Essentially this is because every $f \leq d-1$-dimensional subspace is the intersection of $d-f$ hyperplanes, and by repeating these $d-f$ hyperplanes many times, any competitive algorithm can be forced arbitrarily close to their intersection.

Algorithm for chasing hyperplanes: The two main components of our algorithm for chasing hyperplanes in d dimensions are:

- A reduction from hyperplane chasing in d dimensions to lazy hyperplane chasing in $d-1$ dimensions. This is a discretized version of the continuous reduction given in [10] for $d = 3$. In this section we give an overview of the reduction, and the analysis, but defer the formal proof to the full version of the paper.
- Our reduction from Lazy Convex Body Chasing to Convex Body Chasing in the previous section.

Combining these two components gives a reduction from subspace chasing in d dimensions to subspace chasing in $d-1$ dimensions. One then obtains a $2^{O(d)}$-competitive algorithm by repeated applications of these reductions, and the use of any of the $O(1)$-competitive algorithms for lazy point chasing (or online convex optimization) when $d = 1$ [2, 4, 16].

Description of reduction from hyperplane chasing in dimension d to lazy hyperplane chasing in dimension $d-1$: Let A_{LHC} be the algorithm for lazy hyperplane chasing in dimension $d-1$. We maintain a bijective mapping from the $\mathbb{R}^{d-1}$ space S that the algorithm A_{LHC} moves in to the last hyperplane in $\mathbb{R}^d$. Initially, this mapping is an arbitrary one that maps the origin of $\mathbb{R}^{d-1}$ to the origin of $\mathbb{R}^d$. Call this hyperplane $\mathcal{F}_0$.

Each time a new hyperplane $\mathcal{F}_i$ in $\mathbb{R}^d$ arrives, the algorithm moves in the following way:

- If $\mathcal{F}_i$ is parallel to $\mathcal{F}_{i-1}$, then the algorithm moves to the nearest position in $\mathcal{F}_i$.

– If $\mathcal{F}_i$ is not parallel to $\mathcal{F}_{i-1}$, the two hyperplanes intersect in a $(d-2)$-dimensional subspace $\mathcal{I}_i$. Let $\alpha_i \leq \pi/2$ radians be the angle between hyperplanes $\mathcal{F}_{i-1}$ and $\mathcal{F}_i$. The algorithm then calls A_{LHC} with $(\mathcal{I}_i, \alpha_i)$. Let p_i be the point within $\mathcal{F}_{i-1}$ that A_{LHC} moves to. The bijection between $\mathcal{F}_i$ and S is obtained from the bijection between $\mathcal{F}_{i-1}$ and S by rotating α_i radians around $\mathcal{I}_i$. The algorithm then moves to the location of p_i in $\mathcal{F}_i$.

Analysis overview: Consecutive parallel hyperplanes is the easy case. In this case, one can assume, at a loss of a factor of $\sqrt{2}$ in competitive ratio, that the optimal solution moves to the closest point on the new parallel hyperplane. Thus any competitive ratio c that one can prove under the assumption that consecutive hyperplanes are not parallel will hold in general as long as $c \geq \sqrt{2}$.

When there are no two consecutive non-parallel hyperplanes, we show that the cost for the reduction algorithm and the cost for A_{LHC} are within a constant of each other, and similarly the optimal cost for hyperplane chasing in d dimensions and the optimal cost for lazy hyperplane chasing are within a constant of each other. Intuitively this is because the additional movement costs incurred in (non-lazy) hyperplane chasing can be related to the angle between the last two hyperplanes, and thus to the distance cost (for not being on the hyperplane) that has to be paid in lazy hyperplane chasing. From this we can conclude that:

Theorem 1. *There is a $2^{O(d)}$-competitive algorithm for subspace chasing in d dimensions.*

We note that our algorithm can be implemented to run in time polynomial in d and n.

4 Line Chasing

We give an online algorithm for chasing lines and line segments in any dimension, and show that it is $O(1)$-competitive. Let $\mathcal{E}_i$ be the unique line that is an extension of the line segment $\mathcal{F}_i$.

Algorithm Description: Let $\mathcal{P}_i$ be a plane containing the line $\mathcal{E}_i$ that is parallel to the line $\mathcal{E}_{i-1}$ (note this is uniquely defined when $\mathcal{E}_i$ and $\mathcal{E}_{i-1}$ are not parallel). The algorithm first moves to the closest point $h_i \in \mathcal{P}_i$. The algorithm then moves to the closest point g_i in $\mathcal{E}_i$.

– If $\mathcal{E}_{i-1}$ and $\mathcal{E}_i$ are parallel, then the algorithm stays at g_i.
– If $\mathcal{E}_{i-1}$ and $\mathcal{E}_i$ are not parallel, let m_i be the intersection of $\mathcal{E}_i$ and the projection of $\mathcal{E}_{i-1}$ onto $\mathcal{P}_i$. Let $\beta \in (0,1)$ be some constant that we shall set later. The algorithm makes an adjustment by moving toward m_i along $\mathcal{E}_i$ until it has traveled a distance of $\beta \cdot \rho(h_i, g_i)$, or until it reaches m_i.
– Finally, the algorithm moves to the closest point p_i in $\mathcal{F}_i$.

Theorem 2. *This algorithm is $O(1)$-competitive for Line Chasing in any dimension.*

Proof. Initially assume that all the line segments are lines. Let $h_i^\star$ be the projection of the adversary's position, just before $\mathcal{F}_i$ arrives, onto $\mathcal{P}_i$. We will assume that the adversary first moves to $h_i^\star$, and then to the nearest point $g_i^\star$ on $\mathcal{F}_i$, and then to some arbitrary final point $p_i^\star$ on $\mathcal{F}_i$. This assumption increases the adversary's movement cost by at most a factor of $\sqrt{3}$ (a similar observation is made in [10]). We will charge the algorithm's cost for moving to $\mathcal{P}_i$ to the adversary's cost of moving to $\mathcal{P}_i$. Thus by losing at most a factor of $\sqrt{3}$ in the competitive ratio, we can assume that the adversary and the algorithm are at positions $h_i^\star$ and h_i right before $\mathcal{F}_i$ arrives.

Let $\textsc{Alg}_i$ and $\textsc{Opt}_i$ denote the movement cost of the algorithm and the adversary, respectively, in response to $\mathcal{F}_i$. Let the potential function Φ_i be $\delta \cdot \rho(p_i, p_i^\star)$ for some to be determined constant $\delta > 1$. To show that the algorithm is c-competitive it will be sufficient to show that, for each i,

$$\textsc{Alg}_i + \Phi_i - \Phi_{i-1} \leq c \cdot \textsc{Opt}_i. \tag{13}$$

We will only initially consider the adversary's movement cost until it reaches $g_i^\star$. Equation (13) will continue to hold for any adversary's movement after $g_i^\star$ if

$$c \geq \delta. \tag{14}$$

We consider three cases. The notation that we will use is illustrated in Figure 1.

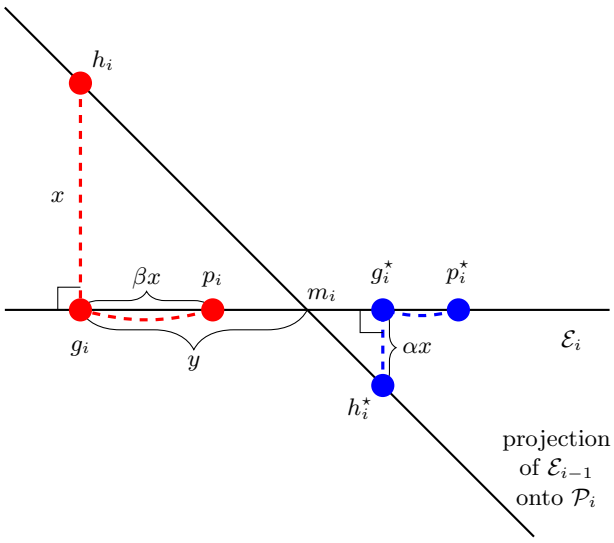

Fig. 1. An overview of the used points and distances in $\mathcal{P}_i$.

In the first case assume $\rho(h_i^\star, g_i^\star) \geq \gamma x$, where $x = \rho(h_i, g_i)$, and $\gamma > 0$ is a constant that we define later. Intuitively, this is the easiest case as the adversary's

cost will pay for both the algorithm's movement cost and the increase in the potential due to this movement. For Equation (13) to hold, it is sufficient that

$$(1+\beta)x + \delta((1+\beta)x + \gamma x) \leq c\gamma x,$$

or equivalently

$$c \geq \frac{(1+\beta) + \delta((1+\beta)+\gamma)}{\gamma}. \tag{15}$$

In the remaining two cases assume that $\rho(h_i^\star, g_i^\star) = \alpha x \leq \gamma x$, and let $y = \rho(m_i, g_i)$.

In the second case assume that $x \leq y$. Intuitively in this case the decrease in the potential due to the algorithm's adjustment on $\mathcal{F}_i$ decreases the potential enough to pay for the algorithm's movement costs. Since $\beta < 1$ and $x \leq y$, the algorithm will not have to stop at m_i when moving a distance of βx on $\mathcal{F}_i$ toward m_i. If

$$\gamma \leq 1 - \beta, \tag{16}$$

the algorithm will also not cross $g_i^\star$ while adjusting on $\mathcal{F}_i$, as this would contradict the assumption that $\rho(h_i^\star, g_i^\star) \leq \gamma x$. Equation (13) will be hardest to satisfy when $\Phi_i - \Phi_{i-1}$ is maximal. This will occur when $g_i^\star$ is maximally far from g_i, which in turn occurs when the points $g_i^\star$, m_i, and g_i lie on $\mathcal{F}_i$ in that order. In that case, Equation (13) evaluates to

$$(1+\beta)x + \delta((1+\alpha)y - \beta x - (1+\alpha)\sqrt{x^2+y^2}) \leq c\alpha x.$$

Setting $L = \frac{1+\beta-\delta\beta-c\alpha}{\delta(1+\alpha)}$, this is equivalent to

$$Lx + y \leq \sqrt{x^2+y^2}. \tag{17}$$

When $x \leq y$, Equation (17) holds if $L \leq 0$. This in turn holds when

$$\delta \geq \frac{1+\beta}{\beta}. \tag{18}$$

Finally we consider the third case that $x \geq y$. Intuitively in this case the decrease in the potential due to the algorithm's movement toward $\mathcal{F}_i$ decreases the potential enough to pay for the algorithms movement costs. First consider the algorithm's move from p_{i-1} to g_i. Again, the value of $\Phi_i - \Phi_{i-1}$ is maximized when $g_i^\star$ is on the other side of m_i as is g_i. In that case, the value of $\Phi_i - \Phi_{i-1}$ due to this move is at most

$$\delta((1+\alpha)y - (1+\alpha)\sqrt{x^2+y^2}).$$

Note that the maximum possible increase in the potential due to the algorithm moving from g_i to p_i is $\delta\beta x$. Thus for Equation (13) to hold, it is sufficient that

$$(1+\beta)x + \delta\beta x + \delta((1+\alpha)y - (1+\alpha)\sqrt{x^2+y^2}) \leq c\alpha x.$$

Setting $L = \frac{1+\beta+\delta\beta-c\alpha}{\delta(1+\alpha)}$, this is equivalent to

$$Lx + y \leq \sqrt{x^2 + y^2}. \tag{19}$$

When $x \geq y$, Equation (19) holds if $L \leq \sqrt{2} - 1$. This in turn holds when

$$\delta \geq \frac{1+\beta}{\sqrt{2} - 1 - \beta}. \tag{20}$$

We now need to find a feasible setting of β, γ, δ, and compute the resulting competitive ratio. Setting $\beta = \frac{\sqrt{2}-1}{2}$ and $\gamma = \frac{3-\sqrt{2}}{2}$, and $\delta = \frac{\sqrt{2}+1}{\sqrt{2}-1}$ one can see that equations (16), (18), and (20) hold. The minimum c satisfying (15) is $c \approx 16.22$ and also satisfies (14). Then given that we overestimate the adversary's cost by a most a factor of $\sqrt{3}$, this gives us a competitive ratio for lines of approximately 28.1, approximately the same competitive ratio as obtained in [10].

If $\mathcal{F}_i$ is a line segment, then we need to account for the additional movement along $\mathcal{E}_i$ to reach $\mathcal{F}_i$. However, as we set $\delta > 1$, the decrease in the potential can pay for this movement cost. □

Acknowledgement

We thank Nikhil Bansal, Anupam Gupta, Cliff Stein, Ravishankar Krishnaswamy, and Adam Wierman for helpful discussions. We also thank an anonymous reviewer for pointing out an important subtlety in one of our proofs.

References

1. L. L. H. Andrew, S. Barman, K. Ligett, M. Lin, A. Meyerson, A. Roytman, and A. Wierman. A tale of two metrics: Simultaneous bounds on competitiveness and regret. In *Conference on Learning Theory*, pages 741–763, 2013.
2. L. L. H. Andrew, S. Barman, K. Ligett, M. Lin, A. Meyerson, A. Roytman, and A. Wierman. A tale of two metrics: Simultaneous bounds on competitiveness and regret. *CoRR*, abs/1508.03769, 2015.
3. N. Bansal, N. Buchbinder, and J. Naor. Towards the randomized k-server conjecture: A primal-dual approach. In *ACM-SIAM Symposium on Discrete Algorithms*, pages 40–55, 2010.
4. N. Bansal, A. Gupta, R. Krishnaswamy, K. Pruhs, K. Schewior, and C. Stein. A 2-competitive algorithm for online convex optimization with switching costs. In *Workshop on Approximation Algorithms for Combinatorial Optimization Problems*, pages 96–109, 2015.
5. Y. Bartal, B. Bollobás, and M. Mendel. Ramsey-type theorems for metric spaces with applications to online problems. *J. Comput. Syst. Sci.*, 72(5):890–921, 2006.
6. Y. Bartal, N. Linial, M. Mendel, and A. Naor. On metric Ramsey-type phenomena. In *ACM Symposium on Theory of Computing*, pages 463–472, 2003.
7. A. Borodin, N. Linial, and M. E. Saks. An optimal on-line algorithm for metrical task system. *J. ACM*, 39(4):745–763, 1992.

8. A. Coté, A. Meyerson, and L. Poplawski. Randomized k-server on hierarchical binary trees. In *ACM Symposium on Theory of Computing*, pages 227–234, 2008.
9. A. Fiat and M. Mendel. Better algorithms for unfair metrical task systems and applications. *SIAM J. Comput.*, 32(6):1403–1422, 2003.
10. J. Friedman and N. Linial. On convex body chasing. *Discrete & Computational Geometry*, 9:293–321, 1993.
11. H. Fujiwara, K. Iwama, and K. Yonezawa. Online chasing problems for regular polygons. *Inf. Process. Lett.*, 108(3):155–159, 2008.
12. A. R. Karlin, M. S. Manasse, L. A. McGeoch, and S. S. Owicki. Competitive randomized algorithms for nonuniform problems. *Algorithmica*, 11(6):542–571, 1994.
13. E. Koutsoupias and D. S. Taylor. The CNN problem and other k-server variants. *Theor. Comput. Sci.*, 324(2-3):347–359, 2004.
14. M. Lin, Z. Liu, A. Wierman, and L. L. H. Andrew. Online algorithms for geographical load balancing. In *International Green Computing Conference*, pages 1–10, 2012.
15. M. Lin, A. Wierman, L. L. H. Andrew, and E. Thereska. Online dynamic capacity provisioning in data centers. In *Allerton Conference on Communication, Control, and Computing*, pages 1159–1163, 2011.
16. M. Lin, A. Wierman, L. L. H. Andrew, and E. Thereska. Dynamic right-sizing for power-proportional data centers. *IEEE/ACM Trans. Netw.*, 21(5):1378–1391, 2013.
17. M. Lin, A. Wierman, A. Roytman, A. Meyerson, and L. L. H. Andrew. Online optimization with switching cost. *SIGMETRICS Performance Evaluation Review*, 40(3):98–100, 2012.
18. Z. Liu, M. Lin, A. Wierman, S. H. Low, and L. L. H. Andrew. Greening geographical load balancing. In *ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems*, pages 233–244, 2011.
19. A. W. Marshall and I. Olkin. *Inequalities: Theory of Majorization and Its Application*. Academic Press, 1979.
20. R. Sitters. The generalized work function algorithm is competitive for the generalized 2-server problem. *SIAM J. Comput.*, 43(1):96–125, 2014.
21. K. Wang, M. Lin, F. Ciucu, A. Wierman, and C. Lin. Characterizing the impact of the workload on the value of dynamic resizing in data centers. In *IEEE INFOCOM*, pages 515–519, 2013.
22. A. Wierman. Personal communication, 2015.