Proceedings of Machine Learning Research vol 178:1–41, 2022 35th Annual Conference on Learning Theory

Efficient Convex Optimization Requires Superlinear Memory

Annie Marsden MARSDEN@STANFORD.EDU
Stanford University

Vatsal Sharan VSHARAN@USC.EDU
University of Southern California *

Aaron Sidford SIDFORD@STANFORD.EDU
Stanford University

Gregory Valiant VALIANT@STANFORD.EDU
Stanford University

Editors: Po-Ling Loh and Maxim Raginsky

Abstract

We show that any memory-constrained, first-order algorithm which minimizes d-dimensional, 1-Lipschitz convex functions over the unit ball to $1/\operatorname{poly}(d)$ accuracy using at most $d^{1.25-\delta}$ bits of memory must make at least $\tilde{\Omega}(d^{1+(4/3)\delta})$ first-order queries (for any constant $\delta \in [0, 1/4]$). Consequently, the performance of such memory-constrained algorithms are a polynomial factor worse than the optimal $\tilde{O}(d)$ query bound for this problem obtained by cutting plane methods that use $\tilde{O}(d^2)$ memory. This resolves one of the open problems in the COLT 2019 open problem publication of Woodworth and Srebro.

Keywords: Convex optimization, first-order methods, cutting plane methods, memory lower bounds

1. Introduction

Minimizing a convex objective function $f : \mathbb{R}^d \to \mathbb{R}$ given access to a first order oracle—that returns the function evaluation and (sub)gradient $(f(\mathbf{x}), \nabla f(\mathbf{x}))$ when queried for point $\mathbf{x}$—is a canonical problem and fundamental primitive in machine learning and optimization.

There are methods that, given any 1-Lipschitz, convex $f : \mathbb{R}^d \to \mathbb{R}$ accessible via a first-order oracle, compute an ϵ-approximate minimizer over the unit ball with just $\mathcal{O}(\min\{\epsilon^{-2}, d\log(1/\epsilon)\})$ first order queries. This query complexity is worst-case optimal (Nemirovski and Yudin, 1983) and foundational in optimization theory. $\mathcal{O}(\epsilon^{-2})$ queries is achievable using subgradient descent; this is a simple, widely-used, eminently practical method that solves the problem using a total of $\mathcal{O}(d\epsilon^{-2})$ computation time (assuming arithmetic operations on $\mathcal{O}(\log(d/\epsilon))$-bit numbers take constant time). On the other hand, building on the $O(d^2\log(1/\epsilon))$ query complexity of the well-known ellipsoid method (Yudin and Nemirovskii, 1976; Shor, 1977), different cutting plane methods achieve a query complexity of $\mathcal{O}(d\log(1/\epsilon))$, e.g. center of mass with sampling based techniques (Levin, 1965; Bertsimas and Vempala, 2004), volumetric center (Vaidya, 1989; Atkinson and Vaidya, 1995), inscribed ellipsoid (Khachiyan et al., 1988; Nesterov, 1989); these methods are perhaps less frequently used in practice and large-scale learning and all use at least $\Omega(d^3\log(1/\epsilon))$-time, even with recent improvements (Lee et al., 2015; Jiang et al., 2020).[1]

* Part of the work was done while the author was at MIT.

1. This arises from taking at least $\Omega(d\log(1/\epsilon))$ iterations of working in some change of basis or solving a linear system, each of which take at least $\Omega(d^2)$-time naively.

Though state-of-the-art cutting plane methods have larger computational overhead and are sometimes regarded as impractical in different settings, for small enough ϵ, they give the state-of-the-art query bounds. Further, in different theoretical settings, e.g. semidefinite programming (Anstreicher, 2000; Sivaramakrishnan and Mitchell, 2012; Lee et al., 2015), submodular optimization (McCormick, 2005; Grötschel et al., 2012; Lee et al., 2015; Jiang, 2021) and equilibrium computation (Papadimitriou and Roughgarden, 2008; Jiang and Leyton-Brown, 2011), cutting-plane-methods have yielded state-of-the-art runtimes at various points of time. This leads to the natural question of what is needed of a method to significantly outperform gradient descent and take advantage of the improved query complexity enjoyed by cutting plane methods? Can we design methods that obtain optimal query complexities while maintaining the practicality of gradient descent methods?

Towards addressing this question, in a COLT 2019 open problem, Woodworth and Srebro (2019) suggested using memory as a lens. While subgradient descent can be implemented using just $\mathcal{O}(d\log(1/\epsilon))$-bits of memory (Woodworth and Srebro, 2019) all known methods that achieve a query complexity significantly better than gradient descent, e.g. cutting plane methods, require $\Omega(d^2\log(1/\epsilon))$ bits of memory. Understanding the trade-offs in memory and query complexity could inform the design of future efficient optimization methods.

In this paper we show that memory *does* play a critical role in attaining optimal query complexity for convex optimization. Our main result is the following theorem which shows that any algorithm whose memory usage is sufficiently small (though still superlinear) must make polynomially more queries to a first-order oracle than cutting plane methods. Specifically, any algorithm that uses significantly less than $d^{1.25}$ bits of memory requires a polynomial factor more first order queries than the optimal $\mathcal{O}(d\log(d))$ queries achieved by quadratic memory cutting plane methods.

Theorem 1 *For some $\epsilon \geq 1/\operatorname{poly}(d)$ and any $\delta \in [0, 1/4]$ the following is true: any algorithm which outputs an ϵ-optimal point with probability at least $2/3$ given first order oracle access to any 1-Lipschitz convex function must use either at least $d^{1.25-\delta}$ bits of memory or make $\tilde{\Omega}(d^{1+\frac{4}{3}\delta})$ first order queries (where the $\tilde{\Omega}$ notation hides poly-logarithmic factors in d).*

Beyond shedding light on the complexity of a fundamental memory-constrained optimization problem, we provide several tools for establishing such lower bounds. In particular, we introduce a set of properties which are sufficient for an optimization problem to exhibit a memory-lower bound and provide an information-theoretic framework to prove these lower bounds. We hope these tools are an aid to future work on the role of memory in optimization.

This work fits within the broader context of understanding fundamental resource tradeoffs for optimization and learning. For many settings, establishing (unconditional) query/time or memory/time tradeoffs is notoriously hard—perhaps akin to P vs NP (e.g. providing time lower bounds for cutting plane methods). Questions of memory/query and memory/data tradeoffs, however, have a more information theoretic nature and hence seem more approachable. Together with the increasing importance of memory considerations in large-scale optimization and learning, there is a strong case for pinning down the landscape of such tradeoffs, which may offer a new perspective on the current suite of algorithms and inform the effort to develop new ones.

1.1. Technical Overview and Contributions

To prove Theorem 1, we provide an explicit distribution over functions that is hard for any memory-constrained randomized algorithm to optimize. Though the proof requires care and we introduce

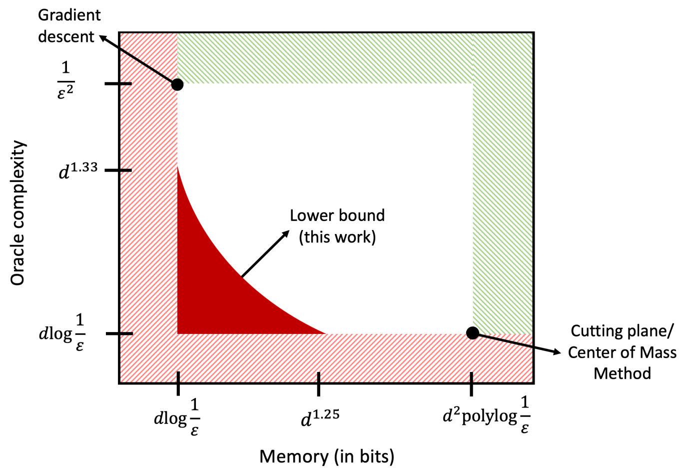

Figure 1: Tradeoffs between available memory and first-order oracle complexity for minimizing 1-Lipschitz convex functions over the unit ball (adapted from Woodworth and Srebro (2019)). The dashed red region corresponds to information-theoretic lower bounds on the memory and query-complexity. The dashed green region corresponds to known upper bounds. This work shows that the solid red region is not achievable for any algorithm.

a variety of machinery to obtain it, this lower bounding family of functions is simple to state. The function is a variant of the so-called "Nemirovski" function, which has been used to show lower bounds for highly parallel non-smooth convex optimization (Nemirovski, 1994; Bubeck et al., 2019; Balkanski and Singer, 2018).

Formally, our difficult class of functions for memory size M is constructed as follows: for some $\gamma > 0$ and some $N = \tilde{\mathcal{O}}\left(d^2/M\right)$ let $\mathbf{v}_1, \ldots, \mathbf{v}_N$ be unit vectors drawn i.i.d. from the d dimensional scaled hypercube $\mathbf{v}_i \overset{\text{i.i.d.}}{\sim} \mathsf{Unif}\left(d^{-1/2}\mathcal{H}_d\right)$ and let $\mathbf{a}_1, \ldots, \mathbf{a}_{\lfloor d/2 \rfloor}$ be drawn i.i.d. from the hypercube, $\mathbf{a}_j \sim \mathsf{Unif}\left(\mathcal{H}_d\right)$ where $\alpha\mathcal{H}_d := \{\pm\alpha\}^d$. Let $\mathbf{A} = \left(\mathbf{a}_1, \ldots, \mathbf{a}_{\lfloor d/2 \rfloor}\right)$ and define

$$F(\mathbf{x}) = (1/d^6) \max\left\{d^5 \left\|\mathbf{A}\mathbf{x}\right\|_\infty - 1, \max_{i \in [N]} \mathbf{v}_i^\top \mathbf{x} - i\gamma\right\}.$$

Rather than give a direct proof of Theorem 1 using this explicit function we provide a more abstract framework which gives broader insight into which kinds of functions could lead to non-trivial memory-constrained lower bounds, and which might lead to tighter lower bounds in the future. To that end we introduce the notion of a *memory-sensitive class* which delineates the key properties of a distribution over functions that lead to memory-constrained lower bounds. We show that for such functions, the problem of memory constrained optimization is at least as hard as the following problem of finding a set of vectors which are approximately orthogonal to another set of vectors:

Definition 2 (Informal version of the Orthogonal Vector Game) *Given $\mathbf{A} \in \{\pm 1\}^{d/2 \times d}$, the Player's objective is to return a set of k vectors $\{\mathbf{y}_1, \ldots, \mathbf{y}_k\}$ which satisfy*

1. *$\forall i \in [k], \mathbf{y}_i$ is approximately orthogonal to all the rows of $\mathbf{A}$: $\left\|\mathbf{A}\mathbf{y}_i\right\|_\infty / \left\|\mathbf{y}_i\right\|_2 \leq d^{-4}$.*
2. *The set of vectors $\{\mathbf{y}_1, \ldots, \mathbf{y}_k\}$ is robustly linearly independent: denoting $S_0 = \varnothing, S_i = \mathsf{span}(\mathbf{y}_1, \ldots, \mathbf{y}_i)$, $\left\|\mathrm{Proj}_{S_{i-1}}(\mathbf{y}_i)\right\|_2 / \left\|\mathbf{y}_i\right\|_2 \leq 1 - 1/d^2$,*

where the notation $\mathrm{Proj}_S(\mathbf{x})$ *denotes the vector in the subspace* S *which is closest in* $\|\cdot\|_2$ *to* $\mathbf{x}$*. The game proceeds as follows: The Player first gets to observe* $\mathbf{A}$ *and store a* M*-bit long* Message *about* $\mathbf{A}$*. She does not subsequently have free access to* $\mathbf{A}$*, but can adaptively make up to* m *queries as follows: for* $i \in [m]$*, based on* Message *and all previous queries and their results, she can request any row* $i \in [d/2]$ *of the matrix* $\mathbf{A}$*. Finally, she outputs a set of* k *vectors as a function of* Message *and all* m *queries and their results.*

Note that the Player can trivially win the game for $M \geq \Omega(dk), m = 0$ (by just storing a satisfactory set of k vectors in the Message) and for $M = 0, m = d/2$ (by querying all rows of $\mathbf{A}$). We show a lower bound that this is essentially all that is possible: *for* $\mathbf{A}$ *sampled uniformly at random from* $\{\pm 1\}^{d/2 \times d}$*, if* M *is a constant factor smaller than* dk*, then the Player must make at least* $d/5$ *queries to win with probability at least* $2/3$. Our analysis proceeds via an intuitive information-theoretic framework, which could have applications for showing query lower bounds for memory-constrained algorithms in other optimization and learning settings. We sketch the analysis in Section 3.2.

1.2. Related Work

Memory-sample tradeoffs for learning There is a recent line of work to understand learning under information constraints such as limited memory or communication constraints (Balcan et al., 2012; Duchi et al., 2013; Zhang et al., 2013; Garg et al., 2014; Shamir, 2014; Arjevani and Shamir, 2015; Steinhardt and Duchi, 2015; Steinhardt et al., 2016; Braverman et al., 2016; Dagan and Shamir, 2018; Dagan et al., 2019; Woodworth et al., 2021). Most of these results obtain lower bounds for the regime when the available memory is less than that required to store a single datapoint (with the notable exception of Dagan and Shamir (2018) and Dagan et al. (2019)). However the breakthrough paper Raz (2017) showed an exponential lower bound on the number of random examples needed for learning parities with memory as large as quadratic. Subsequent work extended and refined this result to multiple learning problems over finite fields (Moshkovitz and Moshkovitz, 2017; Beame et al., 2018; Moshkovitz and Moshkovitz, 2018; Kol et al., 2017; Raz, 2018; Garg et al., 2018).

Most related to our line of work is Sharan et al. (2019), which considers the continuous valued learning/optimization problem of performing linear regression given access to randomly drawn examples from an isotropic Gaussian. They show that any sub-quadratic memory algorithm for the problem needs $\Omega(d \log\log(1/\epsilon))$ samples to find an ϵ-optimal solution for $\epsilon \leq 1/d^{\Omega(\log d)}$, whereas in this regime an algorithm with memory $\tilde{\mathcal{O}}(d^2)$ can find an ϵ-optimal solution with only d examples. Since each example provides an unbiased estimate of the expected regression loss, this translates to a lower bound for convex optimization given access to a stochastic gradient oracle. However the upper bound of d examples is not a generic convex optimization algorithm/convergence rate but comes from the fact that the linear systems can be solved to the required accuracy using d examples.

There is also significant work on memory lower bounds for streaming algorithms, e.g. (Alon et al., 1999; Bar-Yossef et al., 2004; Clarkson and Woodruff, 2009; Dagan et al., 2019), where the setup is that the algorithm only gets a single-pass over a data stream.

The work of Dagan et al. (2019) mentioned earlier uses an Approximate Null-Vector Problem (ANVP) to show memory lower bounds, which shares some similarities with the Orthogonal Vector Game used in our proof (informally presented in Definition 2). In the ANVP, the objective is to find a single vector approximately orthogonal to a stream of vectors. The goal of the ANVP is similar to the goal of the Orthogonal Vector Game, which is to find a set of such vectors. However, the key difference is that the ANVP is in the (one-pass) streaming setting, whereas the Orthogonal

Vector Game allows for stronger access to the input in two senses. First, the Orthogonal Vector Game algorithms see the entire input and can store a M-bit Message about the input, and second, the algorithms adaptively query the input. The main challenge in our analysis (discussed later in Section 3.2) is to bound the power of these adaptively issued queries in conjunction with the Message.

Lower bounds for convex optimization Starting with the early work of Nemirovski and Yudin (1983), there is extensive literature on lower bounds for convex optimization. Some of the key results in this area include classical lower bounds for finding approximate minimizers of Lipschitz functions with access to a subgradient oracle (Nemirovski and Yudin, 1983; Nesterov, 2003; Braun et al., 2017), including recent progress on lower bounds for randomized algorithms (Woodworth and Srebro, 2016, 2017; Simchowitz et al., 2018; Simchowitz, 2018; Braverman et al., 2020; Sun et al., 2021). There is also work on the effect of parallelism on these lower bounds (Nemirovski, 1994; Balkanski and Singer, 2018; Diakonikolas and Guzmán, 2019; Bubeck et al., 2019). For more details, we refer the reader to surveys such as Nesterov (2003) and Bubeck (2014).

Memory-limited optimization algorithms While the focus of this work is lower bounds, there is a long line of work on developing memory-efficient optimization algorithms, including various techniques that leverage second-order structure via first-order methods such as Limited-memory-BFGS (Nocedal, 1980; Liu and Nocedal, 1989) and the conjugate gradient (CG) method for solving linear systems (Hestenes and Stiefel, 1952) and various non-linear extensions of CG (Fletcher and Reeves, 1964; Hager and Zhang, 2006) and methods based on subsampling and sketching the Hessian (Pilanci and Wainwright, 2017; Xu et al., 2020; Roosta-Khorasani and Mahoney, 2019). A related line of work is on communication-efficient optimization algorithms for distributed settings (see Jaggi et al. (2014); Shamir et al. (2014); Jordan et al. (2018); Alistarh et al. (2018); Ye and Abbe (2018) and references therein).

2. Setup and Overview of Results

We consider optimization methods for convex, Lipschitz-continuous functions $F : \mathbb{R}^d \to \mathbb{R}$ over the unit-ball[2] with access to a first-order oracle. Our goal is to understand how the (oracle) query complexity of algorithms is affected by restrictions on the memory available to the algorithm. Our results apply to a rather general definition of *memory constrained* first-order algorithms, which includes algorithms that use arbitrarily large memory at query time but can save at most M bits of information in between interactions with the first-order oracle. More formally we have Definition 3.

Definition 3 (M-bit memory constrained deterministic algorithm) *An M-bit* (memory constrained) deterministic algorithm *with first-order oracle access, $\mathcal{A}_{det}$, is the iterative execution of a sequence of functions $\{\phi_{query,t}, \phi_{update,t}\}_{t\geq 1}$, where t denotes the iteration number. The function $\phi_{query,t}$ maps the $(t-1)^{th}$ memory state of size at most M bits to the t^{th} query vector, $\phi_{query,t}(\mathsf{Memory}_{t-1}) = \mathbf{x}_t$. The algorithm is allowed to use an arbitrarily large amount of memory to execute $\phi_{query,t}$. Upon querying $\mathbf{x}_t$, some first order oracle returns $F(\mathbf{x}_t)$ and a subgradient $\mathbf{g}_{\mathbf{x}_t} \in \partial F(\mathbf{x}_t)$. The second function maps the first order information and the old memory state to a new memory state of at most M bits,* $\mathsf{Memory}_t = \phi_{update,t}\left(\mathbf{x}_t, F(\mathbf{x}_t), \mathbf{g}_{\mathbf{x}_t}, \mathsf{Memory}_{t-1}\right)$. *Again, the algorithm may use unlimited memory to execute $\phi_{update,t}$.*

2. By a blackbox-reduction in Appendix D.3 our results extend to unconstrained optimization while only losing $\mathrm{poly}(d)$ factors in the accuracy ϵ for which the lower bound applies.

Note that our definition of a memory constrained algorithm is equivalent to the definition given by Woodworth and Srebro (2019). Our analysis also allows for randomized algorithms which will often be denoted as $\mathcal{A}_{\text{rand}}$.

Definition 4 (M-bit memory constrained randomized algorithm) *An* M-bit (memory constrained) randomized algorithm *with first-order oracle access, $\mathcal{A}_{rand}$, is a deterministic algorithm with access to a string R of uniformly random bits which has length 2^d.*[3]

In what follows we use the notation $\mathcal{O}_F(\mathbf{x})$ to denote the first-order oracle which, when queried at some vector $\mathbf{x}$, returns the pair $(F(\mathbf{x}), \mathbf{g}_\mathbf{x})$ where $\mathbf{g}_\mathbf{x} \in \partial F(\mathbf{x})$ is a subgradient of F at $\mathbf{x}$. We will also refer to a sub-gradient oracle and use the overloaded notation $\mathbf{g}_F(\mathbf{x})$ to denote the oracle which simply returns $\mathbf{g}_\mathbf{x}$. It will be useful to refer to the sequence of vectors $\mathbf{x}$ queried by some algorithm $\mathcal{A}$ paired with the subgradient $\mathbf{g}$ returned by the oracle.

Definition 5 (Query sequence) *Given an algorithm $\mathcal{A}$ with access to some first-order oracle $\mathcal{O}$ we let $\mathsf{Seq}(\mathcal{A}, \mathcal{O}) = \{(\mathbf{x}_i, \mathbf{g}_i)\}_i$ denote the* query sequence *of vectors $\mathbf{x}_i$ queried by the algorithm paired with the subgradient $\mathbf{g}_i$ returned by the oracle $\mathcal{O}$.*

2.1. Proof Strategy

We describe a broad family of optimization problems which may be sensitive to memory constraints. As suggested by the Orthogonal Vector Game (Definition 2), the primitive we leverage is that finding vectors orthogonal to the rows of a given matrix requires either large memory or many queries to observe the rows of the matrix. With that intuition in mind, let $f : \mathbb{R}^d \to \mathbb{R}$ be a convex function, let $\mathbf{A} \in \mathbb{R}^{n \times d}$, let η be a scaling parameter, and let ρ be a shift parameter; define $F_{f,\mathbf{A},\eta,\rho}(\mathbf{x})$ as the maximum of $f(\mathbf{x})$ and $\eta \|\mathbf{A}\mathbf{x}\|_\infty - \rho$:

$$F_{f,\mathbf{A},\eta,\rho}(\mathbf{x}) := \max \left\{ f(\mathbf{x}), \eta \|\mathbf{A}\mathbf{x}\|_\infty - \rho \right\}. \tag{2.1}$$

We often drop the dependence of η and ρ and write $F_{f,\mathbf{A},\eta,\rho}$ simply as $F_{f,\mathbf{A}}$. Intuitively, for large enough scaling η and appropriate shift ρ, minimizing the function $F_{f,\mathbf{A}}(\mathbf{x})$ requires minimizing $f(\mathbf{x})$ close to the null space of the matrix $\mathbf{A}$. Any algorithm which uses memory $\Omega(nd)$ can learn and store $\mathbf{A}$ in $\mathcal{O}(d)$ queries so that all future queries are sufficiently orthogonal to $\mathbf{A}$; thus this memory rich algorithm can achieve the information-theoretic lower bound for minimizing $f(\mathbf{x})$ roughly constrained to the nullspace of $\mathbf{A}$.

However, if $\mathbf{A}$ is a random matrix with sufficiently large entropy then $\mathbf{A}$ cannot be compressed to fewer than $\Omega(nd)$ bits. Thus, for $n = \Omega(d)$, an algorithm which uses only memory $o(d^{2-\delta})$ bits for some constant $0 < \delta \leq 1$ cannot remember all the information about $\mathbf{A}$. Suppose the function f is such that in order to continue to observe *informative subgradients* (which we define formally in Definition 9), it is not sufficient to submit queries that belong to some small dimensional subspace of the null space of $\mathbf{A}$. Then a memory constrained algorithm must re-learn enough information about $\mathbf{A}$ in order to find a vector in the null space of $\mathbf{A}$ and make a query which returns an informative subgradient for f.

3. We remark that 2^d can be replaced by any finite-valued function of d.

2.2. Proof Components

In summary of Section 2.1, there are two critical features of $F_{f,\mathbf{A}}$ which lead to a difficult optimization problem for memory constrained algorithms. First, the distribution underlying random instances of $\mathbf{A}$ must ensure that $\mathbf{A}$ cannot be compressed. Second, the function f must be such that queries which belong to a small dimensional subspace cannot return too many informative subgradients. We formalize these features by defining a *memory-sensitive base* and a *memory-sensitive class*. We show that if $\mathbf{A}$ is drawn uniformly at random from a memory-sensitive base and if f and a corresponding first-order oracle are drawn according to a memory-sensitive class then the resulting distribution over $F_{f,\mathbf{A}}(\mathbf{x})$ and its first order oracle provides a difficult instance for memory constrained algorithms to optimize.

Definition 6 ((k, c_B)-memory-sensitive base) *Consider a sequence of sets $\{B_d\}_{d>0}$ where $B_d \subset \mathbb{R}^d$. We say $\{B_d\}$ is a (k, c_B)-*memory-sensitive base *if for any d, $|B_d| < \infty$ and for $\mathbf{h} \sim \mathsf{Unif}(B_d)$ the following hold: for any $t \in (0, 1/2]$ and any matrix $\mathbf{Z} = (\mathbf{z}_1, \ldots, \mathbf{z}_k) \in \mathbb{R}^{d\times k}$ with orthonormal columns, $\mathbb{P}(\|\mathbf{h}\|_2 > d) \le 2^{-d}$ and $\mathbb{P}(\left\|\mathbf{Z}^\top \mathbf{h}\right\|_\infty \le t) \le 2^{-c_B k}$.*

We use $\mathbf{A} \sim \mathsf{Unif}(B_d^n)$ to mean $\mathbf{A} = (\mathbf{a}_1, \ldots, \mathbf{a}_n)^\top$ where each $\mathbf{a}_i \sim \mathsf{Unif}(B_d)$. Theorem 7 shows that the sequence of hypercubes is a memory-sensitive base, proven in Appendix A.

Theorem 7 *Recall that $\mathcal{H}_d = \{\pm 1\}^d$ denotes the hypercube. The sequence $\{\mathcal{H}_d\}_{d=d_0}^{\infty}$ is a $(k, c_{\mathcal{H}})$ memory-sensitive base for all $k \in [d]$ with some absolute constant $c_{\mathcal{H}} > 0$.*

Next we consider distributions over functions f paired with some subgradient oracle $\mathbf{g}_f$. Given f and some $\mathbf{A} \sim \mathsf{Unif}(B_d^n)$ we consider minimizing $F_{f,\mathbf{A}}$ as in Eq. (2.1) with access to an induced first-order oracle:

Definition 8 (Induced First-Order Oracle $\mathcal{O}_{F_{f,\mathbf{A}}}$) *Given a subgradient oracle for f, $\mathbf{g}_f : \mathbb{R}^d \to \mathbb{R}^d$, matrix $\mathbf{A} \in \mathbb{R}^{n\times d}$, and parameters η and ρ, let*

$$\mathbf{g}_{\mathbf{A}}(\mathbf{x}) := \mathbf{a}_{i_{\min}},\ \textit{where } i_{\min} := \min\left\{i \in [n] \textit{ s.t. } \mathbf{a}_i^\top \mathbf{x} = \|\mathbf{A}\mathbf{x}\|_\infty\right\},$$
$$\mathbf{g}_{F_{f,\mathbf{A}}}(\mathbf{x}) := \begin{cases} \mathbf{g}_{\mathbf{A}}(\mathbf{x}), & \textit{if } \eta\,\|\mathbf{A}\mathbf{x}\|_\infty - \rho \ge f(\mathbf{x}),\\ \mathbf{g}_f(\mathbf{x}), & \textit{else.}\end{cases}$$

The induced first-order oracle *for $F_{f,\mathbf{A}}(\mathbf{x})$, denoted as $\mathcal{O}_{F_{f,\mathbf{A}}}$ returns the pair $(F_{f,\mathbf{A}}(\mathbf{x}), \mathbf{g}_{F_{f,\mathbf{A}}}(\mathbf{x}))$.*

We also define *informative subgradients*, formalizing the intuition described at the beginning of this section.

Definition 9 (Informative subgradients) *Given a query sequence $\mathsf{Seq}(\mathcal{A}, \mathcal{O}_{F_{f,\mathbf{A}}}) = \{(\mathbf{x}_i, \mathbf{g}_i)\}_{i=1}^T$ we construct the sub-sequence of* informative subgradients *as follows: proceed from $i = 1, \ldots, T$ and include the pair $(\mathbf{x}_i, \mathbf{g}_i)$ if and only if 1) $f(\mathbf{x}_i) > \eta\,\|\mathbf{A}\mathbf{x}_i\|_\infty - \rho$ and 2) no pair $(\mathbf{x}, \mathbf{g})$ such that $\mathbf{g} = \mathbf{g}_i$ has been selected in the sub-sequence so far. If $(\mathbf{x}_i, \mathbf{g}_i)$ is the j^{th} pair included in the sub-sequence define $t_j = i$ and we call $\mathbf{x}_i$ the j^{th} informative query and $\mathbf{g}_i$ the j^{th} informative subgradient.*

We can now proceed to the definition of a $(L, M, N, \epsilon^\star)$-memory-sensitive class.

Definition 10 **($(L, N, k, \epsilon^\star)$-memory-sensitive class)** *Let $\mathcal{F}$ be a distribution over functions $f : \mathbb{R}^d \to \mathbb{R}$ paired with a valid subgradient oracle $\mathbf{g}_f : \mathbb{R}^d \to \mathbb{R}^d$. For Lipschitz constant L, "depth" N, "round-length" k, and optimality threshold $\epsilon^\star$, we call $\mathcal{F}$ a $(L, N, k, \epsilon^\star)$-*memory-sensitive class *if one can sample from $\mathcal{F}$ with at most 2^d uniformly random bits, and there exists a choice of η and ρ (from Eq. 2.1) such that for any $\mathbf{A} \in \mathbb{R}^{n \times d}$ with rows bounded in ℓ_2-norm by d and any (potentially randomized) unbounded-memory algorithm $\mathcal{A}_{rand}$ which makes at most Nd queries: if $(f, \mathbf{g}_f) \sim \mathcal{F}$ then with probability at least 2/3 (over the randomness in $(f, \mathbf{g}_f)$ and $\mathcal{A}_{rand}$) the following hold simultaneously:*

1. ***Regularity**: $F_{f,\mathbf{A},\eta,\rho}$ is convex and L-Lipschitz.*
2. ***Robust independence of informative queries**: If $\{\mathbf{x}_{t_j}\}$ is the sequence of informative queries generated by $\mathcal{A}_{rand}$ (as per Theorem 9) and $S_j := \mathsf{span}\left(\{\mathbf{x}_{t_i} : \max(1, j-k) \leq i \leq j\}\right)$ then $\forall j \in [N]$*
$$\left\|\mathrm{Proj}_{S_{j-1}}(\mathbf{x}_{t_j})\right\|_2 / \left\|\mathbf{x}_{t_j}\right\|_2 \leq 1 - 1/d^2. \tag{2.2}$$
3. ***Approximate orthogonality**: Any query $\mathbf{x}$ with $F_{f,\mathbf{A},\eta,\rho}(\mathbf{x}) \neq \eta \|\mathbf{A}\mathbf{x}\|_\infty - \rho$ satisfies*
$$\mathbf{g}_{F_{f,\mathbf{A}}}(\mathbf{x}) = \mathbf{v}_1 \text{ or } \|\mathbf{A}\mathbf{x}\|_\infty / \|\mathbf{x}\|_2 \leq d^{-4}. \tag{2.3}$$
4. ***Necessity of informative subgradients**: If $r < N$ then for any $i \leq t_r$ (where t_r is as in Definition 9),*
$$F_{f,\mathbf{A}}(\mathbf{x}_i) - F_{f,\mathbf{A}}(\mathbf{x}^\star) \geq \epsilon^\star. \tag{2.4}$$

Informally, the **robust independence of informative queries** condition ensures that any informative query is robustly linearly independent with respect to the previous k informative queries. The **approximate orthogonality** condition ensures that any query $\mathbf{x}$ which reveals a gradient from f which is not $\mathbf{v}_1$ is approximately orthogonal to the rows of $\mathbf{A}$. Finally, the **necessity of informative subgradients** condition ensures that any algorithm which has queried only $r < N$ informative subgradients from f has optimality gap at least $\epsilon^\star$.

The following construction, discussed in the introduction, is a concrete instance of a $(d^6, N, \lceil 20M/(c_{\mathcal{H}}d)\rceil, 1/(20\sqrt{N}))$-memory-sensitive class with $N \approx (d^2/M)^{1/3}$. Theorem 12 proves that the construction is memory sensitive, and the proof appears in Section A.

Definition 11 (Nemirovski class) *For a given $\gamma > 0$ and for $i \in [N]$ draw $\mathbf{v}_i \overset{i.i.d.}{\sim} \mathsf{Unif}\left(d^{-1/2}\mathcal{H}_d\right)$ and set $f(\mathbf{x}) := \max_{i \in [N]} \mathbf{v}_i^\top \mathbf{x} - i\gamma$ and $\mathbf{g}_f(\mathbf{x}) = \mathbf{v}_{i_{\min}}$, where $i_{\min} := \min\{i \in [N] \text{ s.t. } \mathbf{v}_i^\top \mathbf{x} - i\gamma = f(\mathbf{x})\}$. Let $\mathcal{N}_{N,\gamma}$ be the distribution of $(f, \mathbf{g}_f)$, and for a fixed matrix $\mathbf{A}$ let $\mathcal{N}_{N,\gamma,\mathbf{A}}$ be the distribution of the induced first order oracle in Definition 8.*

Theorem 12 *For large enough dimension d there is an absolute constant $c > 0$ such that for any given k, the Nemirovski function class with $\gamma = (400k \log(d)/d)^{1/2}$, $L = d^6$, $N = c(d/(k \log d))^{1/3}$, and $\epsilon^\star = 1/(20\sqrt{N})$ is a $(L, N, k, \epsilon^\star)$-memory-sensitive class.*

2.3. Main Results

With the definitions from the previous section in place we may now state our main technical theorem for proving lower bounds on memory constrained optimization.

Theorem 13 (From memory-sensitivity to lower bounds) *For $d \in \mathbb{Z}_{>0}$, memory constraint $M \geq d$, and $\epsilon \in \mathbb{R}_{>0}$, let $\{B_d\}_{d>0}$ be a (k, c_B)-memory-sensitive base and let $\mathcal{F}$ be a (L, N, k, ϵ)-memory-sensitive class where $k = \lceil 20M/(c_B d) \rceil$, $N = 1/(20\epsilon)^2$, and $\epsilon \leq L/(4\sqrt{d})$. Further, let $\mathbf{A} \sim \mathsf{Unif}(B_d^n)$, let $(f, \mathbf{g}_f) \sim \mathcal{F}$, and consider $F_{f,\mathbf{A}}$ with oracle $\mathcal{O}_{F_{f,\mathbf{A}}}$ as in Definition 8. Any M-bit (potentially) randomized algorithm (as per Definition 4) which outputs an ϵ-optimal point for $F_{f,\mathbf{A}}$ with probability at least $2/3$ requires at least $c_B d^2/(10^6 \epsilon^2 M) = \Omega(c_B d^2 N/M)$ first-order oracle queries.*

Note that by Woodworth and Srebro (2019), the center of mass algorithm can output an ϵ-optimal point for any L-Lipschitz function using $\mathcal{O}\left(d^2 \log^2 (LB/\epsilon)\right)$ bits of memory and $\mathcal{O}\left(d \log(LB/\epsilon)\right)$ first-order oracle queries. Comparing this with the lower bound from Theorem 13 we see that for a given $\epsilon \leq \sqrt{L/d}$ if there exists a $(L, 1/(20\epsilon)^2, k, \epsilon)$ memory-sensitive class, then M-bit algorithms become unable to achieve the optimal quadratic memory query complexity once the memory is smaller than $o\left(d/(\epsilon^2 \log(LB/\epsilon)\right)$ many bits. Further, Theorem 13 yields a natural approach for obtaining improved lower bounds. If one exhibits a memory-sensitive class with depth $N = d$, then Theorem 13 would imply that any algorithm using memory of size at most $M = d^{2-\delta}$ requires at least $\Omega\left(d^{1+\delta}\right)$ many first-order oracle queries in order to output an ϵ-optimal point.

Theorem 1 follows by setting parameters and applying Theorems 7, 12 and 13.

Proof [Proof of Theorem 1] Consider M-bit memory constrained (potentially) randomized algorithms (as per Definition 4) where M can be written in the form $d^{1.25-\delta}$ for some $\delta \in [0, 1/4]$. By Theorem 12, for some absolute constant $c > 0$ and for d large enough and any given k, if $\gamma = \sqrt{400k \log d/d}$, $N = c(d/(k \log d))^{1/3}$, and $\epsilon = 1/(20d^6\sqrt{N})$, the Nemirovski function class from Definition 11 is $(1, N, k, \epsilon)$-memory-sensitive (where we rescaled by the Lipschitz constant $1/d^6$). Set $k = \lceil 20M/(c_{\mathcal{H}} d) \rceil$ (where $c_{\mathcal{H}}$ is as in Theorem 7). Let $(f, \mathbf{g}_f) \sim \mathcal{N}_{\gamma,N}$ and $\mathbf{A} \sim \mathsf{Unif}\left(\mathcal{H}_d^n\right)$ and suppose $\mathcal{A}_{\text{rand}}$ is an M-bit algorithm which outputs an ϵ-optimal point for $F_{f,\mathbf{A}}$ with failure probability at most $1/3$. By Theorem 13, $\mathcal{A}_{\text{rand}}$ requires at least $c_{\mathcal{H}} N d^2/M \geq c\left(d^2/M\right)^{4/3}\left(1/\log^{1/3} d\right)$ many queries. Recalling that $M = d^{1.25-\delta}$ for $\delta \in [0, 1/4]$ results in a lower bound of $\tilde{\Omega}(d^{1+\frac{4}{3}\delta})$ queries. ∎

Why $d^{4/3}$: Parameter tradeoffs in our lower bound. An idealized lower bound in this setting would arise from exhibiting an f such that to obtain any informative gradient requires $\Theta(d)$ more queries to re-learn $\mathbf{A}$ *and* in order to optimize f we need to observe at least $\Omega(d)$ such informative gradients. Our proof deviates from this ideal on both fronts. We do prove that we need $\Theta(d)$ queries to get any informative gradients, though with this number we cannot preclude getting M/d informative gradients. Second, our Nemirovski function *can* be optimized while learning only $O((d^2/M)^{1/3})$-informative gradients (there are modifications to Nemirovski that increase this number (Bubeck et al., 2019), though for those functions, informative gradients can be observed while working within a small subspace). For our analysis we have, very roughly, that $\Omega(d)$ queries are needed to observe M/d informative gradients and we must observe $(d^2/M)^{1/3}$ informative gradients to optimize the Nemirovski function. Therefore optimizing the Nemirovksi function requires $\Omega(d) \times \frac{(d^2/M)^{1/3}}{M/d} = (d^2/M)^{4/3}$ many queries; and so when $M = O(d)$ we have a lower bound of $d^{4/3}$ queries.

3. Proof Overview

Here we give the proof of Theorem 13. In Section 3.1 we present our first key idea, which relates the optimization problem in (2.1) to the Orthogonal Vector Game (which was introduced informally in Definition 2). Next, in Section 3.2, we show that winning this Orthogonal Vector Game is difficult with memory constraints.

3.1. The Orthogonal Vector Game

We formally define the Orthogonal Vector Game in Game 1 and then relate it to minimizing $F_{f,\mathbf{A}}$ (Eq. (2.1)).

Game 1: Orthogonal Vector Game

Input : d, k, m, B_d, M

1 *Oracle:* For $n \leftarrow d/2$, sample $\mathbf{A} \sim \mathsf{Unif}(B_d^n)$.

2 *Oracle:* Sample a string R of uniformly random bits of length $3 \cdot 2^d$ (the Player has read-only access to R throughout the game).

3 *Player:* Observe $\mathbf{A}$ and R, and store a M-bit message, Message, about the matrix $\mathbf{A}$.

4 **for** $i \in [m]$ **do**

5 *Player:* Based on Message, R, and any previous queries and responses, use a deterministic algorithm to submit a query $\mathbf{x}_i \in \mathbb{R}^d$.

6 *Oracle:* As the response to query $\mathbf{x}_i$, return $\mathbf{g}_i = \mathbf{g}_{\mathbf{A}}(\mathbf{x}_i)$ (see Definition 8).

7 *Player:* Let $\mathbf{X}$ and $\mathbf{G}$ be matrices with queries $\{\mathbf{x}_i, i \in [m]\}$ and responses $\{\mathbf{g}_i, i \in [m]\}$ as rows, respectively. Based on $(\mathbf{X}, \mathbf{G}, \text{Message}, R)$, use a deterministic function to return vectors $\{\mathbf{y}_1, \dots, \mathbf{y}_k\}$ to the Oracle. Let $\mathbf{Y}$ be the matrix with these vectors as rows.

8 The Player wins if the returned vectors are *successful*, where a vector $\mathbf{y}_i$ is successful if
(1) $\|\mathbf{A}\mathbf{y}_i\|_\infty / \|\mathbf{y}_i\|_2 \le d^{-4}$, and
(2) $\mathrm{Proj}_{S_{i-1}}(\mathbf{y}_i) / \|\mathbf{y}_i\|_2 \le 1 - 1/d^2$ where $S_0 := \varnothing$, and $S_i := \mathsf{span}(\mathbf{y}_1, \dots, \mathbf{y}_i)$.

Lemma 14 (Optimizing $F_{f,\mathbf{A}}$ is harder than winning the Orthogonal Vector Game) *Let $\mathbf{A} \sim \mathsf{Unif}(B_d^n)$ and $(f, \mathbf{g}_f) \sim \mathcal{F}$, where $\{B_d\}_{d>0}$ is a (k, c_B)-memory-sensitive base and $\mathcal{F}$ is a $(L, N, k, \epsilon^\star)$ memory-sensitive class. If there exists an M-bit algorithm with first-order oracle access that outputs an $\epsilon^\star$-optimal point for minimizing $F_{f,\mathbf{A}}$ using $m\lfloor N/(k+1)\rfloor$ queries with probability at least 2/3 over the randomness in the algorithm and choice of f and $\mathbf{A}$, then the Player can win the Orthogonal Vector Game with probability at least 1/3 over the randomness in the Player's strategy and $\mathbf{A}$.*

Lemma 14 serves as a bridge between the optimization problem of minimizing $F_{f,\mathbf{A}}$ and the Orthogonal Vector Game. To prove Lemma 14, we provide a strategy for the Player to win the Orthogonal Vector Game using an M-bit algorithm $\mathcal{A}_{\text{rand}}$ that minimizes $F_{f,\mathbf{A}}$ to error $\epsilon^\star$. The full strategy is given in Algorithm 2. To summarize the strategy, the Player first samples a random function/oracle pair $(f, \mathbf{g}_f)$ from the memory-sensitive class $\mathcal{F}$ using the random string R available in the Orthogonal Vector Game. She then uses $\mathcal{A}_{\text{rand}}$ to optimize $F_{f,\mathbf{A}}$ (using R to supply random bits if $\mathcal{A}_{\text{rand}}$ is randomized). She issues queries that $\mathcal{A}_{\text{rand}}$ makes to the oracle in the Orthogonal Vector Game, noting that with access to $\mathbf{g}_f(\mathbf{x})$ and the oracle's response $\mathbf{g}_{\mathbf{A}}(\mathbf{x})$, she can implement the sub-gradient oracle $\mathbf{g}_{F_{f,\mathbf{A}}}$ (Definition 8). Finally, she checks if there is a set of successful vectors among the queries she made. In Appendix B we show that the success probability of this strategy is at least $1/3$. The proof hinges on the fact that we can use the memory-sensitive properties of f to argue that informative queries made by $\mathcal{A}_{\text{rand}}$ must also be successful queries in the Orthogonal Vector Game, and that $\mathcal{A}_{\text{rand}}$ must make enough informative queries to find an $\epsilon^\star$-optimal point.

Algorithm 2: Player's strategy for the Orthogonal Vector Game

Input : $d, m, \mathcal{A}_{\text{rand}}, \mathcal{F}, R$

Part 1, *Strategy to store* Message *using* $\mathbf{A}$**:**
- Divide random string R into three equal parts R_1, R_2, R_3, each of length 2^d.
- Using R_1 if needed sample a function/oracle pair $(f, \mathbf{g}_f) \sim \mathcal{F}$.
- **for** $i \in \{1, \dots, \lfloor N/(k+1) \rfloor\}$ **do**
  - Using R_2 if needed, run $\mathcal{A}_{\text{rand}}$ to minimize $F_{f,\mathbf{A}}$ until $i(k+1)$ informative subgradients are observed. Let Memory_i be $\mathcal{A}_{\text{rand}}$'s memory state.
  - Using R_3 if needed, continue running $\mathcal{A}_{\text{rand}}$ to minimize $F_{f,\mathbf{A}}$ until $(k+1)$ more informative subgradients are observed. Let t be the total number of first-order queries made by the algorithm in this step.
  - **if** $t \le m$ **then**
    - **return** Memory_i as the Message to be stored.
- **return** `Failure`

Part 2, *Strategy to make queries***:**
- Use R_1 to resample $(f, \mathbf{g}_f)$ and set $\mathcal{A}_{\text{rand}}$'s memory state to be Message.
- **for** $i \in [m]$, **do**
  - Using R_3 if needed, run $\mathcal{A}_{\text{rand}}$ to issue query $\mathbf{x}_i$ to minimize $F_{f,\mathbf{A}}(\mathbf{x})$.
  - Submit query $\mathbf{x}_i$ to the Orthogonal Vector Game Oracle to get response $\mathbf{g}_i$
  - *Simulation of first-order oracle* $\mathcal{O}_{F_{f,\mathbf{A}}}$**:**
    - $F_{f,\mathbf{A}}(\mathbf{x}_i) = \max(\eta|\mathbf{g}_i^\top \mathbf{x}_i| - \rho, f(\mathbf{x}_i))$
    - **if** $F_{f,\mathbf{A}}(\mathbf{x}_i) \ge f(\mathbf{x}_i)$ **then** $\mathbf{g}_{F_{f,\mathbf{A}}}(\mathbf{x}_i) = \mathbf{g}_i$
    - **else** $\mathbf{g}_{F_{f,\mathbf{A}}}(\mathbf{x}_i) = \mathbf{g}_f(\mathbf{x}_i)$
  - Pass $(F_{f,\mathbf{A}}(\mathbf{x}_i), \mathbf{g}_{F_{f,\mathbf{A}}}(\mathbf{x}_i))$ to $\mathcal{A}_{\text{rand}}$ to update its state.

Part 3, *Strategy to find successful vectors based on* $(\mathbf{X}, \mathbf{G}, \mathsf{Message}, R)$**:**
- **for** *every subset* $\{\mathbf{x}_{m_1}, \dots, \mathbf{x}_{m_k}\}$ *of the queries* $\{\mathbf{x}_1, \dots, \mathbf{x}_m\}$ **do**
  - Check if $|\mathbf{g}_{m_j}^\top \mathbf{x}_{m_j}| / \left\|\mathbf{x}_{m_j}\right\| \le d^{-4}\ \forall j \in [k]$
  - Check if $\{\mathbf{x}_{m_1}, \dots, \mathbf{x}_{m_k}\}$ are robustly linearly independent (Def. 10)
  - **if** *the above two conditions are satisfied* **then**
    - **return** $\{\mathbf{x}_{m_1}, \dots, \mathbf{x}_{m_k}\}$ to the Oracle
- **return** `Failure`

3.2. Analyzing the Orthogonal Vector Game: Proof Sketch

With Lemma 14 in hand, the next step towards proving Theorem 13 is to establish a query lower bound for any memory-constrained algorithm which wins the Orthogonal Vector Game. Although our results hold in greater generality, i.e. when $\mathbf{A} \sim \mathsf{Unif}(B_d^n)$ for some memory-sensitive base $\{B_d\}_{d>0}$, we first provide some intuition by considering the concrete case where the rows of $\mathbf{A}$ are drawn uniformly at random from the hypercube. We also consider an alternative oracle model which we call the *Index-Oracle model* for which it is perhaps easier to get intuition, and for which our lower bound still holds: suppose the Player can specify any index $i \in [n]$ as a query and ask the oracle to reveal the i^{th} row of $\mathbf{A}$ (this is the setup in the informal version of the game in Definition 2). Note that this is in contrast to the oracle in the Orthogonal Vector Game which instead responds with the row of the matrix $\mathbf{A}$ which is a subgradient of the query made by the algorithm. As described in Section 1.1, the instance where $M = \Omega(kd), m = 0$ and the instance where $M = 0, m = n$ are trivially achievable for the Player. We show that these trivial strategies are essentially all that is possible:

Theorem 15 (Informal version of Theorem 16) *For some constant $c > 0$ and any $k \geq \Omega(\log(d))$, if $M \leq cdk$ and the Player wins the Orthogonal Vector Game with probability at least $1/3$, then $m \geq d/5$.*

Note that when $m = 0$, it is not difficult to show that the Player requires memory roughly $\Omega(kd)$ to win the game with any decent probability. This is because each vector which is approximately orthogonal to the rows of $\mathbf{A}$ will not be compressible below $\Omega(d)$ bits with high probability, since rows of $\mathbf{A}$ are drawn uniformly at random from the hypercube. Therefore the Player must use $M \geq \Omega(kd)$ to store k such vectors. The main challenge in the analysis is to bound the power of additional, adaptively issued queries. In particular, since observing every row $\mathbf{a}_i$ gives $\Theta(d)$ bits of information about $\mathbf{A}$, and k successful vectors will only have about $O(kd)$ bits of information about $\mathbf{A}$, perhaps only observing k additional rows of $\mathbf{A}$ is enough to win the game. Slightly more concretely, why can't the Player store some $M \ll kd$ bits of information about $\mathbf{A}$, such that by subsequently strategically requesting $m \ll n$ rows of $\mathbf{A}$ she now knows k vectors which are approximately orthogonal to the rows of $\mathbf{A}$? Our result guarantees that such a strategy is not possible.

We now give some intuition for our analysis. The underlying idea is to calculate the mutual information between $\mathbf{A}$ and $\mathbf{Y}$ conditioned on a fixed random string R and the information $\mathbf{G}$ gleaned by the Player (in our proof we slightly augment the matrix $\mathbf{G}$ and condition on that, but we ignore that detail for this proof sketch). We will use the notation $I(\mathbf{X}; \mathbf{Y}|\mathbf{Z})$ to denote the mutual information between $\mathbf{X}$ and $\mathbf{Y}$ conditioned on $\mathbf{Z}$ and $H(\mathbf{X})$ to denote the entropy of $\mathbf{X}$ (see Cover and Thomas (1991)). Our analysis relies on computing a suitable upper bound and lower bound for $I(\mathbf{A}; \mathbf{Y}|\mathbf{G}, R)$. We begin with the upper bound. We first argue that $\mathbf{Y}$ can be computed just based on $\mathbf{G}$, Message and R (without using $\mathbf{X}$),

$$Y = g(\mathbf{G}, \mathsf{Message}, R), \text{for some function } g.$$

The proof of this statement relies on the fact that the Player's strategy is deterministic conditioned on the random string R. Now using the data processing inequality (Cover and Thomas, 1991) we can say,

$$I(\mathbf{A}; \mathbf{Y}|\mathbf{G}, R) \leq I(\mathbf{A}; \mathbf{G}, \mathsf{Message}, R|\mathbf{G}, R) = I(\mathbf{A}; \mathsf{Message}|\mathbf{G}, R).$$

Next we use the definition of mutual information, the fact that conditioning only reduces entropy, and Shannon's source coding theorem (Cover and Thomas, 1991) to write,

$$\begin{aligned} I(\mathbf{A}; \mathsf{Message}|\mathbf{G}, R) &\leq H(\mathsf{Message}|\mathbf{G}, R) \leq H(\mathsf{Message}) \leq M \\ &\implies I(\mathbf{A}; \mathbf{Y}|\mathbf{G}, R) \leq M. \end{aligned} \tag{3.1}$$

On the other hand, we argue that if $\mathbf{Y}$ is *successful* as per the Orthogonal Vector Game with probability $1/3$ then there must be a larger amount of mutual information between $\mathbf{Y}$ and $\mathbf{A}$ after conditioning on $\mathbf{G}$ and R. Recall that $\mathbf{Y}$ contains k robustly linearly independent vectors in order to win the Orthogonal Vector Game, assume for now that $\mathbf{Y}$ is also orthonormal. Then, we may use the fact that the hypercube is a memory-sensitive base (Definition 6) to derive that for some constant $c > 0$

$$\left|\left\{\mathbf{a} \in \{\pm 1\}^d \text{ s.t. } \|\mathbf{Y}\mathbf{a}\|_\infty \leq d^{-4}\right\}\right| \leq 2^{d-ck}. \tag{3.2}$$

Since $\mathbf{G}$ contains m rows of $\mathbf{A}$, only the remaining $(n-m)$ rows of $\mathbf{A}$ are random after conditioning on $\mathbf{G}$. Assume for now that the remaining $(n-m)$ rows are independent of $\mathbf{G}$ (as would be true under the Index-Oracle model defined earlier in this section). Then for these remaining rows, we use Eq. (3.2) to bound their entropy conditioned on $\mathbf{Y}$:

$$\begin{aligned} I(\mathbf{A}; \mathbf{Y}|\mathbf{G}, R) &= H(\mathbf{A}|\mathbf{G}, R) - H(\mathbf{A}|\mathbf{Y}, \mathbf{G}, R) \geq H(\mathbf{A}|\mathbf{G}, R) - H(\mathbf{A}|\mathbf{Y}) \\ &\geq (n-m)d - (n-m)(d-ck) = c(n-m)k = c(d/2-m)k, \end{aligned} \tag{3.3}$$

where we chose $n = d/2$. Now combining Eq. (3.1) and Eq. (3.3) we obtain a lower bound on the query complexity, $m \geq (d/2) - M/(ck)$. Thus if $M \ll cdk$, then $m = \Omega(d)$.

While this proof sketch contains the skeleton of our proof, as also hinted in the sketch it oversimplifies many points. For example, note that the definition of a memory-sensitive base in Definition 6 requires that $\mathbf{Y}$ be orthonormal to derive Eq. (3.2), but successful vectors only satisfy a robust linear independence property. Additionally, in deriving Eq. (3.3) we implicitly assumed that $\mathbf{G}$ does not reduce the entropy of the rows of $\mathbf{A}$ which are not contained in it. This is not true for the Orthogonal Vector Game, since every response $\mathbf{g}_i$ of the oracle is a subgradient of $\|\mathbf{A}\mathbf{x}\|_\infty$ and therefore depends on *all* the rows of the matrix $\mathbf{A}$. Our full proof which handles these issues and dependencies appears in Section C. As noted before, our lower bound holds not only for the subgradient oracle, but also when the oracle response $\mathbf{g}_i$ can be an arbitrary (possibly randomized) function from $\mathbb{R}^d \to \{\mathbf{a}_1, \ldots, \mathbf{a}_n\}$ (for example, the model where the Player can query any row $\mathbf{a}_i$ of the matrix $\mathbf{A}$). We state the lower bound for the Orthogonal Vector Game which is the main result from Section C below.

Theorem 16 *Suppose for a memory-sensitive base $\{B_d\}_{d=1}^\infty$ with constant $c_B > 0$, $\mathbf{A} \sim \mathsf{Unif}(B_d^n)$. Given $\mathbf{A}$ let the oracle response $\mathbf{g}_i$ to any query $\mathbf{x}_i \in \mathbb{R}^d$ be any (possibly randomized) function from $\mathbb{R}^d \to \{\mathbf{a}_1, \ldots, \mathbf{a}_{d/2}\}$, where $\mathbf{a}_i$ is the i^{th} row of $\mathbf{A}$ (note this includes the subgradient response in the Orthogonal Vector Game). Set $k = \lceil 60M/(c_B d) \rceil$ and assume $k \geq \lceil 30 \log(4d)/c_B \rceil$. For these values of k and M, if the Player wins the Orthogonal Vector Game with probability at least $1/3$, then $m \geq d/5$.*

3.3. Putting things together: Proof of main theorem

Here we combine the results of the previous subsections to prove Theorem 13. Very roughly, by Lemma 14, any algorithm for finding an $\epsilon^\star$-optimal point with probability $2/3$ can be used to win the Orthogonal Vector Game with probability $1/3$. Therefore, combining Theorem 16 and Lemma 14, and noting $k \approx M/d$, any algorithm for finding an ϵ-optimal point with probability $2/3$ must use $m\lfloor N/(k+1)\rfloor \geq cNd^2/M$ queries.

Proof [Proof of Theorem 13] We take $\epsilon = 1/(20\sqrt{N})$ throughout the proof. By (Woodworth and Srebro, 2019, Theorem 5), any algorithm for finding a ϵ-optimal point for a L-Lipschitz function in the unit ball must use memory at least $d\log(\frac{L}{2\epsilon})$. Therefore, since $\epsilon \leq L/(4\sqrt{d})$, without loss of generality we may assume that $k = \lceil 60M/c_B d\rceil \geq \lceil 30\log(4d)/c_B\rceil$ since $M = d\log(\frac{L}{2\epsilon}) \geq d\log(2\sqrt{d}) \geq (d/2)\log(4d)$. Therefore by Theorem 16, if the Player wins the Orthogonal Vector Game with failure probability at most $2/3$, then the number of queries m satisfy $m \geq d/5$. By Lemma 14, any algorithm for finding an ϵ-optimal point with failure probability at most $1/3$ can be used to win the Orthogonal Vector Game with failure probability at most $2/3$. Therefore, combining Theorem 16 and Lemma 14, any algorithm for finding an ϵ-optimal point with failure probability $1/3$ must use

$$m\lfloor N/(k+1)\rfloor \geq \frac{c_B N d^2(1-2/3)}{(2\cdot 20\cdot 5)M}$$

queries. Therefore if $\epsilon = 1/(20\sqrt{N})$ and $c = c_B/(2\cdot 20\cdot 5\cdot 3\cdot 20^2) \geq c_B/10^6$, any memory constrained algorithm requires at least $cd^2/(\epsilon^2 M)$ first-order queries. ■

4. Discussion

Our work is a key step towards establishing optimal memory/query trade-offs for optimization. An immediate open problem suggested by our work is that of improving our bounds, perhaps even to match the quadratic memory upper bound of cutting plane methods. Section 2.3 provides some insight into how one might establish a tighter result using our proof techniques.

Relaxing the type of functions and oracles for which we can prove memory-constrained lower bounds is a natural further step. Interestingly, the proofs in this paper rely on the assumption that our function is Lipschitz (rather than smooth) and that at a point queried we only observe the function's value and a single subgradient (rather than the set of all subgradients or all information about the function in a neighborhood) and consequently standard smoothing techniques do not readily apply (Diakonikolas and Guzmán, 2019; Carmon et al., 2021). Extending our lower bounds to smooth functions and beyond would help illuminate the necessity of large memory footprints for prominent memory-intensive optimization methods (e.g. interior point methods and quasi-Newton methods).

Acknowledgments

Annie Marsden and Gregory Valiant were supported by NSF Awards CCF-1704417, CCF-1813049, Frontier Award 1804222 and DOE award DE-SC0019205. Annie Marsden was also supported by J. Duchi's Office of Naval Research Award YIP N00014-19-2288 and NSF Award HDR 1934578. Aaron Sidford was supported by a Microsoft Research Faculty Fellowship, NSF CAREER Award CCF-1844855, NSF Grant CCF-1955039, a PayPal research award, and a Sloan Research Fellowship.

References

Dan Alistarh, Torsten Hoefler, Mikael Johansson, Nikola Konstantinov, Sarit Khirirat, and Cédric Renggli. The convergence of sparsified gradient methods. *Advances in Neural Information Processing Systems*, 31, 2018.

Noga Alon, Yossi Matias, and Mario Szegedy. The space complexity of approximating the frequency moments. *Journal of Computer and system sciences*, 58(1):137–147, 1999.

Kurt M Anstreicher. The volumetric barrier for semidefinite programming. *Mathematics of Operations Research*, 25(3):365–380, 2000.

Yossi Arjevani and Ohad Shamir. Communication complexity of distributed convex learning and optimization. *Advances in neural information processing systems*, 28, 2015.

David S Atkinson and Pravin M Vaidya. A cutting plane algorithm for convex programming that uses analytic centers. *Mathematical Programming*, 69(1):1–43, 1995.

Maria Florina Balcan, Avrim Blum, Shai Fine, and Yishay Mansour. Distributed learning, communication complexity and privacy. In *Conference on Learning Theory*, pages 26–1. JMLR Workshop and Conference Proceedings, 2012.

Eric Balkanski and Yaron Singer. Parallelization does not accelerate convex optimization: Adaptivity lower bounds for non-smooth convex minimization. *arXiv preprint arXiv:1808.03880*, 2018.

Ziv Bar-Yossef, Thathachar S Jayram, Ravi Kumar, and D Sivakumar. An information statistics approach to data stream and communication complexity. *Journal of Computer and System Sciences*, 68(4):702–732, 2004.

Paul Beame, Shayan Oveis Gharan, and Xin Yang. Time-space tradeoffs for learning finite functions from random evaluations, with applications to polynomials. In *Conference On Learning Theory*, pages 843–856. PMLR, 2018.

Dimitris Bertsimas and Santosh Vempala. Solving convex programs by random walks. *Journal of the ACM (JACM)*, 51(4):540–556, 2004.

Gábor Braun, Cristóbal Guzmán, and Sebastian Pokutta. Lower bounds on the oracle complexity of nonsmooth convex optimization via information theory. *IEEE Transactions on Information Theory*, 63(7):4709–4724, 2017.

Mark Braverman, Ankit Garg, Tengyu Ma, Huy L Nguyen, and David P Woodruff. Communication lower bounds for statistical estimation problems via a distributed data processing inequality. In *Proceedings of the forty-eighth annual ACM symposium on Theory of Computing*, pages 1011–1020, 2016.

Mark Braverman, Elad Hazan, Max Simchowitz, and Blake Woodworth. The gradient complexity of linear regression. In *Conference on Learning Theory*, pages 627–647. PMLR, 2020.

Sébastien Bubeck. Convex optimization: Algorithms and complexity. *arXiv preprint arXiv:1405.4980*, 2014.

Sébastien Bubeck, Qijia Jiang, Yin Tat Lee, Yuanzhi Li, Aaron Sidford, et al. Complexity of highly parallel non-smooth convex optimization. *Advances in neural information processing systems*, 2019.

Yair Carmon, Arun Jambulapati, Yujia Jin, and Aaron Sidford. Thinking inside the ball: Near-optimal minimization of the maximal loss. In *Conference on Learning Theory*, pages 866–882. PMLR, 2021.

Kenneth L Clarkson and David P Woodruff. Numerical linear algebra in the streaming model. In *Proceedings of the forty-first annual ACM symposium on Theory of computing*, pages 205–214, 2009.

T. M. Cover and J. A. Thomas. *Elements of Information Theory*. Wiley, 1991.

Yuval Dagan and Ohad Shamir. Detecting correlations with little memory and communication. In *Conference On Learning Theory*, pages 1145–1198. PMLR, 2018.

Yuval Dagan, Gil Kur, and Ohad Shamir. Space lower bounds for linear prediction in the streaming model. In *Conference on Learning Theory*, pages 929–954. PMLR, 2019.

Jelena Diakonikolas and Cristóbal Guzmán. Lower bounds for parallel and randomized convex optimization. In *Conference on Learning Theory*, pages 1132–1157. PMLR, 2019.

John C Duchi, Michael I Jordan, and Martin J Wainwright. Local privacy and statistical minimax rates. In *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*, pages 429–438. IEEE, 2013.

R. Fletcher and C. M. Reeves. Function minimization by conjugate gradients. *The Computer Journal*, 7(2), 1964.

Ankit Garg, Tengyu Ma, and Huy Nguyen. On communication cost of distributed statistical estimation and dimensionality. *Advances in Neural Information Processing Systems*, 27, 2014.

Sumegha Garg, Ran Raz, and Avishay Tal. Extractor-based time-space lower bounds for learning. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, pages 990–1002, 2018.

Martin Grötschel, László Lovász, and Alexander Schrijver. *Geometric algorithms and combinatorial optimization*, volume 2. Springer Science & Business Media, 2012.

William W Hager and Hongchao Zhang. A survey of nonlinear conjugate gradient methods. *Pacific Journal of Optimization*, 2(1), 2006.

David Lee Hanson and Farroll Tim Wright. A bound on tail probabilities for quadratic forms in independent random variables. *The Annals of Mathematical Statistics*, 42(3):1079–1083, 1971.

M.R. Hestenes and E. Stiefel. Methods of conjugate gradients for solving linear systems. *Journal of Research of the National Bureau of Standards*, 49(6), 1952.

Martin Jaggi, Virginia Smith, Martin Takác, Jonathan Terhorst, Sanjay Krishnan, Thomas Hofmann, and Michael I Jordan. Communication-efficient distributed dual coordinate ascent. *Advances in neural information processing systems*, 27, 2014.

Albert Xin Jiang and Kevin Leyton-Brown. Polynomial-time computation of exact correlated equilibrium in compact games. In *Proceedings of the 12th ACM conference on Electronic commerce*, pages 119–126, 2011.

Haotian Jiang. Minimizing convex functions with integral minimizers. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 976–985. SIAM, 2021.

Haotian Jiang, Yin Tat Lee, Zhao Song, and Sam Chiu-wai Wong. An improved cutting plane method for convex optimization, convex-concave games, and its applications. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*, pages 944–953, 2020.

Michael I Jordan, Jason D Lee, and Yun Yang. Communication-efficient distributed statistical inference. *Journal of the American Statistical Association*, 2018.

Leonid G Khachiyan, Sergei Pavlovich Tarasov, and II Erlikh. The method of inscribed ellipsoids. In *Soviet Math. Dokl*, volume 37, pages 226–230, 1988.

Gillat Kol, Ran Raz, and Avishay Tal. Time-space hardness of learning sparse parities. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, pages 1067–1080, 2017.

Yin Tat Lee, Aaron Sidford, and Sam Chiu-wai Wong. A faster cutting plane method and its implications for combinatorial and convex optimization. In *2015 IEEE 56th Annual Symposium on Foundations of Computer Science*, pages 1049–1065. IEEE, 2015.

Anatoly Yur'evich Levin. An algorithm for minimizing convex functions. In *Doklady Akademii Nauk*, volume 160, pages 1244–1247. Russian Academy of Sciences, 1965.

Dong C. Liu and Jorge Nocedal. On the limited memory BFGS method for large scale optimization. *Mathematical Programming*, 45(1-3), 1989.

S Thomas McCormick. Submodular function minimization. *Handbooks in operations research and management science*, 12:321–391, 2005.

Dana Moshkovitz and Michal Moshkovitz. Mixing implies lower bounds for space bounded learning. In *Conference on Learning Theory*, pages 1516–1566. PMLR, 2017.

Dana Moshkovitz and Michal Moshkovitz. Entropy samplers and strong generic lower bounds for space bounded learning. In *9th Innovations in Theoretical Computer Science Conference*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2018.

Arkadi Nemirovski. On parallel complexity of nonsmooth convex optimization. *Journal of Complexity*, 10(4):451–463, 1994.

Arkadij Nemirovski and David Yudin. Problem complexity and method efficiency in optimization. 1983.

Ju E Nesterov. Self-concordant functions and polynomial-time methods in convex programming. *Report, Central Economic and Mathematic Institute, USSR Acad. Sci*, 1989.

Yurii Nesterov. *Introductory lectures on convex optimization: A basic course*, volume 87. Springer Science & Business Media, 2003.

Jorge Nocedal. Updating quasi-Newton matrices with limited storage. *Mathematics of Computation*, 35(151), 1980.

Christos H Papadimitriou and Tim Roughgarden. Computing correlated equilibria in multi-player games. *Journal of the ACM (JACM)*, 55(3):1–29, 2008.

Mert Pilanci and Martin J Wainwright. Newton sketch: A near linear-time optimization algorithm with linear-quadratic convergence. *SIAM Journal on Optimization*, 27(1):205–245, 2017.

Ran Raz. A time-space lower bound for a large class of learning problems. In *2017 IEEE 58th Annual Symposium on Foundations of Computer Science*, pages 732–742. IEEE, 2017.

Ran Raz. Fast learning requires good memory: A time-space lower bound for parity learning. *Journal of the ACM*, 66(1):1–18, 2018.

Farbod Roosta-Khorasani and Michael W Mahoney. Sub-sampled newton methods. *Mathematical Programming*, 174(1):293–326, 2019.

Mark Rudelson and Roman Vershynin. Hanson-wright inequality and sub-gaussian concentration. *Electronic Communications in Probability*, 18:1–9, 2013.

Ohad Shamir. Fundamental limits of online and distributed algorithms for statistical learning and estimation. *Advances in Neural Information Processing Systems*, 27, 2014.

Ohad Shamir, Nati Srebro, and Tong Zhang. Communication-efficient distributed optimization using an approximate newton-type method. In *International conference on machine learning*, pages 1000–1008. PMLR, 2014.

Vatsal Sharan, Aaron Sidford, and Gregory Valiant. Memory-sample tradeoffs for linear regression with small error. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, pages 890–901, 2019.

Naum Z Shor. Cut-off method with space extension in convex programming problems. *Cybernetics*, 13(1):94–96, 1977.

Max Simchowitz. On the randomized complexity of minimizing a convex quadratic function. *arXiv preprint arXiv:1807.09386*, 2018.

Max Simchowitz, Ahmed El Alaoui, and Benjamin Recht. Tight query complexity lower bounds for pca via finite sample deformed wigner law. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, pages 1249–1259, 2018.

Kartik Sivaramakrishnan and John Mitchell. Properties of a cutting plane method for semidefinite programming. *Pacific Journal of Optimization*, 8:779–802, 10 2012.

Jacob Steinhardt and John Duchi. Minimax rates for memory-bounded sparse linear regression. In *Conference on Learning Theory*, pages 1564–1587. PMLR, 2015.

Jacob Steinhardt, Gregory Valiant, and Stefan Wager. Memory, communication, and statistical queries. In *Conference on Learning Theory*, pages 1490–1516. PMLR, 2016.

Xiaoming Sun, David P Woodruff, Guang Yang, and Jialin Zhang. Querying a matrix through matrix-vector products. *ACM Transactions on Algorithms (TALG)*, 17(4):1–19, 2021.

Pravin M Vaidya. A new algorithm for minimizing convex functions over convex sets. In *30th Annual Symposium on Foundations of Computer Science*, pages 338–343. IEEE Computer Society, 1989.

Roman Vershynin. *High-dimensional probability: An introduction with applications in data science*, volume 47. Cambridge university press, 2018.

Martin J Wainwright. *High-dimensional statistics: A non-asymptotic viewpoint*, volume 48. Cambridge University Press, 2019.

Blake Woodworth and Nathan Srebro. Lower bound for randomized first order convex optimization. *arXiv preprint arXiv:1709.03594*, 2017.

Blake Woodworth and Nathan Srebro. Open problem: The oracle complexity of convex optimization with limited memory. In *Conference on Learning Theory*, pages 3202–3210. PMLR, 2019.

Blake E Woodworth and Nati Srebro. Tight complexity bounds for optimizing composite objectives. *Advances in neural information processing systems*, 29, 2016.

Blake E Woodworth, Brian Bullins, Ohad Shamir, and Nathan Srebro. The min-max complexity of distributed stochastic convex optimization with intermittent communication. In *Conference on Learning Theory*, pages 4386–4437. PMLR, 2021.

Peng Xu, Fred Roosta, and Michael W Mahoney. Newton-type methods for non-convex optimization under inexact hessian information. *Mathematical Programming*, 184(1):35–70, 2020.

Min Ye and Emmanuel Abbe. Communication-computation efficient gradient coding. In *International Conference on Machine Learning*, pages 5610–5619. PMLR, 2018.

DB Yudin and Arkadi S Nemirovskii. Informational complexity and efficient methods for the solution of convex extremal problems. *Matekon*, 13(2):22–45, 1976.

Yuchen Zhang, John C Duchi, Michael I Jordan, and Martin J Wainwright. Information-theoretic lower bounds for distributed statistical estimation with communication constraints. In *Conference on Neural Information Processing Systems*, pages 2328–2336. Citeseer, 2013.

Appendix A. The Hypercube and Nemirovski function class are Memory-Sensitive

In this section we prove that the hypercube and the Nemirovski function class have the stated memory-sensitive properties. The proofs will rely on a few concentration bounds stated and proved in Appendix D.1.

A.1. The Hypercube is a Memory-Sensitive Base

Here we prove that the hypercube is a memory sensitive basis. Recall Theorem 7.

Theorem 7 *Recall that $\mathcal{H}_d = \{\pm 1\}^d$ denotes the hypercube. The sequence $\{\mathcal{H}_d\}_{d=d_0}^{\infty}$ is a $(k, c_{\mathcal{H}})$ memory-sensitive base for all $k \in [d]$ with some absolute constant $c_{\mathcal{H}} > 0$.*

Proof [Proof of Theorem 7] First observe that for any $\mathbf{h} \in \mathcal{H}_d$, $\|\mathbf{h}\|_2 = \sqrt{d}$. Next let $\mathbf{Z} = (\mathbf{z}_1, \ldots, \mathbf{z}_k) \in \mathbb{R}^{d \times k}$ be k orthonormal d-dimensional vectors and let $\mathbf{h} \sim \mathsf{Unif}(\mathcal{H}_d)$. Note that each coordinate of $\mathbf{h}$ is sub-Gaussian with sub-Gaussian norm bounded by 2 and $\mathbb{E}[\mathbf{h}\mathbf{h}^\top] = \mathbf{I}_d$. By Lemma 40 in Appendix D.1 there is an absolute constant $c_{\text{hw}} > 0$ such that for any $s \geq 0$,

$$\mathbb{P}\left(k - \left\|\mathbf{Z}^\top \mathbf{h}\right\|_2^2 > s\right) \leq \exp\left(-c_{\text{hw}} \min\left\{\frac{s^2}{16k}, \frac{s}{4}\right\}\right).$$

Taking $s = k/2$ we observe,

$$\mathbb{P}\left(\left\|\mathbf{Z}^\top \mathbf{h}\right\|_2 < \sqrt{k/2}\right) \leq \mathbb{P}\left(\left|\left\|\mathbf{Z}^\top \mathbf{h}\right\|_2^2 - k\right| > k/2\right) \leq \exp\left(-c_{\text{hw}} k/64\right).$$

Noting that $\left\|\mathbf{Z}^\top \mathbf{h}\right\|_2 \leq \sqrt{k}\left\|\mathbf{Z}^\top \mathbf{h}\right\|_\infty$ we must have that for any $t \leq 1/\sqrt{2}$,

$$\mathbb{P}\left(\left\|\mathbf{Z}^\top \mathbf{h}\right\|_\infty \leq t\right) \leq \exp\left(-c_{\text{hw}} k/64\right) \leq 2^{-ck},$$

for $c = c_{\text{hw}}/(64 \log_e 2)$. ∎

A.2. The Nemirovski function class is Memory-Sensitive

Recall we define $\mathcal{N}_{N,\gamma}$ to be the distribution of $(f, \mathbf{g}_f)$ where for $i \in [N]$ draw $\mathbf{v}_i \overset{\text{i.i.d.}}{\sim} (1/\sqrt{d})\mathsf{Unif}\left(\mathcal{H}_d\right)$ and set

$$f(\mathbf{x}) := \max_{i \in [N]} \mathbf{v}_i^\top \mathbf{x} - i\gamma \quad \text{and} \quad \mathbf{g}_f(\mathbf{x}) = \mathbf{v}_{i_{\min}} \quad \text{for} \quad i_{\min} := \min\left\{i \in [N] \text{ s.t. } \mathbf{v}_i^\top \mathbf{x} - i\gamma = f(\mathbf{x})\right\}.$$

We will show that for $k \leq \lceil 20M/(c_{\mathcal{H}} d)\rceil$, $\gamma = \sqrt{\frac{ck\log(d)}{d}}$, where $c = 400$ is a fixed constant, $\eta = d^5$, $\rho = 1$, if $N \leq (1/32\gamma)^{2/3}$ then $\mathcal{N}_{N,\gamma}$ is a $(d^6, N, k, 1/(20\sqrt{N}))$-memory-sensitive class. To show the required properties of the Nemirovski function, we define *the Resisting oracle* which iteratively constructs the Nemirovski function. The Resisting oracle will be defined via a sequence of N *Phases*, where the j^{th} Phase ends and the next begins when the algorithm makes a query which reveals the j^{th} Nemirovski vector. We begin with some definitions for these Phases.

Definition 17 *For a fixed* $\mathbf{A}$ *and vectors* $\{\mathbf{v}_i, i \in [j]\}$, *the* Nemirovski-induced function for Phase j *is*

$$F^{(j)}_{\mathcal{N},\mathbf{A}}(\mathbf{x}) := \max\left\{\max_{i\in[j]}\left(\mathbf{v}_i^\top \mathbf{x} - i\gamma\right), \eta \left\|\mathbf{A}\mathbf{x}\right\|_\infty - \rho\right\} \qquad \textit{with} \qquad F^{(0)}_{\mathcal{N},\mathbf{A}}(\mathbf{x}) := \eta \left\|\mathbf{A}\mathbf{x}\right\|_\infty - \rho.$$

Analogous to Definition 11, we define the following subgradient oracle,

$$\mathbf{g}_f^{(j)}(\mathbf{x}) := \mathbf{v}_{k_{\min}}, \text{ where } k_{\min} := \min\left\{i \in [j] \text{ s.t. } \mathbf{v}_i^\top \mathbf{x} - i\gamma = F^{(j)}_{\mathcal{N},\mathbf{A}}(\mathbf{x})\right\}.$$

Following Definition 8, this defines a subgradient oracle $\mathbf{g}_F^{(j)}(\mathbf{x})$ for the induced function $F^{(j)}_{\mathcal{N},\mathbf{A}}(\mathbf{x})$.

$$\mathbf{g}_F^{(j)}(\mathbf{x}) := \begin{cases} \mathbf{g}_{\mathbf{A}}(\mathbf{x}), \text{ if } \eta \left\|\mathbf{A}\mathbf{x}\right\|_\infty - \rho \geq f(\mathbf{x}), \\ \mathbf{g}_f^{(j)}(\mathbf{x}), \text{ else.} \end{cases}$$

The first-order oracle $\mathcal{O}_{F^{(j)}_{\mathcal{N},\mathbf{A}}}$ which returns the pair $(F^{(j)}_{\mathcal{N},\mathbf{A}}(\mathbf{x}), \mathbf{g}_F^{(j)}(\mathbf{x}))$ will be used by the Resisting oracle in Phase j. With these definitions in place, we can now define the Resisting oracle in Algorithm 3.

Algorithm 3: The Resisting oracle

// Initialize the oracle (before calling query) by storing the input, setting the current phase number ($j = 1$), and computing $\mathbf{v}_1$

initialize($\mathbf{A} \in \mathbb{R}^{n\times d}, N > 0, d > 0$):

  Save $\mathbf{A} \in \mathbb{R}^{n\times d}$, $N > 0$, and $d > 0$

  Set $\mathbf{v}_1 \overset{\text{i.i.d.}}{\sim} (1/\sqrt{d})\mathsf{Unif}\left(\mathcal{H}_d\right)$ and $j \leftarrow 1$

// Compute the oracle output for query vector $\mathbf{x}$ (after having called initialize)

query($\mathbf{x} \in B_2^d$):

  $(\alpha, \mathbf{v}) \leftarrow (F^{(j)}_{\mathcal{N},\mathbf{A}}(\mathbf{x}), \mathbf{g}_F^{(j)}(\mathbf{x}))$ // store oracle response

  if $\mathbf{g}_F^{(j)}(\mathbf{x}) = \mathbf{v}_j$ **then** // Check if time for Phase $j + 1$

    Begin Phase $j + 1$ of Resisting oracle by setting $\mathbf{v}_{j+1} \overset{\text{i.i.d.}}{\sim} (1/\sqrt{d})\mathsf{Unif}\left(\mathcal{H}_d\right)$ and $j \leftarrow j + 1$

  if *final query to oracle and* $j < N$ **then**

    For all $j' > j$, Resisting oracle draws $\mathbf{v}_{j'} \overset{\text{i.i.d.}}{\sim} (1/\sqrt{d})\mathsf{Unif}\left(\mathcal{H}_d\right)$.

  return: $(\alpha, \mathbf{v})$

As we prove in the following simple lemma, for any fixed algorithm the distribution over the responses from $\mathcal{O}_{F^{(N)}_{\mathcal{N},\mathbf{A}}}$ is the same as the distribution over the responses from the oracle $\mathcal{N}_{N,\gamma,\mathbf{A}}$ for the Nemirovksi function defined in Definition 11. Therefore, it will be sufficient to analyze the Resisting oracle and $\mathcal{O}_{F^{(N)}_{\mathcal{N},\mathbf{A}}}$.

Lemma 18 *The vectors $\{\mathbf{v}_i, i \in [N]\}$ for the final induced oracle $\mathcal{O}_{F^{(N)}_{\mathcal{N},\mathbf{A}}}(\mathbf{x}) = (F^{(N)}_{\mathcal{N},\mathbf{A}}(\mathbf{x}), \mathbf{g}_F^{(N)}(\mathbf{x}))$ are sampled from the same distribution as the vectors $\{\mathbf{v}_i, i \in [N]\}$ for the Nemirovski function $\mathcal{N}_{N,\gamma,\mathbf{A}}$ (Definition 11). Moreover, for the same setting of the vectors $\{\mathbf{v}_i, i \in [N]\}$, any fixed matrix $\mathbf{A}$ and any fixed algorithm, the responses of both these oracles are identical.*

Proof

The proof follows from definitions. For the first part, note that the Nemirovski vectors in Algorithm 3 are sampled at random independent of the Algorithm's queries, and from the same distribution as in Definition 11. For the second part, note that by definition, for a fixed set of vectors $\{\mathbf{v}_i, i \in [N]\}$ and matrix $\mathbf{A}$, the first-order oracle $\mathcal{O}_{F^{(N)}_{\mathcal{N},\mathbf{A}}}$ is identical to $\mathcal{N}_{N,\gamma,\mathbf{A}}$ from Definition 11.
■

Our main goal now is to prove the following two statements (1) with high probability, the Resisting oracle is consistent with itself, i.e. its outputs are the same as the outputs of $\mathcal{O}_{F^{(N)}_{\mathcal{N},\mathbf{A}}} = (F^{(N)}_{\mathcal{N},\mathbf{A}}(\mathbf{x}), \mathbf{g}_F^{(N)}(\mathbf{x}))$, (2) the Resisting oracle has the properties required from a memory-sensitive class. To prove these statements, we define an event E in Definition 19, and then argue that the failure probability of E is at most $1/d$. Event E allows us to argue both that the Resisting oracle is identical to $\mathcal{O}_{F^{(N)}_{\mathcal{N},\mathbf{A}}}$, as well as to argue that $\mathcal{N}_{N,\gamma}$ is memory-sensitive class.

Definition 19 *Let E denote the event that $\forall j \in [N]$, the Nemirovski vector $\mathbf{v}_j$ has the following properties:*

1. *For any query $\mathbf{x}$ submitted by the Algorithm in Phase 1 through j, $|\mathbf{x}^\top \mathbf{v}_j| \le \sqrt{\frac{10\log(d)}{d}}$.*
2. $\left\|\mathrm{Proj}_{S_j}(\mathbf{v}_j)\right\|_2 \le \sqrt{\frac{30k\log(d)}{d}}$ *where S_j is defined as in Definition 10.*

Lemma 20 *Recall the definition of event E in Definition 19. Suppose that for every Phase $j \in [N]$, the Algorithm makes at most d^2 queries in that Phase. Then $\Pr[E] \ge 1 - (1/d)$.*

Proof We prove for any fixed $j \in [N]$, the failure probability of event E in Phase j is at most $1/d^2$ and then apply a union bound over all $j \in [N]$ (where $N \le d$). For the purpose of the proof, we leverage an oracle which knows the internal randomness of the Algorithm. Thus we think of the Algorithm's strategy as deterministic conditioned on its memory state and some random string R (as in Definition 4). Consider an oracle which knows the function $F^{(j-1)}_{\mathcal{N},\mathbf{A}}$, R and the state of the Algorithm when it receives $\mathbf{v}_{j-1}$ as the response from the Resisting oracle at the end of the $(j-1)^{\text{th}}$ Phase. Without knowledge of $\mathbf{v}_j$, the oracle can write down the sequence of the next d^2 vectors the Algorithm would query in the j^{th} Phase, where the next query is made if the previous query does not get the new Nemirovski vector $\mathbf{v}_j$ as the response (i.e., the previous query does not end the j^{th} Nemirovski Game). Let this set of vectors be T_j. Note that from the perspective of this oracle, the Algorithm first specifies this set T_j of the next d^2 queries it wants to submit, assuming the previous one does not obtain the new Nemirovski vector $\mathbf{v}_j$ as the gradient. Separately, the Resisting oracle samples $\mathbf{v}_j \overset{\text{i.i.d.}}{\sim} \left((1/\sqrt{d})\mathsf{Unif}\,(\mathcal{H}_d)\right)$, and therefore $\mathbf{v}_j$ is sampled independently of all these d^2 queries in the set T_j.

The proof now follows via simple concentration bounds. First consider property (1) of event E. By the argument above we see that for any any vector $\mathbf{x}$ which is in the set T_j or was submitted by the Algorithm in any of the previous $(j-1)$ Phases, by Corollary 37, $\mathbf{x}^\top \mathbf{v}_j$ is a zero-mean sub-Gassian with parameter at most $\|\mathbf{x}\|_2 / \sqrt{d} \le 1/\sqrt{d}$. Then by Fact 38, for any $t \ge 0$,

$$\mathbb{P}\left(\left|\mathbf{x}^\top \mathbf{v}_j\right| \ge t\right) \le 2\exp(-dt^2/2).$$

We remark that to invoke Corollary 37, we do not condition on whether or not vector $\mathbf{x}$ will ultimately be queried by the Algorithm (this is critical to maintain the independence of $\mathbf{x}$ and $\mathbf{v}_j$). Picking $t = \sqrt{10\log(d)/d}$ we have with failure probability at most $2/d^5$,

$$\left|\mathbf{x}^\top \mathbf{v}_j\right| \le \sqrt{\frac{10\log(d)}{d}}.$$

Since the Algorithm makes at most d^2 queries in any Phase, there are at most Nd^2 vectors in the union of the set T_j of possible queries which the Algorithm makes in the j^{th} Phase, and the set of queries submitted by the Algorithm in any of the previous Phases. Therefore by a union bound, property (1) of event E is satisfied by $\mathbf{v}_j$ with failure probability at most $2Nd^2/d^5 \le 2/d^2$, where we used that $N \le d$.

We now turn to property (2) of event E. Note that S_j depends on $\mathbf{x}_{t_j}$, which is the first query for which the Resisting oracle returns $\mathbf{v}_j$. However, using the same idea as above, we consider set T_j of the next d^2 queries which the Algorithm would make, and then do a union bound over every such query. For any query $\mathbf{x} \in T_j$, consider the subspace

$$S_{j-1,\mathbf{x}} \coloneqq \mathsf{span}\left(\{\mathbf{x}_{t_i} : j-k \le i < j, i > 0\}, \mathbf{x}\right).$$

Note that $S_j = S_{j-1,\mathbf{x}_{t_j}}$. Recalling the previous argument, $\mathbf{v}_j$ is independent of all vectors in the above set $\{\{\mathbf{x}_{t_i} : j-k \le i < j, i > 0\}, \mathbf{x}\}$. Therefore by Corollary 41 there is an absolute constant c such that with failure probability at most c/d^5,

$$\left\|\mathrm{Proj}_{S_{j-1,\mathbf{x}}}(\mathbf{v}_j)\right\|_2 \le \sqrt{\frac{30k\log(d)}{d}}.$$

Under the assumption that the Algorithm makes at most d^2 queries in the j^{th} Phase, the successful query $\mathbf{x}_{t_j}$ must be in the set T_j. Therefore, by a union bound over the d^2 queries which the Algorithm submits,

$$\left\|\mathrm{Proj}_{S_j}(\mathbf{v}_j)\right\|_2 \le \sqrt{\frac{30k\log(d)}{d}},$$

with failure probability at most $c/d^3 \le 1/d^2$ (for d large enough). Therefore, both properties (1) and (2) are satisfied for vector $\mathbf{v}_j$ with failure probability at most $1/d^2$ and thus by a union bound over all N Nemirovski vectors $\mathbf{v}_j$, E happens with failure probability at most $1/d$. ■

Now that we have justified that event E has small failure probability we will argue that conditioned on E, the Resisting oracle returns consistent first order information. This is critical to our analysis because we prove memory-sensitive properties through the lens of the Resisting oracle.

Lemma 21 *Under event E defined in Definition 19, the Resisting oracle's responses are consistent with itself, i.e., for any query made by the Algorithm, the Resisting oracle returns an identical first order oracle response to the final oracle $(F_{\mathcal{N},\mathbf{A}}^{(N)}(\mathbf{x}), \mathbf{g}_F^{(N)}(\mathbf{x}))$.*

Proof Assume event E and fix any $j \in [N-1]$. Let $\mathbf{x}$ be any query the Algorithm submits sometime before the end of the j^{th} Phase. Under event E for any $j' > j$,

$$|\mathbf{x}^\top \mathbf{v}_j| \leq \sqrt{\frac{10\log(d)}{d}}, \quad |\mathbf{x}^\top \mathbf{v}_{j'}| \leq \sqrt{\frac{10\log(d)}{d}}.$$

Therefore, since $\gamma = \sqrt{ck\log(d)/d}$, for any $c > 10$ and any $j' > j$,

$$\mathbf{x}^\top(\mathbf{v}_j - \mathbf{v}_{j'}) < \gamma.$$

In particular this implies that for $j' > j$

$$\mathbf{x}^\top \mathbf{v}_j - j\gamma > \mathbf{x}^\top \mathbf{v}_{j'} - j'\gamma.$$

Therefore for all $j' > j$, $\mathbf{v}_{j'}$ is not the gradient $\mathbf{g}_N(\mathbf{x})$. This implies that in the j^{th} Phase, the Resisting oracle always responds with the correct function value $F_{\mathcal{N},\mathbf{A}}^{(j)}(\mathbf{x}) = F_{\mathcal{N},\mathbf{A}}^{(N)}(\mathbf{x})$ and subgradient $\mathbf{g}_F^{(j)}(\mathbf{x}) = \mathbf{g}_F^{(N)}(\mathbf{x})$. ■

Now we address the memory-senstive properties of the function class by analyzing the Resisting oracle. Here, event E is crucial since it ensures that the Nemirovski vectors lie outside the span of previous queries which returned a unique gradient.

Remark 22 *Recall from Definition 9 that $\mathbf{x}_{t_j}$ is the query which returns the j^{th} informative subgradient from the function f. In terms of the Nemirovski function and the Resisting oracle, we note that $\mathbf{x}_{t_j}$ is the query which ends Phase j of the Resisting oracle (the first query such that the Resisting oracle returns $\mathbf{v}_j$ as the gradient in response).*

Lemma 23 *Recall the definition of $\mathbf{x}_{t_j}$ and S_j from Theorems 9 and 10 and that $\gamma = \sqrt{\frac{ck\log(d)}{d}}$. If $c \geq 400$, then under event E from Definition 19,*

$$\frac{\left\|\text{Proj}_{S_{j-1}}(\mathbf{x}_{t_j})\right\|_2}{\left\|\mathbf{x}_{t_j}\right\|_2} \leq 1 - (1/d^2).$$

Proof We prove Lemma 23 by providing both an upper and lower bound on $\mathbf{x}_{t_j}^\top(\mathbf{v}_j - \mathbf{v}_{j-1})$ under event E. Towards this end, observe that for any $\mathbf{x} \in S_{j-1}$,

$$\mathbf{x}^\top \mathbf{v}_{j-1} \leq \|\mathbf{x}\|_2 \left\|\text{Proj}_{S_{j-1}}(\mathbf{v}_{j-1})\right\|_2.$$

Therefore, recalling that every query $\mathbf{x}$ satisfies $\|\mathbf{x}\|_2 \leq 1$, we see that under E,

$$\text{Proj}_{S_{j-1}}(\mathbf{x}_{t_j})^\top \mathbf{v}_{j-1} \leq \sqrt{\frac{30k\log(d)}{d}}.$$

Next, writing $\mathbf{x}_{t_j}$ as $\mathbf{x}_{t_j} = \mathrm{Proj}_{S_{j-1}}(\mathbf{x}_{t_j}) + \mathrm{Proj}_{S_{j-1}^{\perp}}(\mathbf{x}_{t_j})$ and recalling that $\left\|\mathbf{x}_{t_j}\right\|_2 \le 1$ we observe that

$$\left\|\mathrm{Proj}_{S_{j-1}^{\perp}}(\mathbf{x}_{t_j})\right\|_2^2 = \left\|\mathbf{x}_{t_j}\right\|_2^2 - \left\|\mathrm{Proj}_{S_{j-1}}(\mathbf{x}_{t_j})\right\|_2^2 \le 1 - \left\|\mathrm{Proj}_{S_{j-1}}(\mathbf{x}_{t_j})\right\|_2^2 / \left\|\mathbf{x}_{t_j}\right\|_2^2.$$

Recalling that $\|\mathbf{v}_j\|_2 = 1$,

$$\mathrm{Proj}_{S_{j-1}^{\perp}}(\mathbf{x}_{t_j})^{\top}\mathbf{v}_{j-1} \le \left\|\mathrm{Proj}_{S_{j-1}^{\perp}}(\mathbf{x}_{t_j})\right\|_2 \|\mathbf{v}_{j-1}\|_2 \le \left(1 - \left\|\mathrm{Proj}_{S_{j-1}}(\mathbf{x}_{t_j})\right\|_2^2 / \left\|\mathbf{x}_{t_j}\right\|_2^2\right)^{1/2}.$$

We also note that under event E,

$$|\mathbf{x}_{t_j}^{\top}\mathbf{v}_j| \le \sqrt{\frac{10\log(d)}{d}}.$$

Therefore,

$$\begin{aligned}
\mathbf{x}_{t_j}^{\top}(\mathbf{v}_j - \mathbf{v}_{j-1}) &\le \left|\mathbf{x}_{t_j}^{\top}\mathbf{v}_j\right| + \left|\mathrm{Proj}_{S_{j-1}}(\mathbf{x}_{t_j})^{\top}\mathbf{v}_{j-1}\right| + \left|\mathrm{Proj}_{S_{j-1}^{\perp}}(\mathbf{x}_{t_j})^{\top}\mathbf{v}_{j-1}\right| \\
&\le \sqrt{\frac{10\log(d)}{d}} + \sqrt{\frac{30k\log(d)}{d}} + \sqrt{1 - \left\|\mathrm{Proj}_{S_{j-1}}(\mathbf{x}_{t_j})\right\|_2^2 / \left\|\mathbf{x}_{t_j}\right\|_2^2} \\
&\le \sqrt{\frac{80k\log(d)}{d}} + \sqrt{1 - \left\|\mathrm{Proj}_{S_{j-1}}(\mathbf{x}_{t_j})\right\|_2^2 / \left\|\mathbf{x}_{t_j}\right\|_2^2}.
\end{aligned}$$

Note that $\mathbf{v}_j$ is an informative gradient and therefore $\mathbf{v}_j \in \partial\mathcal{N}(\mathbf{x}_{t_j})$. Thus

$$\mathbf{x}_{t_j}^{\top}\mathbf{v}_{j-1} - (j-1)\gamma \le \mathbf{x}_{t_j}^{\top}\mathbf{v}_j - j\gamma, \qquad \text{or equivalently} \qquad \mathbf{x}_{t_j}^{\top}(\mathbf{v}_j - \mathbf{v}_{j-1}) \ge \gamma.$$

Therefore,

$$\sqrt{\frac{80k\log d}{d}} + \sqrt{1 - \left\|\mathrm{Proj}_{S_{j-1}}(\mathbf{x}_{t_j})\right\|_2^2 / \left\|\mathbf{x}_{t_j}\right\|_2^2} \ge \gamma.$$

Rearranging terms and using that $(1-x)^{1/2} \le 1 - (1/4)x$ for $x \ge 0$ gives us

$$\left\|\mathrm{Proj}_{S_{j-1}}(\mathbf{x}_{t_j})\right\|_2^2 / \left\|\mathbf{x}_{t_j}\right\|_2 \le \sqrt{1 - \left(\gamma - \sqrt{\frac{80k\log d}{d}}\right)^2} \le 1 - \frac{1}{4}\left(\gamma - \sqrt{\frac{80k\log d}{d}}\right)^2.$$

Plugging in $\gamma = \sqrt{\frac{ck\log(d)}{d}}$, for $c \ge 400$ gives us the claimed result for d large enough,

$$\left\|\mathrm{Proj}_{S_{j-1}}(\mathbf{x}_{t_j})\right\|_2^2 / \left\|\mathbf{x}_{t_j}\right\|_2 \le 1 - \frac{1}{4}\frac{k\log d}{d}\left(\sqrt{400} - \sqrt{80}\right)^2 \ge 1 - \frac{1}{d^2}.$$

■

We next turn our attention to the optimality achievable by an algorithm which has seen a limited number of Nemirovski vectors. Lemma 24 lower bounds the best function value achievable by such an algorithm.

Lemma 24 *Recall that $\gamma = \sqrt{\frac{ck\log(d)}{d}}$ and suppose $c > 10$. Assume the Algorithm has proceeded to Phase r of the Resisting oracle and chooses to not make any more queries. Let Q be the set of all queries made so far by the Algorithm. Then under event E defined in Definition 19,*

$$\min_{\mathbf{x}\in Q} F_{\mathcal{N},\mathbf{A}}^{(N)}(\mathbf{x}) \geq -(r+1)\gamma.$$

Proof We first note that

$$\min_{\mathbf{x}\in Q} F_{\mathcal{N},\mathbf{A}}^{(N)}(\mathbf{x}) \geq \min_{\mathbf{x}\in Q} F_{\mathcal{N},\mathbf{A}}^{(r)}(\mathbf{x}).$$

Now under event E, all queries $\mathbf{x} \in Q$ made by the Algorithm satisfy,

$$|\mathbf{x}^\top \mathbf{v}_r| \leq \sqrt{\frac{10\log(d)}{d}}.$$

Assuming event E, by Lemma 21 if the Resisting oracle responds with some Nemirovski vector as the subgradient, since this Nemirovski vector must be some $\mathbf{v}_i$ for $i < r$ and since the returned subgradients are valid subgradients of the final function $F_{\mathcal{N},\mathbf{A}}^{(N)}$, the Nemirovski vector $\mathbf{v}_r$ could not have been a valid subgradient of the query. Therefore, all queries $\mathbf{x} \in Q$ must satisfy,

$$F_{\mathcal{N},\mathbf{A}}^{(r)}(\mathbf{x}) \geq \mathbf{x}^\top \mathbf{v}_r - r\gamma \geq -\sqrt{\frac{10\log(d)}{d}} - r\gamma \geq -(r+1)\gamma$$

where in the last inequality we use the fact that $\gamma \geq \sqrt{\frac{10\log(d)}{d}}$. ∎

Lemma 25 upper bounds the minimum value of the final function $F_{\mathcal{N},\mathbf{A}}^{(N)}(\mathbf{x})$, and will be used to establish an optimality gap later.

Lemma 25 *Recall that $\gamma = \sqrt{\frac{ck\log(d)}{d}}$ and suppose $c > 2$. For any $\mathbf{A} \in \mathbb{R}^{n\times d}$ where $n \leq d/2$ and sufficiently large d, with failure probability at most $2/d$ over the randomness of the Nemirovski vectors $\{\mathbf{v}_j, j \in [N]\}$,*

$$\min_{\|\mathbf{x}\|_2\leq 1} F_{\mathcal{N},\mathbf{A}}^{(N)}(\mathbf{x}) \leq -\frac{1}{8\sqrt{N}}. \tag{A.1}$$

Proof Let the rank of $\mathbf{A}$ be r. Let $\mathbf{Z} \in \mathbb{R}^{d\times(d-r)}$ be an orthonormal matrix whose columns are an orthonormal basis for the null space $A^\perp$ of $\mathbf{A}$. We construct a vector $\overline{\mathbf{x}}$ which (as we will show) attains $F_{\mathcal{N},\mathbf{A}}(\overline{\mathbf{x}}) \leq -1/8\sqrt{N}$ as follows,

$$\overline{\mathbf{x}} = \frac{-1}{2\sqrt{N}} \sum_{i\in[N]} \mathrm{Proj}_{A^\perp}(\mathbf{v}_i) = \frac{-1}{2\sqrt{N}} \sum_{i\in[N]} \mathbf{Z}\mathbf{Z}^\top \mathbf{v}_i = \mathbf{Z}\mathbf{Z}^\top \left(\frac{-1}{2\sqrt{N}} \sum_{i\in[N]} \mathbf{v}_i\right).$$

Our proof proceeds by providing an upper bound for $F_{\mathcal{N},\mathbf{A}}(\overline{\mathbf{x}})$. First we bound $\|\overline{\mathbf{x}}\|_2^2$. Let $\bar{\mathbf{v}} = \left(\frac{-1}{2\sqrt{N}} \sum_{i\in[N]} \mathbf{v}_i\right)$. Note that

$$\|\overline{\mathbf{x}}\|_2^2 = \left\|\mathbf{Z}^\top \bar{\mathbf{v}}\right\|_2^2.$$

By Fact 36, each coordinate of $\bar{\mathbf{v}}$ is sub-Gaussian with sub-Gaussian norm $\alpha/\sqrt{d}$ (where α is the absolute constant in Fact 36). Also $\mathbb{E}[\bar{\mathbf{v}}\bar{\mathbf{v}}^\top] = (1/4d)\mathbf{I}_d$. Therefore by Lemma 40, with failure probability at most $2\exp(-c_1 d)$ for some absolute constant $c_1 > 0$,

$$\|\overline{\mathbf{x}}\|_2^2 \leq \frac{d-r}{4d} + \frac{1}{2} \leq 1.$$

Therefore $\overline{\mathbf{x}}$ lies in the unit ball with failure probability at most $2\exp(-c_1 d)$. Next fix any $\mathbf{v}_j \in \{\mathbf{v}_i\}_{i\in[N]}$ and consider

$$\begin{aligned}
\mathbf{v}_j^\top \overline{\mathbf{x}} &= \left(\mathbf{v}_j^\top \left(\mathbf{Z}\mathbf{Z}^\top \left(\frac{-1}{2\sqrt{N}} \sum_{i\in[N], i\neq j} \mathbf{v}_i\right)\right)\right) - \frac{1}{2\sqrt{N}} \mathbf{v}_j^\top \mathbf{Z}\mathbf{Z}^\top \mathbf{v}_j \\
&= \left((\mathbf{Z}\mathbf{Z}^\top \mathbf{v}_j)^\top \left(\frac{-1}{2\sqrt{N}} \sum_{i\in[N], i\neq j} \mathbf{v}_i\right)\right) - \frac{1}{2\sqrt{N}} \left\|\mathbf{Z}^\top \mathbf{v}_j\right\|_2^2. \qquad \text{(A.2)}
\end{aligned}$$

Let $\bar{\mathbf{v}}_{-j} = \left(\frac{-1}{2\sqrt{N}} \sum_{i\in[N], i\neq j} \mathbf{v}_i\right)$. By Fact 36, each coordinate of $\bar{\mathbf{v}}_{-j}$ is sub-Gaussian with sub-Gaussian parameter $1/\sqrt{4d}$. Also by Fact 36, $(\mathbf{Z}\mathbf{Z}^\top \mathbf{v}_j)^\top \bar{\mathbf{v}}_{-j}$ is sub-Gaussian with sub-Gaussian parameter $\left\|\mathbf{Z}\mathbf{Z}^\top \mathbf{v}_j\right\|_2 / \sqrt{4d} = \left\|\mathbf{Z}^\top \mathbf{v}_j\right\|_2 / \sqrt{4d}$. Now using Fact 38, with failure probability at most $2/d^C$,

$$\left|(\mathbf{Z}\mathbf{Z}^\top \mathbf{v}_j)^\top \bar{\mathbf{v}}_{-j}\right| \leq \frac{\sqrt{2C\log(d)} \left\|\mathbf{Z}^\top \mathbf{v}_j\right\|_2}{\sqrt{4d}}. \qquad \text{(A.3)}$$

We now turn to bounding $\left\|\mathbf{Z}^\top \mathbf{v}_j\right\|_2$. Note that each coordinate of $\mathbf{v}_j$ is sub-Gaussian with sub-Gaussian norm $2/\sqrt{d}$ and $\mathbb{E}[\mathbf{v}_j \mathbf{v}_j^\top] = (1/d)\mathbf{I}_d$. Therefore, by Lemma 40, with failure probability at most $2\exp(-c_2 d)$ for some absolute constant $c_2 > 0$,

$$\frac{1}{4} \leq \frac{d-r}{d} - 1/4 \leq \left\|\mathbf{Z}^\top \mathbf{v}_j\right\|_2^2 \leq \frac{d-r}{d} + 1/4 \leq 2 \qquad \text{(A.4)}$$

were we have used the fact that $r \leq d/2$. Then if Eq. (A.3) and Eq. (A.4) hold,

$$\left|(\mathbf{Z}\mathbf{Z}^\top \mathbf{v}_j)^\top \bar{\mathbf{v}}_{-j}\right| \leq \sqrt{\frac{C\log(d)}{d}}. \qquad \text{(A.5)}$$

Moreover, under Eq. (A.4),

$$\frac{1}{2\sqrt{N}} \left\|\mathbf{Z}^\top \mathbf{v}_j\right\|_2^2 \geq \frac{1}{8\sqrt{N}}. \qquad \text{(A.6)}$$

Combining Eq. (A.2), Eq. (A.5), and Eq. (A.6) we find that with failure probability at most $2/d^C + 2\exp(-c_2 d)$,

$$\mathbf{v}_j^\top \overline{\mathbf{x}} \leq -\frac{1}{8\sqrt{N}} + \sqrt{\frac{C\log(d)}{d}}.$$

Then, using a union bound we conclude that with failure probability at most $N\left(2/d^C + 2\exp(-c_2 d)\right)$,

$$\max_{j\in[N]} \mathbf{v}_j^\top \overline{\mathbf{x}} \leq -\frac{1}{8\sqrt{N}} + \sqrt{\frac{C\log(d)}{d}}.$$

Finally since $\eta \|\mathbf{A}\mathbf{x}\|_\infty - \rho = -\rho = -1 \leq -1/(8\sqrt{N})$, we conclude

$$F^{(N)}_{\mathcal{N},\mathbf{A}}(\overline{\mathbf{x}}) \leq -\frac{1}{8\sqrt{N}} + \sqrt{\frac{C\log(d)}{d}} - \gamma.$$

We now take $C = 2$, which gives us a overall failure probability of $2\exp(-c_1 d) + N\left(2/d^C + 2\exp(-c_2 d)\right) \leq 2/d$ (for sufficiently large d and since $N \leq d$). Noting that $\gamma > \sqrt{2\log(d)/d}$ finishes the proof. ■

The following simple lemma shows that all the Nemirovski vectors are unique with high probability, and will be useful for our definition of informative subgradients.

Lemma 26 *For sufficiently large d, with failure probability at most $1/d$, $\mathbf{v}_i \neq \mathbf{v}_j\ \forall\ i, j \in [N], i \neq j$.*

Proof The proof follows from the birthday paradox since $\mathbf{v}_i$ are drawn from the uniform distribution over a support of size 2^d: the probability of $\mathbf{v}_i \neq \mathbf{v}_j\ \forall\ i, j \in [N], i \neq j$ is equal to,

$$1 \cdot \frac{2^d - 1}{2^d} \cdot \frac{2^d - 2}{2^d} \cdots \frac{2^d - (N-1)}{2^d} \geq \left(\frac{2^d - d}{2^d}\right)^d \geq 1/d$$

for sufficiently large d. ■

Finally, we note that for the orthogonality condition from the definition of a memory-sensitive class is satisfied with our definition of the subgradient oracle.

Lemma 27 *For any $j \in [N]$ the following is true: For any $\mathbf{x} \in \mathbb{R}^d$ such that $F^{(j)}_{\mathcal{N},\mathbf{A}}(\mathbf{x}) \neq \eta \|\mathbf{A}\mathbf{x}\|_\infty - \rho$, either $\mathbf{g}^{(j)}_F(\mathbf{x}) = \mathbf{v}_1$ or $\|\mathbf{A}\mathbf{x}\|_\infty / \|\mathbf{x}\|_2 \leq 1/d^4$.*

Proof Fix any $j \in [N]$ and assume $\mathbf{x}$ is such that $F^{(j)}_{\mathcal{N},\mathbf{A}}(\mathbf{x}) \neq \eta \|\mathbf{A}\mathbf{x}\|_\infty - \rho$. First, if $\mathbf{g}^{(j)}_F(\mathbf{x}) \neq \mathbf{v}_1$, then $\mathbf{g}^{(j)}_F(\mathbf{x})$ is a Nemirovski vector $\mathbf{v}_k$ for some $k \in \{2, \ldots, j\}$. This implies that $\eta \|\mathbf{A}\mathbf{x}\|_\infty - \rho \leq \mathbf{v}_k^\top \mathbf{x} - k\gamma$. Observe that $\mathbf{v}_k^\top \mathbf{x} - j\gamma \leq \|\mathbf{v}_k\|_2 \|\mathbf{x}\|_2$ and that for any $j \in [N]$, $\|\mathbf{v}_k\|_2 = 1$. Therefore

$$\eta \|\mathbf{A}\mathbf{x}\|_\infty - \rho \leq \|\mathbf{x}\|_2.$$

Next we bound $\|\mathbf{x}\|_2$ from below. For $k \in \{2, \ldots, j\}$, $\mathbf{g}^{(j)}_F(\mathbf{x}) = \mathbf{v}_k$ implies that

$$\mathbf{v}_k^\top \mathbf{x} - k\gamma \geq \mathbf{v}_{k-1}^\top \mathbf{x} - (k-1)\gamma \implies (\mathbf{v}_k - \mathbf{v}_{k-1})^\top \mathbf{x} \geq \gamma.$$

Therefore,

$$\|\mathbf{x}\|_2 \geq \frac{(\mathbf{v}_k - \mathbf{v}_{k-1})^\top}{\|\mathbf{v}_k - \mathbf{v}_{k-1}\|_2}\mathbf{x} \geq \frac{\gamma}{\|\mathbf{v}_k - \mathbf{v}_{k-1}\|_2} \geq \frac{\gamma}{2}.$$

Since $\|\mathbf{x}\|_2 > 0$, we can write

$$\eta \|\mathbf{A}\mathbf{x}\|_\infty - \rho \leq \|\mathbf{x}\|_2 \implies \frac{\|\mathbf{A}\mathbf{x}\|_\infty}{\|\mathbf{x}\|_2} \leq \frac{1}{\eta} + \frac{\rho}{\eta \|\mathbf{x}\|_2} \leq \frac{1}{\eta} + \frac{2\rho}{\eta\gamma}.$$

Then recalling that $\eta = d^5$, $\rho = 1$ and $\gamma = \sqrt{ck\log(d)/d}$,

$$\frac{\|\mathbf{A}\mathbf{x}\|_\infty}{\|\mathbf{x}\|_2} \leq d^{-4}.$$

Otherwise if $\|\mathbf{A}\mathbf{x}\|_\infty / \|\mathbf{x}\|_2 > 1/d^4$ then $\eta \|\mathbf{A}\mathbf{x}\|_\infty - \rho > d\|\mathbf{x}\|_2 - 1$. Since $F^{(j)}_{\mathcal{N},\mathbf{A}}(\mathbf{x}) \neq \eta\|\mathbf{A}\mathbf{x}\|_\infty - \rho$ then for some Nemirovski vector $\mathbf{v}_k$ for $k \in [j]$, $\mathbf{g}^{(j)}_F(\mathbf{x}) = \mathbf{v}_k$. Therefore we have

$$d\|\mathbf{x}\|_2 - 1 < \eta\|\mathbf{A}\mathbf{x}\|_\infty - \rho < \mathbf{v}_k^\top \mathbf{x} - k\gamma < \|\mathbf{x}\|_2 - k\gamma,$$

and so $\|\mathbf{x}\|_2 < 1/(d-1)$. Thus

$$\mathbf{v}_1^\top \mathbf{x} - \gamma \geq -\|\mathbf{x}\|_2 - \gamma \geq -1/(d-1) - \gamma,$$

while for any $k \geq 2$,

$$\mathbf{v}_k^\top \mathbf{x} - k\gamma \leq \|\mathbf{x}\|_2 - k\gamma \leq 1/(d-1) - 2\gamma.$$

Since $\gamma > 1/(d-1)$ we conclude that for any $k \geq 2$, $\mathbf{v}_k^\top \mathbf{x} - k\gamma < \mathbf{v}_1^\top \mathbf{x} - \gamma$ and therefore $\mathbf{g}^{(j)}_F(\mathbf{x}) = \mathbf{v}_1$.

■

Putting together all the previous results, we show Theorem 28 which establishes that the Nemirovski function class has the required memory-sensitive properties.

Theorem 28 *Consider any $\mathbf{A} \in \mathbb{R}^{n\times d}$ where $n \leq d/2$ and any row has ℓ_2-norm bounded by d. Fix $\eta = d^5$, $\rho = 1$, $\gamma = \sqrt{\frac{ck\log(d)}{d}}$ for $c = 400$, and $k \leq \lceil 20M/(c_{\mathcal{H}}d)\rceil$. Let $N \leq (1/32\gamma)^{2/3}$ and note that by our definition of γ and bound on k we may equivalently assume $N \leq \frac{1}{205}\left(\frac{c_{\mathcal{H}}d^2}{M\log(d)}\right)^{1/3}$. Then $\mathcal{N}_{N,\gamma}$ is a $(d^6, N, k, 1/16\sqrt{N})$-memory-sensitive class. That is, we can sample $(f, \mathbf{g}_f)$ from $\mathcal{N}_{N,\gamma}$ with at most d^2 random bits, and for $(f, \mathbf{g}_f) \sim \mathcal{N}_{N,\gamma}$, with failure probability at most $5/d$, any algorithm for optimizing $F_{f,\mathbf{A}}(\mathbf{x})$ which makes fewer than d^2 queries to oracle $\mathbf{g}_f$ has the following properties:*

1. *$F_{f,\mathbf{A}}$ is convex and d^6-Lipschitz.*
2. *With S_j and $\mathbf{x}_{t_j}$ as defined in Definition 10,*

$$\left\|\mathrm{Proj}_{S_{j-1}}(\mathbf{x}_{t_j})\right\|_2 / \left\|\mathbf{x}_{t_j}\right\|_2 \leq 1 - 1/d^2.$$

3. *Any query $\mathbf{x}$ such that $F_{f,\mathbf{A},\eta,\rho}(\mathbf{x}) \neq \eta\|\mathbf{A}\mathbf{x}\|_\infty - \rho$ satisfies*

$$\mathbf{g}_{F_{f,\mathbf{A},}}(\mathbf{x}) = \mathbf{v}_1 \text{ or } \|\mathbf{A}\mathbf{x}\|_\infty / \|\mathbf{x}\|_2 \leq d^{-4}.$$

4. *Any algorithm which has queried $r < N$ unique gradients from f has a sub-optimality gap of at least*

$$F_{f,\mathbf{A}}(\mathbf{x}_T) - F_{f,\mathbf{A}}(\mathbf{x}^\star) \geq \frac{1}{16\sqrt{N}}.$$

Proof Note that by Lemma 18, the first order oracle $\mathcal{O}_{F^{(N)}_{\mathcal{N},\mathbf{A}}}(\mathbf{x}) = (F^{(N)}_{\mathcal{N},\mathbf{A}}(\mathbf{x}), \mathbf{g}^{(N)}_F(\mathbf{x}))$ has the same distribution over responses as $\mathcal{N}_{N,\gamma,\mathbf{A}}$. Therefore we will analyze $\mathcal{O}_{F^{(N)}_{\mathcal{N},\mathbf{A}}}$, using the Resisting oracle.

We consider the following sequence of events: (a) event E defined in Lemma 20 holds, (b) no two Nemirovski vectors are identical, $\mathbf{v}_i \neq \mathbf{v}_j$ for $i \neq j$, (c) Eq. (A.1) regarding the value for

$F^{(N)}_{f,\mathbf{A}}(\mathbf{x}^\star)$ holds. We claim that for any algorithm which makes fewer than d^2 queries, events (a) through (c) happen with failure probability at most $5/d$. To verify this, we do a union bound over the failure probability of each event: for (a), Lemma 20 shows that E holds with failure probability at most $1/d$ for any algorithm which makes fewer than d^2 queries, (b) holds with probability $1/d$ from Lemma 26 and (c) holds with probability $2/d$ from Lemma 24. Now note that by Lemma 21, the Resisting oracle's responses are identical to the responses of oracle $\mathcal{O}_{F^{(N)}_{\mathcal{N},\mathbf{A}}}$ under E, and therefore when conditioned on events (a) through (c). Therefore, we will condition on events (a) through (c) all holding and consider the Resisting oracle to prove all the four parts of the theorem.

We begin with the first part. Note that $f(\mathbf{x})$ has Lipschitz constant bounded by 1. Next note that since each $\mathbf{a} \in B_d$ has $\|\mathbf{a}\|_2 \le d$ and $\eta = d^5$ we have that the $\{\eta \|\mathbf{A}\mathbf{x}\|_\infty - \rho\}$ term has Lipschitz constant bounded by $L \le d^6$. Therefore, $F_{f,\mathbf{A}}$ has Lipschitz constant bounded by d^6. For the second part we first note that under event (b) all Nemirovski vectors are unique, therefore the vectors $\mathbf{x}_{t_j}$ and subspace S_{j-1} are well-defined. Now under event E by applying Lemma 23,

$$\left\|\mathrm{Proj}_{S_{j-1}}(\mathbf{x}_{t_j})\right\|_2 / \left\|\mathbf{x}_{t_j}\right\| \le 1 - 1/d^2.$$

The third part holds by Lemma 27. For the final part, we first note that under event (b) if the Algorithm observes r unique Nemirvoski vectors then it has only proceeded to the $(r+1)^{\text{th}}$ Phase of the Resisting oracle. Now by Lemma 24, under event E, any algorithm in the $(r+1)^{\text{th}}$ Phase of the Resisting oracle has function value at least $-(r+2)\gamma$. Combining this with Eq. (A.1) holding, for any algorithm which observes at most than r unique Nemirovski vectors as gradients,

$$F^{(N)}_{f,\mathbf{A}}(\mathbf{x}_T) - F^{(N)}_{f,\mathbf{A}}(\mathbf{x}^\star) \ge -(r+2)\gamma + 1/(8\sqrt{N}) \ge -(N+2)\gamma + 1/(8\sqrt{N}) \ge 1/(16\sqrt{N}),$$

where the final inequality holds by noting that $N \le (1/32\gamma)^{2/3}$ and so $-(N+2)\gamma \ge -1/(16\sqrt{N})$.
■

Appendix B. From Optimization to Winning the Orthogonal Vector Game

In this section we prove Lemma 14.

Proof [Proof of Lemma 14] We will show correctness for the strategy provided in Algorithm 2 for the Player to use $\mathcal{A}_{\text{rand}}$ (the M-bit algorithm that minimizes $F_{f,\mathbf{A}}$) to win the Orthogonal Vector Game with probability at least $1/3$ over the randomness in R and $\mathbf{A}$. The Player uses the random string $R = (R_1, R_2, R_3)$ to sample a function/oracle pair $(f, \mathbf{g}_f) \sim \mathcal{F}$, and to run the algorithm $\mathcal{A}_{\text{rand}}$ (if it is randomized). Note by Definition 10, $(f, \mathbf{g}_f)$ can be sampled from $\mathcal{F}$ with at most 2^d random bits, therefore the Player can sample $(f, \mathbf{g}_f) \sim \mathcal{F}$ using R_1. The Player uses R_2 and R_3 to supply any random bits for the execution of $\mathcal{A}_{\text{rand}}$, which by Definition 4 of $\mathcal{A}_{\text{rand}}$, is a sufficient number of random bits.

Let G be the event that R_1, R_2, R_3, and $\mathbf{A}$ have the property that (1) $\mathcal{A}_{\text{rand}}$ succeeds in finding an $\epsilon^\star$-optimal point for $F_{f,\mathbf{A}}(\mathbf{x})$, and (2) all the properties of a memory-sensitive class in Definition 10 are satisfied. By the guarantee in Lemma 14, $\mathcal{A}_{\text{rand}}$ finds an $\epsilon^\star$-optimal point with failure probability at most $1/3$ over the randomness in $\mathbf{A}$, R_1 (the randomness in $(f, \mathbf{g}_f)$) and R_2, R_3 (the internal randomness used by $\mathcal{A}_{\text{rand}}$). Also, the properties in Definition 10 are satisfied with failure probability at most $1/3$ by definition. Therefore, by a union bound, $\Pr[G] \geq 1/3$. We will condition on G for the rest of the proof.

We now prove that Part 1 does not fail under event G. Recall the definition of informative subgradients from Definition 9, and note that by Eq. (2.4) in Definition 10, if $\mathcal{A}_{\text{rand}}$ finds an $\epsilon^\star$-optimal point, then $\mathcal{A}_{\text{rand}}$ must observe at least N informative subgradients. Therefore, since $\mathcal{A}_{\text{rand}}$ can find an $\epsilon^\star$-optimal point using $m\lfloor N/(k+1)\rfloor$ queries it can observe N informative subgradients from f using $m\lfloor N/(k+1)\rfloor$ queries. For any execution of the algorithm, let $\mathbf{v}_i$ denote the i^{th} informative subgradient from f. Block the $\mathbf{v}_i$'s into $\lfloor N/(k+1)\rfloor$ groups of size $(k+1)$: $\mathsf{Block}_i = \left\{\mathbf{v}_{i(k+1)+1}, \ldots, \mathbf{v}_{(i+1)(k+1)}\right\}$. If $\mathcal{A}_{\text{rand}}$ observes N informative subgradients from f using $m\lfloor N/(k+1)\rfloor$ queries, then there is an index $i^\star$ such that $\mathcal{A}_{\text{rand}}$ observes $\mathsf{Block}_{i^\star}$ using at most m queries. Therefore, Part 1 does not fail under event G.

In Part 2 of the strategy, the Player no longer has access to $\mathbf{A}$, but by receiving the Oracle's responses $\mathbf{g}_i$ she can still implement the first-order oracle $\mathcal{O}_{F_{f,\mathbf{A}}}$ (as defined in Definition 8) and hence run $\mathcal{A}_{\text{rand}}$. Consequently, to complete the proof we will now show that under event G the Player can find a set of successful vectors in Part 3 and win. By the guarantee of Part 1, the Player has made at least $k+1$ informative queries among the m queries she made in Part 2. By Definition 10, if $\mathbf{x}_i$ is an informative query, then the query satisfies Eq. (2.3) and Eq. (2.2). Using this, if the Player has observed $k+1$ informative subgradients then she's made at least k queries which are successful (where the extra additive factor of one comes from the fixed vector $\mathbf{v}_1$ in Eq. (2.3), and note that the subspace in Eq. (2.2) is defined as the span of all previous k informative queries, and this will contain the subspace defined in the robust linear independence condition). Therefore she will find a set of k successful vectors among the m queries made in Part 2 under event G. Since G happens with probability at least $1/3$, she can win with probability at least $1/3$. ∎

Appendix C. Memory Lower Bounds for the Orthogonal Vector Game

In this section, we prove the lower bound for the Orthogonal Vector Game (Game 1), building on the proof sketch in Section 3.2. We prove a slightly stronger result in that we allow the oracle response $\mathbf{g}_i \in \mathbb{R}^d$ to any query $\mathbf{x}_i \in \mathbb{R}^d$ be an arbitrary (possibly randomized) function from $\mathbb{R}^d \to \{\mathbf{a}_1, \ldots, \mathbf{a}_n\}$.

Recall that $\mathbf{G} = (\mathbf{g}_1, \ldots, \mathbf{g}_m)^\top \in \mathbb{R}^{m\times d}$. We perform a small augmentation of the matrix $\mathbf{G}$ to streamline our analysis. Let $m_{\text{U}} \in [m]$ denote the number of *unique rows* of $\mathbf{A}$ contained in $\mathbf{G}$. We consider a row of $\mathbf{G}$ to be unique if it corresponds to a unique row index of $\mathbf{A}$, however it is still possible that two unique rows as defined are actually the same (in case $\mathbf{A}$ has repeated rows). If $m_{\text{U}} < m$, construct a matrix $\tilde{\mathbf{G}} \in \mathbb{R}^{(2m-m_{\text{U}})\times d}$ by appending $m - m_{\text{U}}$ additional rows of $\mathbf{A}$ to the matrix $\mathbf{G}$, choosing the first $m - m_{\text{U}}$ rows of $\mathbf{A}$ such that $\tilde{\mathbf{G}}$ now has exactly m unique rows. Given $\mathbf{A}$ and $\tilde{\mathbf{G}}$ let $\mathbf{A}' \in \mathbb{R}^{(n-m)\times d}$ denote the matrix $\mathbf{A}$ when the unique rows in $\tilde{\mathbf{G}}$ are removed. Drawing on these definitions, we begin with the following relation between the entropy of $\mathbf{A}$ and $\mathbf{A}'$ conditioned on $\tilde{\mathbf{G}}$ and R.

Lemma 29 $H(\mathbf{A}|\tilde{\mathbf{G}}, R) \leq H(\mathbf{A}'|\tilde{\mathbf{G}}, R) + 2m\log(4n)$.

Proof Note that for any random variable X, there exists a description of the random variable with expected description length L_X bounded as (Cover and Thomas, 1991, Chapter 5),

$$H(X) \leq L_X \leq H(X) + 1. \tag{C.1}$$

Now fix any $\tilde{\mathbf{G}} = \tilde{\mathbf{G}}'$ and $R = R'$. Let $h_{\tilde{\mathbf{G}}',R'} := H(\mathbf{A}'|\tilde{\mathbf{G}} = \tilde{\mathbf{G}}', R = R')$. Note that by definition,

$$H(\mathbf{A}'|\tilde{\mathbf{G}}, R) = \sum_{\tilde{\mathbf{G}}'} \sum_{R'} h_{\tilde{\mathbf{G}}',R'} \, \mathbb{P}(\tilde{\mathbf{G}} = \tilde{\mathbf{G}}') \, \mathbb{P}(R = R').$$

Now by Eq. (C.1) if $H(\mathbf{A}'|\tilde{\mathbf{G}} = \tilde{\mathbf{G}}', R = R') = h_{\tilde{\mathbf{G}}',R'}$, then given $\tilde{\mathbf{G}} = \tilde{\mathbf{G}}'$ and $R = R'$ there exists some description of $\mathbf{A}'$ which has expected description length at most $h_{\tilde{\mathbf{G}}',R'} + 1$. Using this we construct a description of $\mathbf{A}$ given $\tilde{\mathbf{G}} = \tilde{\mathbf{G}}'$ and $R = R'$ which has expected description length at most $h_{\tilde{\mathbf{G}}',R'} + 1 + 2m\log(n)$. To do this, note that if there is already a description of $\mathbf{A}'$ and we are given $\tilde{\mathbf{G}} = \tilde{\mathbf{G}}'$, then to specify $\mathbf{A}$ we only need to additionally specify the m rows which were removed from $\mathbf{A}$ to construct $\mathbf{A}'$. For every row which was removed from $\mathbf{A}$, we can specify its row index in the original matrix $\mathbf{A}$ using $\lceil \log(n) \rceil \le \log(2n)$ bits, and specify which of the rows of $\tilde{\mathbf{G}}'$ it is using another $\lceil \log(2m - m_{\mathrm{U}}) \rceil \le \lceil \log(2m) \rceil \le \log(4n)$ bits. Therefore, given $\tilde{\mathbf{G}} = \tilde{\mathbf{G}}'$ and $R = R'$,

$$\begin{aligned} H(\mathbf{A}|\tilde{\mathbf{G}} = \tilde{\mathbf{G}}', R = R') &\le h_{\tilde{\mathbf{G}}',R'} + 1 + m\log(2n) + m\log(4n) \le h_{\tilde{\mathbf{G}}',R'} + 2m\log(4n) \\ \implies H(\mathbf{A}|\tilde{\mathbf{G}}, R) &= \sum_{\tilde{\mathbf{G}}'} \sum_{R'} H(\mathbf{A}|\tilde{\mathbf{G}} = \tilde{\mathbf{G}}', R = R') \, \mathbb{P}(\tilde{\mathbf{G}} = \tilde{\mathbf{G}}') \, \mathbb{P}(R = R') \\ &\le H(\mathbf{A}'|\tilde{\mathbf{G}}, R) + 2m\log(4n). \end{aligned}$$

■

By a similar analysis we can bound the entropy of $\tilde{\mathbf{G}}$.

Lemma 30 $H(\tilde{\mathbf{G}}) \le m\log(|B_d|) + 2m\log(2n)$.

Proof We claim that $\tilde{\mathbf{G}}$ can be written with at most $m\log(|B_d|) + 2m\log(2n)$ bits, therefore its entropy is at most $m\log(|B_d|) + 2m\log(2n)$ (by Eq. (C.1)). To prove this, note that $\tilde{\mathbf{G}}$ consists of rows of $\mathbf{A}$ which are sampled from B_d, and recall that $\tilde{\mathbf{G}}$ has exactly m unique rows. Therefore can first write down all the m unique rows of $\tilde{\mathbf{G}}$ using $m\log(|B_d|)$ bits, and then for each of the at most $(2m - m_{\mathrm{U}}) \le 2m$ rows of $\tilde{\mathbf{G}}$ we can write down which of the unique rows it is with $\lceil \log(m) \rceil \le \log(2n)$ bits each, for a total of at most $m\log(|B_d|) + 2m\log(2n)$ bits. ■

The following lemma shows that Y is a deterministic function of $\tilde{\mathbf{G}}$, $\mathsf{Message}$ and R.

Lemma 31 *The queries $\{\mathbf{x}_i, i \in [m]\}$ are a deterministic function of* $\mathbf{G}$, $\mathsf{Message}$ *and R. Therefore* $\mathbf{Y} = g(\tilde{\mathbf{G}}, \mathsf{Message}, R)$ *for some function g.*

Proof Given $\mathbf{A}$ and R we observe that the Player is constrained to using a deterministic algorithm to both construct message $\mathsf{Message}$ to store and to determine the queries $(\mathbf{x}_1, \ldots, \mathbf{x}_m)$. Therefore we see that for any i there exists a function ϕ_i such that for any $\mathsf{Message}$, R, and responses $(\mathbf{g}_1, \ldots, \mathbf{g}_{i-1})$, $\mathbf{x}_i = \phi_i(\mathsf{Message}, R, \mathbf{x}_1, \mathbf{g}_1, \ldots, \mathbf{x}_{i-1}, \mathbf{g}_{i-1})$. Next we remark that there exists a function ϕ_i' such that $\mathbf{x}_i = \phi_i'(\mathsf{Message}, R, \mathbf{g}_1, \ldots, \mathbf{g}_{i-1})$. We can prove this by induction. For the base case we simply note that $\phi_1' = \phi_1$ and $\mathbf{x}_1 = \phi_1'(\mathsf{Message}, R)$. Next assume the inductive hypothesis that for any $j \le i-1$ we have $\mathbf{x}_j = \phi_j'(\mathsf{Message}, R, \mathbf{g}_1, \ldots, \mathbf{g}_{j-1})$. Then

$$\begin{aligned} \mathbf{x}_i &= \phi_i(\mathsf{Message}, R, \mathbf{x}_1, \mathbf{g}_1, \ldots, \mathbf{x}_{i-1}, \mathbf{g}_{i-1}) \\ &= \phi_i(\mathsf{Message}, R, \phi_1'(M, R), \mathbf{g}_1, \ldots, \phi_{i-1}'(\mathsf{Message}, R, \mathbf{g}_1, \ldots, \mathbf{g}_{i-2}), \mathbf{g}_{i-1}). \end{aligned}$$

Thus we define

$$\phi'_i(\mathsf{Message}, R, \mathbf{g}_1, \dots, \mathbf{g}_{i-1}) := \phi_i\Big(\mathsf{Message}, R, \phi'_1(M, R), \mathbf{g}_1, \dots, \phi'_{i-1}(\mathsf{Message}, R, \mathbf{g}_1, \dots, \mathbf{g}_{i-2}), \mathbf{g}_{i-1}\Big).$$

Therefore given $(\mathsf{Message}, R, \mathbf{G})$, it is possible to reconstruct the queries $\mathbf{X}$. Since $\mathbf{Y}$ is a deterministic function of $(\mathbf{X}, \mathbf{G}, \mathsf{Message}, R)$ in the Orthogonal Vector Game, and $\mathbf{G}$ is just the first m rows of $\tilde{\mathbf{G}}$, $\mathbf{Y} = g(\tilde{\mathbf{G}}, \mathsf{Message}, R)$ for some function g. ■

As sketched out in Section 3.2, in the following two lemmas we compute $I(\mathbf{A}'; \mathbf{Y}|\tilde{\mathbf{G}}, R)$ in two different ways.

Lemma 32 $I(\mathbf{A}'; \mathbf{Y}|\tilde{\mathbf{G}}, R) \le M$.

Proof We note that by Lemma 31, $\mathbf{Y} = g(\tilde{\mathbf{G}}, \mathsf{Message}, R)$ and therefore by the data processing inequality,

$$I(\mathbf{A}'; \mathbf{Y}|\tilde{\mathbf{G}}, R) \le I(\mathbf{A}'; \tilde{\mathbf{G}}, \mathsf{Message}, R|\tilde{\mathbf{G}}, R) = I(\mathbf{A}'; \mathsf{Message}|\tilde{\mathbf{G}}, R).$$

We conclude by noting that

$$I(\mathbf{A}'; \mathsf{Message}|\tilde{\mathbf{G}}, R) \le H(\mathsf{Message}|\tilde{\mathbf{G}}, R) \le H(\mathsf{Message}) \le M$$

where in the last step we use the fact that Message is M-bit long. ■

Lemma 33 *Suppose $\mathbf{Y}$ wins the Two Player Orthogonal Vector Game with failure probability at most δ. Then if $\{B_d\}_{d=1}^{\infty}$ is a memory sensitive base with constant $c_B > 0$,*

$$I(\mathbf{A}'; \mathbf{Y}|\tilde{\mathbf{G}}, R) \ge (n - m)(1 - \delta)\frac{c_B}{2}k - 4m\log(4n).$$

Proof We have

$$I(\mathbf{A}'; \mathbf{Y}|\tilde{\mathbf{G}}, R) = H(\mathbf{A}'|\tilde{\mathbf{G}}, R) - H(\mathbf{A}'|\mathbf{Y}, \tilde{\mathbf{G}}, R). \tag{C.2}$$

We will lower bound $I(\mathbf{A}'; \mathbf{Y}|\tilde{\mathbf{G}}, R)$ by providing a lower bound for $H(\mathbf{A}'|\tilde{\mathbf{G}}, R)$ and an upper bound for $H(\mathbf{A}'|\mathbf{Y}, \tilde{\mathbf{G}}, R)$. To that end, by Lemma 29 we have

$$H(\mathbf{A}'|\tilde{\mathbf{G}}, R) \ge H(\mathbf{A}|\tilde{\mathbf{G}}, R) - 2m\log(4n).$$

We can lower bound $H(\mathbf{A}|\tilde{\mathbf{G}}, R)$ as follows,

$$\begin{aligned} H(\mathbf{A}|\tilde{\mathbf{G}}, R) &= H(\mathbf{A}|R) - I(\mathbf{A}; \tilde{\mathbf{G}}|R) = H(\mathbf{A}|R) - \Big(H(\tilde{\mathbf{G}}|R) - H(\tilde{\mathbf{G}}|\mathbf{A}, R)\Big) \\ &\ge H(\mathbf{A}|R) - H(\tilde{\mathbf{G}}|R) \ge H(\mathbf{A}|R) - H(\tilde{\mathbf{G}}) \end{aligned}$$

where the inequalities use the fact that entropy is non-negative, and that conditioning reduces entropy. We now note that since $\mathbf{A}$ is independent of R, $H(\mathbf{A}|R) = H(\mathbf{A})$. Thus we have

$$H(\mathbf{A}'|\tilde{\mathbf{G}}, R) \ge H(\mathbf{A}) - H(\tilde{\mathbf{G}}) - 2m\log(4n),$$

and so recalling Lemma 30 and that $H(\mathbf{A}) = n\log(|B_d|)$ we conclude

$$H(\mathbf{A}'|\tilde{\mathbf{G}}, R) \geq n\log(|B_d|) \ - m\log(|B_d|) - 4m\log(4n). \tag{C.3}$$

Next we upper bound $H(\mathbf{A}'|\mathbf{Y}, \tilde{\mathbf{G}}, R)$. First we note

$$H(\mathbf{A}'|\mathbf{Y}, \tilde{\mathbf{G}}, R) \leq H(\mathbf{A}'|\mathbf{Y}) \leq (n-m)H(\mathbf{a}|\mathbf{Y})$$

where $\mathbf{a}$ is any one of the rows of $\mathbf{A}'$. Next recall that if $\mathbf{Y} = (\mathbf{y}_1, \ldots, \mathbf{y}_k)^\top \in \mathbb{R}^{k\times d}$ wins the Two Player Orthogonal Vector Game then for any $\mathbf{y}_i$, $\|\mathbf{A}\mathbf{y}_i\|_\infty / \|\mathbf{y}_i\|_2 \leq d^{-4}$ and for $S_i = \mathsf{span}(\mathbf{y}_1, \ldots, \mathbf{y}_{i-1})$, we have $\left\|\mathrm{Proj}_{S_i}(\mathbf{y}_i)\right\|_2 / \|\mathbf{y}_i\|_2 \leq 1 - (1/d^2)$. Since these properties are normalized by $\|\mathbf{y}_i\|_2$ we may, without loss of generality, overload our notation for $\mathbf{y}_i$ by normalizing, we set $\mathbf{y}_i \leftarrow \mathbf{y}_i / \|\mathbf{y}_i\|_2$. Note that the entropy of $\mathbf{a}$ conditioned on any value of $\mathbf{Y} = \mathbf{Y}'$ is bounded above by the entropy of $\mathbf{a}'$ where the law of $\mathbf{a}'$ corresponds to the uniform distribution over the set $\left\{\mathbf{a} \in B_d \text{ s.t. } \|\mathbf{Y}'\mathbf{a}\|_\infty \leq \frac{1}{d^4}\right\}$ and this has entropy $\log\left(\left|\left\{\mathbf{a} \in B_d \text{ s.t. } \|\mathbf{Y}'\mathbf{a}\|_\infty \leq \frac{1}{d^4}\right\}\right|\right)$. We also note that since $\mathbf{Y} = g(\tilde{\mathbf{G}}, \mathsf{Message}, R)$ (Lemma 31), and all of $\tilde{\mathbf{G}}$, $\mathsf{Message}$ and R take values on a finite support, $\mathbf{Y}$ also takes values on some finite support ($\mathbf{Y}$ can still be real-valued, but it's support must be finite). Therefore we can write,

$$\begin{aligned}
H(\mathbf{a}|\mathbf{Y}) &= \sum_{\mathbf{Y}'} \left[H\left(\mathbf{a}|\mathbf{Y} = \mathbf{Y}'\right)\right] \mathbb{P}(\mathbf{Y} = \mathbf{Y}') \\
&= \sum_{\mathbf{Y}':\mathbf{Y}' \text{ wins}} \left[H\left(\mathbf{a}|\mathbf{Y} = \mathbf{Y}'\right)\right] \mathbb{P}(\mathbf{Y} = \mathbf{Y}') + \sum_{\mathbf{Y}':\mathbf{Y}' \text{ loses}} \left[H\left(\mathbf{a}|\mathbf{Y} = \mathbf{Y}'\right)\right] \mathbb{P}(\mathbf{Y} = \mathbf{Y}') \\
&\leq \sum_{\mathbf{Y}':\mathbf{Y}' \text{ wins}} \left[H\left(\mathbf{a}|\mathbf{Y} = \mathbf{Y}'\right)\right] \mathbb{P}(\mathbf{Y} = \mathbf{Y}') + p(\mathbf{Y} \text{ loses})H(\mathbf{a}) \\
&\leq \sum_{\mathbf{Y}':\mathbf{Y}' \text{ wins}} \log\left(\left|\left\{\mathbf{a} \in B_d \text{ s.t. } \left\|\mathbf{Y}'\mathbf{a}\right\|_\infty \leq \frac{1}{d^4}\right\}\right|\right) \mathbb{P}(\mathbf{Y} = \mathbf{Y}') + p(\mathbf{Y} \text{ loses})H(\mathbf{a}) \\
&\leq (1-\delta)\log\left(\max_{\mathbf{Y}':\mathbf{Y}' \text{wins}} \left|\left\{\mathbf{a} \in B_d \text{ s.t. } \left\|\mathbf{Y}'\mathbf{a}\right\|_\infty \leq \frac{1}{d^4}\right\}\right|\right) + \delta\log(|B_d|).
\end{aligned} \tag{C.4}$$

Thus for any $\mathbf{Y} = (\mathbf{y}_1, \ldots, \mathbf{y}_k)$ which wins the Two Player Orthogonal Vector Game, we will upper bound $\left|\left\{\mathbf{a} \in B_d \text{ s.t. } \|\mathbf{Y}\mathbf{a}\|_\infty \leq 1/d^4\right\}\right|$. We will use the following lemma (proved in Appendix D.2) which constructs a partial orthonormal basis from a set of robustly linearly independent vectors.

Lemma 34 *Let $\delta \in (0,1]$ and suppose we have $n_0 \leq d$ unit norm vectors $\mathbf{y}_1, \ldots, \mathbf{y}_{n_0} \in \mathbb{R}^d$. Let $S_i := \mathsf{span}\left(\{\mathbf{y}_1, \ldots, \mathbf{y}_i\}\right), S_0 := \phi$. Suppose that for any $i \in [n_0]$,*

$$\left\|\mathrm{Proj}_{S_{i-1}}(\mathbf{y}_i)\right\|_2 \leq 1 - \delta.$$

Let $\mathbf{Y} = (\mathbf{y}_1, \ldots, \mathbf{y}_{n_0}) \in \mathbb{R}^{d\times n_0}$. There exists $n_1 = \lfloor n_0/2 \rfloor$ orthonormal vectors $\mathbf{m}_1, \ldots, \mathbf{m}_{n_1}$ such that, letting $\mathbf{M} = (\mathbf{m}_1, \ldots, \mathbf{m}_{n_1})$, for any $\mathbf{a} \in \mathbb{R}^d$,

$$\left\|\mathbf{M}^\top \mathbf{a}\right\|_\infty \leq \frac{d}{\delta}\left\|\mathbf{Y}^\top \mathbf{a}\right\|_\infty.$$

By Lemma 34 there exist $\lfloor k/2 \rfloor$ orthonormal vectors $\{\mathbf{z}_1, \dots, \mathbf{z}_{\lfloor k/2 \rfloor}\}$ such that for $\mathbf{Z} = (\mathbf{z}_1, \dots, \mathbf{z}_{\lfloor k/2 \rfloor})$ and for any $\mathbf{a} \in \mathbb{R}^d$,

$$\left\| \mathbf{Z}^\top \mathbf{a} \right\|_\infty \le d^3 \left\| \mathbf{Y}^\top \mathbf{a} \right\|_\infty . \tag{C.5}$$

Define $S := \{\mathbf{a} \in B_d \text{ s.t. } \|\mathbf{Z}\mathbf{a}\|_\infty \le 1/d\}$. By Eq. (C.5),

$$\left| \left\{ \mathbf{a} \in B_d \text{ s.t. } \left\| \mathbf{Y}^\top \mathbf{a} \right\|_\infty \le \frac{1}{d^4} \right\} \right| \le |S| .$$

Observing that B_d is a memory-sensitive base as per Definition 6,

$$|S| \le \mathbb{P}_{\mathbf{a} \sim \mathsf{Unif}(B_d)} \left(\left\| \mathbf{Z}^\top \mathbf{a} \right\|_\infty \le 1/d \right) |B_d| \le 2^{-c_B k/2} |B_d| ,$$

for some constant $c_B > 0$. From Eq. (C.4),

$$H(\mathbf{a}|\mathbf{Y}) \le (1-\delta) \log \left(2^{-c_B k/2} |B_d| \right) + \delta \log (|B_d|) = \log(|B_d|) - (1-\delta) c_B k/2.$$

So

$$H(\mathbf{A}'|\mathbf{Y}) \le (n-m) \left(\log(|B_d|) - (1-\delta) c_B k/2 \right) .$$

Recalling Eq. (C.2) and Eq. (C.3) we conclude,

$$\begin{aligned} I(\mathbf{A}'; \mathbf{Y}|\tilde{\mathbf{G}}, R) &\ge \log(|B_d|)(n-m) - 4m \log(4n) - (n-m) \left(\log(|B_d|) - (1-\delta) c_B k/2 \right) \\ &= (n-m)(1-\delta) \frac{c_B}{2} k - 4m \log(4n). \end{aligned}$$

■

By combining Lemma 32 and 33, we prove the memory-query tradeoff for the Orthogonal Vector Game.

Theorem 16 *Suppose for a memory-sensitive base $\{B_d\}_{d=1}^\infty$ with constant $c_B > 0$, $\mathbf{A} \sim \mathsf{Unif}(B_d^n)$. Given $\mathbf{A}$ let the oracle response $\mathbf{g}_i$ to any query $\mathbf{x}_i \in \mathbb{R}^d$ be any (possibly randomized) function from $\mathbb{R}^d \to \{\mathbf{a}_1, \dots, \mathbf{a}_{d/2}\}$, where $\mathbf{a}_i$ is the i^{th} row of $\mathbf{A}$ (note this includes the subgradient response in the Orthogonal Vector Game). Set $k = \lceil 60M/(c_B d) \rceil$ and assume $k \ge \lceil 30 \log(4d)/c_B \rceil$. For these values of k and M, if the Player wins the Orthogonal Vector Game with probability at least $1/3$, then $m \ge d/5$.*

Proof
By Lemma 32 we have that

$$I(\mathbf{A}'; \mathbf{Y}|\tilde{\mathbf{G}}, R) \le M.$$

However if the player returns some $\mathbf{Y}$ which wins the Orthogonal Vector Game with failure probability at most δ then by Lemma 33,

$$I(\mathbf{A}'; \mathbf{Y}|\tilde{\mathbf{G}}, R) \ge (n-m)(1-\delta) \frac{c_B}{2} k - 4m \log(4n).$$

Therefore we must have

$$m \geq \frac{n(1-\delta)c_B k/2 - M}{(1-\delta)c_B k/2 + 4\log(4n)}.$$

Note that $k = \lceil \frac{20\log(4d)}{(1-\delta)c_B} \rceil$ for $\delta = 2/3$. Therefore $(1-\delta)c_B k/2 \geq 4\log(4n)$. Using this and the fact that $n = d/2$,

$$m \geq \frac{n(1-\delta)c_B k/2 - M}{(1-\delta)c_B k} \geq \frac{n}{2} - \frac{M}{(1-\delta)c_B k} \geq \frac{n}{2} - \frac{d}{20} \geq d/5.$$

■

Appendix D. Additional Helper Lemmas

D.1. Useful concentration bounds

In this section we establish some concentration bounds which we repeatedly use in our analysis.

Definition 35 (sub-Gaussian random variable) *A zero-mean random variable X is* sub-Gaussian *if for some constant σ and for all $\lambda \in \mathbb{R}$, $\mathbb{E}[e^{\lambda X}] \leq e^{\lambda^2\sigma^2/2}$. We refer to σ as the* sub-Gaussian parameter of X. *We also define the* sub-Gaussian norm $\|X\|_{\psi_2}$ *of a sub-Gaussian random variable X as follows,*

$$\|X\|_{\psi_2} := \inf\left\{K > 0 \text{ such that } \mathbb{E}[\exp(X^2/K^2)] \leq 2\right\}.$$

We use that $X \sim \mathsf{Unif}(\{\pm 1\})$ is sub-Gaussian with parameter $\sigma = 1$ and sub-Gaussian norm $\|X\|_{\psi_2} \leq 2$. The following fact about sums of independent sub-Gaussian random variables is useful in our analysis.

Fact 36 *(Vershynin, 2018) For any n, if $\{X_i, i \in [n]\}$ are independent zero-mean sub-Gaussian random variables with sub-Gaussian parameter σ_i, then $X' = \sum_{i\in[n]} X_i$ is a zero-mean sub-Gaussian with parameter $\sqrt{\sum_{i\in[n]}\sigma_i^2}$ and norm bounded as $\|X'\|_{\psi_2} \leq \alpha\sqrt{\sum_{i\in[n]}\|X_i\|_{\psi_2}^2}$, for some universal constant α.*

Using Fact 36 we obtain the following corollary which we use directly in our analysis.

Corollary 37 *For $\mathbf{v} \sim (1/\sqrt{d})\mathsf{Unif}(\mathcal{H}_d)$ and any fixed vector $\mathbf{x} \in \mathbb{R}^d$, $\mathbf{x}^\top\mathbf{v}$ is a zero-mean sub-Gaussian with sub-Gaussian parameter at most $\|\mathbf{x}\|_2/\sqrt{d}$ and sub-Gaussian norm at most $\|\mathbf{x}^\top\mathbf{v}\|_{\psi_2} \leq 2\alpha\|\mathbf{x}\|_2/\sqrt{d}$.*

The following result bounds the tail probability of sub-Gaussian random variables.

Fact 38 *(Wainwright, 2019) For a random variable X which is zero-mean and sub-Gaussian with parameter σ, then for any t, $\mathbb{P}(|X| \geq t) \leq 2\exp(-t^2/(2\sigma^2))$.*

We will use the following concentration bound for quadratic forms.

Theorem 39 (Hanson-Wright inequality (Hanson and Wright, 1971; Rudelson and Vershynin, 2013)) *Let $\mathbf{x} = (X_1, \dots, X_d) \in \mathbb{R}^d$ be a random vector with i.i.d components X_i which satisfy $\mathbb{E}[X_i] = 0$ and $\|X_i\|_{\psi_2} \leq K$ and let $\mathbf{M} \in \mathbb{R}^{n \times n}$. Then for some absolute constant $c_{hw} > 0$ and for every $t \geq 0$,*

$$\max\left\{\mathbb{P}\left(\mathbf{x}^\top \mathbf{M}\mathbf{x} - \mathbb{E}\left[\mathbf{x}^\top \mathbf{M}\mathbf{x}\right] > t\right), \mathbb{P}\left(\mathbb{E}\left[\mathbf{x}^\top \mathbf{M}\mathbf{x}\right] - \mathbf{x}^\top \mathbf{M}\mathbf{x} > t\right)\right\}$$
$$\leq \exp\left(-c_{hw} \min\left\{\frac{t^2}{K^4 \|\mathbf{M}\|_F^2}, \frac{t}{K^2 \|\mathbf{M}\|_2}\right\}\right).$$

where $\|\mathbf{M}\|_2$ is the operator norm of $\mathbf{M}$ and $\|\mathbf{M}\|_F$ is the Frobenius norm of $\mathbf{M}$.

The following lemma will let us analyze projections of sub-Gaussian random variables (getting concentration bounds which look like projections of Gaussian random variables).

Lemma 40 *Let $\mathbf{x} = (X_1, \dots, X_d) \in \mathbb{R}^d$ be a random vector with i.i.d sub-Gaussian components X_i which satisfy $\mathbb{E}[X_i] = 0$, $\|X_i\|_{\psi_2} \leq K$, and $\mathbb{E}[\mathbf{x}\mathbf{x}^\top] = s^2 \mathbf{I}_d$. Let $\mathbf{Z} \in \mathbb{R}^{d \times r}$ be a matrix with orthonormal columns $(\mathbf{z}_1, \dots, \mathbf{z}_r)$. There exists some absolute constant $c_{hw} > 0$ (which comes from the Hanson-Wright inequality) such that for $t \geq 0$,*

$$\max\left\{\mathbb{P}\left(\left\|\mathbf{Z}^\top \mathbf{x}\right\|_2^2 - rs^2 > t\right), \mathbb{P}\left(rs^2 - \left\|\mathbf{Z}^\top \mathbf{x}\right\|_2^2 > t\right)\right\} \leq \exp\left(-c_{hw} \min\left\{\frac{t^2}{rK^4}, \frac{t}{K^2}\right\}\right). \tag{D.1}$$

Proof We will use the Hanson-Wright inequality (see Theorem 39). We begin by computing $\mathbb{E}\left[\|\mathbf{Z}^\top \mathbf{x}\|_2^2\right]$,

$$\mathbb{E}\left[\left\|\mathbf{Z}^\top \mathbf{x}\right\|_2^2\right] = \mathbb{E}\left[\mathbf{x}^\top \mathbf{Z}\mathbf{Z}^\top \mathbf{x}\right] = \mathbb{E}\left[\mathbf{tr}\left(\mathbf{Z}^\top \mathbf{x}\mathbf{x}^\top \mathbf{Z}\right)\right] = \mathbf{tr}\left(\mathbf{Z}^\top \mathbb{E}\left[\mathbf{x}\mathbf{x}^\top\right] \mathbf{Z}\right)$$
$$= s^2\, \mathbf{tr}\left(\mathbf{Z}^\top \mathbf{Z}\right) = s^2\, \mathbf{tr}(\mathbf{I}_{r \times r}) = rs^2.$$

Next we bound the operator norm and Frobenius-norm of $\mathbf{Z}\mathbf{Z}^\top$. The operator norm of $\mathbf{Z}\mathbf{Z}^\top$ is bounded by 1 since the columns of $\mathbf{Z}$ are orthonormal vectors. For the Frobenius-norm of $\mathbf{Z}\mathbf{Z}^\top$, note that

$$\left\|\mathbf{Z}\mathbf{Z}^\top\right\|_{\mathrm{F}}^2 = \mathbf{tr}\left(\mathbf{Z}\mathbf{Z}^\top \mathbf{Z}\mathbf{Z}^\top\right) = \mathbf{tr}\left(\mathbf{Z}\mathbf{Z}^\top\right) = \sum_{i \in [r]} \mathbf{tr}\left(\mathbf{z}_i \mathbf{z}_i^\top\right) = \sum_{i \in [r]} \|\mathbf{z}_i\|_2^2 = r.$$

Applying the Hanson-Wright inequality (Theorem 39) with these bounds on $\mathbb{E}\left[\|\mathbf{Z}^\top \mathbf{x}\|_2^2\right]$, $\|\mathbf{Z}\mathbf{Z}^\top\|_2^2$, and $\|\mathbf{Z}\mathbf{Z}^\top\|_{\mathrm{F}}^2$ yields (D.1) and completes the proof. ■

Corollary 41 *Let $\mathbf{v} \sim (1/\sqrt{d})\mathsf{Unif}\left(\mathcal{H}_d\right)$ and let S be a fixed r-dimensional subspace of $\mathbb{R}^d$ (independent of $\mathbf{v}$). Then with probability at least $1 - 2\exp(c_{hw})/d^5$,*

$$\|\mathrm{Proj}_S(\mathbf{v})\|_2 \leq \sqrt{\frac{30 \log(d) r}{d}}.$$

Proof Let $\mathbf{z}_1, \ldots, \mathbf{z}_r$ be an orthonormal basis for subspace S. Note that

$$\|\mathrm{Proj}_S(\mathbf{v})\|_2^2 = \left\|\mathbf{Z}^\top \mathbf{v}\right\|_2^2.$$

Next, for $\mathbf{v} = (\mathbf{v}_1, \ldots, \mathbf{v}_d)$ we have $\mathbb{E}[\mathbf{v}\mathbf{v}^\top] = (1/d)\mathbf{I}$ and $\|\mathbf{v}_i\|_{\psi_2} \leq 2/\sqrt{d}$, thus we may set $s = 1/\sqrt{d}$ and $K = 2/\sqrt{d}$ and apply Lemma 40,

$$\mathbb{P}\left(\left|\|\mathrm{Proj}_S(\mathbf{v})\|_2^2 - \frac{r}{d}\right| > t\right) \leq 2\exp\left(-c_{\mathrm{hw}} \min\left\{\frac{d^2t^2}{16r}, \frac{dt}{4}\right\}\right).$$

Picking $t = 20\log(d)r/d$ concludes the proof. ∎

D.2. A property of robustly linearly independent vectors: Proof of Lemma 34

Proof Since for any $i \in [n_0]$, $\left\|\mathrm{Proj}_{S_{i-1}}(\mathbf{y}_i)\right\|_2 \leq 1 - \delta < 1$,

$$\mathsf{dim}\ \mathsf{span}(\mathbf{y}_1, \ldots, \mathbf{y}_{n_0}) = n_0.$$

Therefore we may construct an orthonormal basis $\mathbf{b}_1, \ldots, \mathbf{b}_{n_0}$ via the Gram–Schmidt process,

$$\mathbf{b}_i := \frac{\mathbf{y}_i - \mathrm{Proj}_{S_{i-1}}(\mathbf{y}_i)}{\left\|\mathbf{y}_i - \mathrm{Proj}_{S_{i-1}}(\mathbf{y}_i)\right\|_2}.$$

Let $\mathbf{B} = (\mathbf{b}_1, \ldots, \mathbf{b}_{n_0}) \in \mathbb{R}^{d \times n_0}$ and note that with this construction there exists a vector of coefficients $\mathbf{c}^{(i)} \in \mathbb{R}^{n_0}$ such that

$$\mathbf{y}_i = \sum_{j \in [i]} \mathbf{c}_j^{(i)} \mathbf{b}_j = \mathbf{B}\mathbf{c}^{(i)}.$$

Let $\mathbf{C} = \left(\mathbf{c}^{(1)}, \ldots, \mathbf{c}^{(n_0)}\right) \in \mathbb{R}^{n_0 \times n_0}$ and observe that $\mathbf{Y} = \mathbf{B}\mathbf{C}$. Denote the singular values of $\mathbf{C}$ as $\sigma_1 \geq \cdots \geq \sigma_{n_0}$. We aim to bound the singular values of $\mathbf{C}$ from below. To this end, observe that $\mathbf{C}$ is an upper triangular matrix such that for any $i \in [n_0]$, $|\mathbf{C}_{ii}| \geq \sqrt{\delta}$. Indeed, note that

$$\mathbf{C}_{ii}^2 = \left\|\mathbf{y}_i - \mathrm{Proj}_{S_{i-1}}(\mathbf{y}_i)\right\|_2^2 = \|\mathbf{y}_i\|_2^2 - \left\|\mathrm{Proj}_{S_{i-1}}(\mathbf{y}_i)\right\|_2^2 \geq 1 - (1-\delta)^2 \geq \delta.$$

Therefore,

$$\prod_{i \in [n_0]} \sigma_i = \det(\mathbf{C}) \geq \delta^{n_0/2}$$

where in the last step we use the fact that the determinant of a triangular matrix is the product of its diagonal entries. Next we consider the singular value decomposition of $\mathbf{C}$: let $\mathbf{U} \in \mathbb{R}^{n_0 \times n_0}, \mathbf{V} \in \mathbb{R}^{n_0 \times n_0}$ be orthogonal matrices and $\mathbf{\Sigma} \in \mathbb{R}^{n_0 \times n_0}$ be the diagonal matrix such that $\mathbf{C} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top$. For the remainder of the proof assume without loss of generality that the columns of $\mathbf{Y}$ and $\mathbf{B}$ are ordered such that $\Sigma_{ii} = \sigma_i$. We will upper bound the singular values of $\mathbf{C}$ as follows: observe that since $\|\mathbf{y}_i\|_2 = 1$, for any vector $\mathbf{w}$ such that $\|\mathbf{w}\|_2 \leq 1$,

$$\left\|\mathbf{Y}^\top \mathbf{w}\right\|_2 \leq \sqrt{d}, \qquad \text{and therefore} \qquad \left\|\mathbf{C}^\top \mathbf{B}^\top \mathbf{w}\right\|_2 \leq \sqrt{d}.$$

In particular consider $\mathbf{w} = \mathbf{B}\mathbf{u}_i$ where $\mathbf{u}_i$ denotes the i^{th} column of $\mathbf{U}$. Since $\mathbf{B}^\top\mathbf{B} = \mathbf{I}$ and $\mathbf{U}, \mathbf{V}$ are orthogonal we have

$$\left\|\mathbf{C}^\top\mathbf{B}^\top\mathbf{w}\right\|_2 = \|\mathbf{V}\mathbf{\Sigma}\mathbf{e}_i\|_2 = \sigma_i \le \sqrt{d}.$$

Note that for $m \in [n_0]$, if $i > n_0 - m$ then $\sigma_i \le \sigma_{n_0-m}$ and therefore

$$\delta^{n_0/2} \le \prod_{i\in[n_0]} \sigma_i \le \sigma_{n_0-m}^m \prod_{i\in[n_0-m]} \sigma_i \le \sigma_{n_0-m}^m \left(\sqrt{d}\right)^{n_0-m} \implies \sigma_{n_0-m} \ge \left(\delta^{n_0} d^{(m-n_0)}\right)^{\frac{1}{2m}}.$$

In particular when $m = n_1 = \lceil n_0/2 \rceil$ we have $\sigma_{n_0-n_1} \ge \delta/\sqrt{d}$. Therefore for any $i \le n_0 - n_1 = \lfloor \frac{n_0}{2} \rfloor$,

$$\sigma_i \ge \delta/\sqrt{d}.$$

Recall that $\mathbf{u}_i$ denotes the i^{th} column of $\mathbf{U}$ and let $\mathbf{v}_i$ denote the i^{th} column of $\mathbf{V}$. Observe that

$$\mathbf{U}\mathbf{\Sigma} = \mathbf{B}^\top\mathbf{Y}\mathbf{V} \qquad \implies \qquad \mathbf{u}_i = \frac{1}{\sigma_i}\mathbf{B}^\top\mathbf{Y}\mathbf{v}_i.$$

Extend the basis $\{\mathbf{b}_1, \ldots, \mathbf{b}_{n_0}\}$ to $\mathbb{R}^d$ and let $\tilde{\mathbf{B}} = (\mathbf{B}, \mathbf{b}_{n_0+1}, \ldots, \mathbf{b}_d)$ be the $d \times d$ matrix corresponding to this orthonormal basis. Note that if $j > n_0$, then for any $\mathbf{y}_i$, $\mathbf{b}_j^\top \mathbf{y}_i = 0$. Define

$$\tilde{\mathbf{u}}_i := \frac{1}{\sigma_i}\tilde{\mathbf{B}}^\top\mathbf{Y}\mathbf{v}_i \in \mathbb{R}^d.$$

For $i \in [n_1]$ define $\mathbf{m}_i = \tilde{\mathbf{B}}\tilde{\mathbf{u}}_i$ and let $\mathbf{M} = (\mathbf{m}_1, \ldots, \mathbf{m}_{n_1})$. Note that $\{\mathbf{m}_1, \ldots, \mathbf{m}_{n_1}\}$ is an orthonormal set. The result then follows since for any $\mathbf{a} \in \mathbb{R}^d$,

$$\left|\mathbf{m}_i^\top\mathbf{a}\right| = \left|\tilde{\mathbf{u}}_i^\top\tilde{\mathbf{B}}^\top\mathbf{a}\right| = \left|\frac{1}{\sigma_i}\mathbf{v}_i^\top\mathbf{Y}^\top\tilde{\mathbf{B}}\tilde{\mathbf{B}}^\top\mathbf{a}\right| = \frac{1}{\sigma_i}\left|\mathbf{v}_i^\top\mathbf{Y}^\top\mathbf{a}\right| \le \frac{1}{\sigma_i}\|\mathbf{v}_i\|_1 \left\|\mathbf{Y}^\top\mathbf{a}\right\|_\infty \le \frac{d}{\delta}\left\|\mathbf{Y}^\top\mathbf{a}\right\|_\infty.$$

■

D.3. From Constrained to Unconstrained Lower Bounds

Here we provide a generic, black-box reduction from approximate Lipschitz convex optimization over a unit ball to unconstrained approximate Lipschitz convex optimization. The reduction leverages a natural extension of functions on the unit ball to $\mathbb{R}^d$, provided and analyzed in the following lemma.

Lemma 42 *Let $f : B_2^d \to \mathbb{R}$ be a convex, L-Lipschitz function and let $g_{r,L}^f : \mathbb{R}^d \to \mathbb{R}$ and $h_{r,L}^f : \mathbb{R}^d \to \mathbb{R}$ be defined for all $\mathbf{x} \in \mathbb{R}^d$ and some $r \in (0,1)$ as*

$$g_{r,L}^f(\mathbf{x}) := \begin{cases} \max\{f(\mathbf{x}), h_{r,L}^f(\mathbf{x})\} & \text{for } \|\mathbf{x}\|_2 < 1 \\ h_{r,L}^f(\mathbf{x}) & \text{for } \|\mathbf{x}\|_2 \ge 1 \end{cases} \text{ and } h_{r,L}^f(\mathbf{x}) := f(\mathbf{0}) + \frac{L}{1-r}\left((1+r)\|\mathbf{x}\|_2 - 2r\right).$$

Then, $g_{r,L}^f(\mathbf{x})$ is a convex, $L\left(\frac{1+r}{1-r}\right)$-Lipschitz function with $g_{r,L}^f(\mathbf{x}) = f(\mathbf{x})$ for all $\mathbf{x}$ with $\|\mathbf{x}\|_2 \le r$.

Proof For all $\mathbf{x} \in B_2^d$ note that

$$h_{r,L}^f(\mathbf{x}) - f(\mathbf{x}) = f(\mathbf{x}) - f(\mathbf{0}) + \frac{L}{1-r}\left((1+r)\|\mathbf{x}\|_2 - 2r\right). \tag{D.2}$$

Now $|f(\mathbf{x}) - f(\mathbf{0})| \le L\|\mathbf{x}\|_2$ since f is L-Lipschitz. Correspondingly, when $\|\mathbf{x}\|_2 = 1$ this implies

$$h_{r,L}^f(\mathbf{x}) - f(\mathbf{x}) \ge -L + \frac{L}{1-r}\left((1+r) - 2r\right) = 0\,.$$

Consequently, $h_{r,L}^f(\mathbf{x}) \ge f(\mathbf{x})$ when $\|\mathbf{x}\|_2 = 1$ and $g_{r,L}^f(\mathbf{x}) = h_{r,L}^f(\mathbf{x}) = \max\{f(\mathbf{x}), h(\mathbf{x})\}$. Further, since $h_{r,L}^f$ is $\left(\frac{1+r}{1-r}\right)$-Lipschitz and convex and f is L-Lipschitz and convex this implies that $g_{r,L}^f$ is $\left(\frac{1+r}{1-r}\right)$-Lipschitz and convex as well. Finally, if $\|\mathbf{x}\|_2 \le r$ then again by (D.2) and that $|f(\mathbf{x}) - f(\mathbf{0})| \le L\|\mathbf{x}\|_2$ we have

$$h_{r,L}^f(\mathbf{x}) - f(\mathbf{x}) \le L\|\mathbf{x}\|_2 + \frac{L}{1-r}\left((1+r)\|\mathbf{x}\|_2 - 2r\right) = \frac{2L}{1-r}\left(\|\mathbf{x}\|_2 - r\right) \le 0.$$

Therefore $g_{r,L}^f(\mathbf{x}) = f(\mathbf{x})$ for all $\mathbf{x}$ with $\|\mathbf{x}\|_2 \le r$. ■

Leveraging this extension we obtain the following result.

Lemma 43 *Suppose any M-memory constrained randomized algorithm must make (with high probability in the worst case) at least $\mathcal{T}_\epsilon$ queries to a first-order oracle for convex, L-Lipschitz $f : B_2^d \to \mathbb{R}$ in order to compute an ϵ-optimal point for some $\epsilon < L$.[4] Then any M-memory constrained randomized algorithm must make at least $\mathcal{T}_\epsilon/2$ queries to a first-order oracle (with high probability in the worst case) to compute an $\epsilon/2$-optimal point for a $\mathcal{O}(L^2/\epsilon)$-Lipschitz, convex $g : \mathbb{R}^d \to \mathbb{R}$ even though the minimizer of g is guaranteed to lie in B_2^d.*

Proof Let $\mathbf{x}^\star$ be a minimizer of f. By convexity, for all $\alpha \in [0,1]$ we have that

$$\begin{aligned} f(\alpha\mathbf{x}^\star) - f(\mathbf{x}^\star) &= f(\alpha\mathbf{x} + (1-\alpha)\mathbf{0}) - f(\mathbf{x}^\star) \\ &\le \alpha\left[f(\mathbf{x}^\star) - f(\mathbf{x}^\star)\right] + (1-\alpha)\left[f(\mathbf{0}) - f(\mathbf{x}^\star)\right] \\ &\le (1-\alpha)\|\mathbf{0} - \mathbf{x}^\star\|_2 \le (1-\alpha)L\,. \end{aligned}$$

Consequently, for all $\delta > 0$ and $\alpha_\delta \coloneqq 1 - \delta/L$ we have that $f(\alpha_\delta \mathbf{x}^\star)$ is δ-optimal.

Correspondingly, consider the function $g \coloneqq g_{\alpha_{\epsilon/2},L}^f$ as defined in Lemma 42. Note that g is $L\left(\frac{1+\alpha_\delta}{1-\alpha_\delta}\right) = \mathcal{O}(L^2/\delta)$ Lipschitz. The minimizer of g also lies in B_2^d since g monotonically increases for $\|\mathbf{x}\|_2 \ge 1$. Suppose there is an algorithm to compute an $\epsilon/2$-optimal minimizer of all $\mathcal{O}(L^2/\delta)$-Lipschitz, convex functions $g : \mathbb{R}^d \to \mathbb{R}$ whose minimizer lies in B_2^d, with high probability using $\mathcal{T}$ queries to a first order oracle for g. Note that this algorithm can be implemented instead with $2\mathcal{T}$ queries to f by simply querying $f(\mathbf{0})$ along with any query, and then using the defining of g from Lemma 42 (note that it is also possible to just compute and store $f(\mathbf{0})$ once in the beginning and hence only require $\mathcal{T} + 1$ queries, but this would require us to also analyze the number of bits

4. Note that 0 queries are need when $\epsilon \ge L$.

of precision to which $f(\mathbf{0})$ should be stored, which this proof avoids for simplicity). If $\mathbf{x}_{\rm out}$ is the output of the algorithm then

$$\frac{\epsilon}{2} \geq g(\mathbf{x}_{\rm out}) - \inf_x g(\mathbf{x}) \geq g(\mathbf{x}_{\rm out}) - g(\alpha_{\epsilon/2}\mathbf{x}^\star) \geq g(\mathbf{x}_{\rm out}) - f(\mathbf{x}^\star) - \frac{\epsilon}{2}\,.$$

Thus $g(\mathbf{x}_{\rm out}) \leq f(\mathbf{x}^\star) + \epsilon$. Let $\tilde{\mathbf{x}}_{\rm out} = \min\{1, \|\mathbf{x}_{\rm out}\|_2^{-1}\}\mathbf{x}_{\rm out}$. As g monotonically increases for $\|\mathbf{x}\|_2 \geq 1$, we have that $g(\mathbf{x}_{\rm out}) \geq g(\tilde{\mathbf{x}}_{\rm out}) = f(\tilde{\mathbf{x}}_{\rm out})$ and thus with $2\mathcal{T}$ queries to f the algorithm can compute an ϵ-approximate minimizer of f. Consequently, $\mathcal{T} \geq \mathcal{T}_\epsilon/2$ as desired and the result follows. ■

Though this section focused on the unit ℓ_2 norm ball in this work, the analysis in this section generalizes to arbitrary norms.