Graph Oracle Models, Lower Bounds, and Gaps for Parallel Stochastic Optimization

Blake Woodworth
Toyota Technological
Institute at Chicago
blake@ttic.edu

Jialei Wang
Two Sigma Investments
jialei.wang@twosigma.com

Adam Smith
Boston University
ads22@bu.edu

Brendan McMahan
Google
mcmahan@google.com

Nathan Srebro
Toyota Technological
Institute at Chicago
nati@ttic.edu

Abstract

We suggest a general oracle-based framework that captures different parallel stochastic optimization settings described by a dependency graph, and derive generic lower bounds in terms of this graph. We then use the framework and derive lower bounds for several specific parallel optimization settings, including delayed updates and parallel processing with intermittent communication. We highlight gaps between lower and upper bounds on the oracle complexity, and cases where the "natural" algorithms are not known to be optimal.

1 Introduction

Recently, there has been great interest in stochastic optimization and learning algorithms that leverage parallelism, including e.g. delayed updates arising from pipelining and asynchronous concurrent processing, synchronous single-instruction-multiple-data parallelism, and parallelism across distant devices. With the abundance of parallelization settings and associated algorithms, it is important to precisely formulate the problem, which allows us to ask questions such as "is there a better method for this problem than what we have?" and "what is the best we could possibly expect?"

Oracle models have long been a useful framework for formalizing stochastic optimization and learning problems. In an oracle model, we place limits on the algorithm's access to the optimization objective, but not what it may do with the information it receives. This allows us to obtain sharp lower bounds, which can be used to argue that an algorithm is optimal and to identify gaps between current algorithms and what might be possible. Finding such gaps can be very useful—for example, the gap between the first order optimization lower bound of Nemirovski et al. [20] and the best known algorithms at the time inspired Nesterov's accelerated gradient descent algorithm [21].

We propose an oracle framework for formalizing different parallel optimization problems. We specify the structure of parallel computation using an "oracle graph" which indicates how an algorithm accesses the oracle. Each node in the graph corresponds to a single stochastic oracle query, and that query (e.g. the point at which a gradient is calculated) must be computed using only oracle accesses in ancestors of the node. We generally think of each stochastic oracle access as being based on a single data sample, thus involving one or maybe a small number of vector operations.

In Section 3 we devise generic lower bounds for parallel optimization problems in terms of simple properties of the associated oracle graph, namely the length of the longest dependency chain and the total number of nodes. In Section 4 we study specific parallel optimization settings in which

many algorithms have been proposed, formulate them as graph-based oracle parallel optimization problems, instantiate our lower bounds, and compare them with the performance guarantees of specific algorithms. We highlight gaps between the lower bound and the best known upper bound and also situations where we can devise an optimal algorithm that matches the lower bound, but where this is not the "natural" and typical algorithm used in this settings. The latter indicates either a gap in our understanding of the "natural" algorithm or a need to depart from it.

Previously suggested models   Previous work studied communication lower bounds for parallel convex optimization where there are M machines each containing a local function (e.g. a collection of samples from a distribution). Each machine can perform computation on its own function, and then periodically every machine is allowed to transmit information to the others. In order to prove meaningful lower bounds based on the number of rounds of communication, it is necessary to prevent the machines from simply transmitting their local function to a central machine, or else any objective could be optimized in one round. There are two established ways of doing this. First, one can allow arbitrary computation on the local machines, but restrict the number of bits that can be transmitted in each round. There is work focusing on specific statistical estimation problems that establishes communication lower bounds via information-theoretic arguments [7, 11, 28]. Alternatively, one can allow the machines to communicate real-valued vectors, but restrict the types of computation they are allowed to perform. For instance, Arjevani and Shamir [3] present communication complexity lower bounds for algorithms which can only compute vectors that lie in a certain subspace, which includes e.g. linear combinations of gradients of their local function. Lee et al. [15] assume a similar restriction, but allow the data defining the local functions to be allocated to the different machines in a strategic manner. Our framework applies to general stochastic optimization problems and does not impose any restrictions on what computation the algorithm may perform, and is thus a more direct generalization of the oracle model of optimization.

2   The graph-based oracle model

We consider the following stochastic optimization problem

$$\min_{x \in \mathbb{R}^m : \|x\| \leq B} F(x) := \mathbb{E}_{z \sim \mathcal{P}} \left[f(x; z) \right] \tag{1}$$

The problem (1) captures many important tasks, such as supervised learning, in which case $f(x; z)$ is the loss of a model parametrized by x on data instance z and the goal is to minimize the population risk $\mathbb{E}\left[f(x; z) \right]$. We assume that $f(\cdot; z)$ is convex, L-Lipschitz, and H-smooth for all z. We also allow f to be non-smooth, which corresponds to $H = \infty$. A function g is L-Lipschitz when $\|g(x) - g(y)\| \leq L \|x - y\|$ for all x, y, and it is H-smooth when it is differentiable and its gradient is H-Lipschitz. We consider optimization algorithms that use either a stochastic gradient or stochastic prox oracle ($\mathcal{O}_{\text{grad}}$ and $\mathcal{O}_{\text{prox}}$ respectively):

$$\mathcal{O}_{\text{grad}}(x, z) = (f(x; z), \ \nabla f(x; z)) \tag{2}$$

$$\mathcal{O}_{\text{prox}}(x, \beta, z) = \left(f(x; z), \ \nabla f(x; z), \ \text{prox}_{f(\cdot; z)}(x, \beta) \right) \tag{3}$$

$$\text{where} \quad \text{prox}_{f(\cdot; z)}(x, \beta) = \arg \min_y f(y; z) + \frac{\beta}{2} \|y - x\|^2 \tag{4}$$

The prox oracle is quite powerful and provides global rather than local information about f. In particular, querying the prox oracle with $\beta = 0$ fully optimizes $f(\cdot; z)$.

As stated, z is an argument to the oracle, however there are two distinct cases. In the "fully stochastic" oracle setting, the algorithm receives an oracle answer corresponding to a random $z \sim \mathcal{P}$. We also consider a setting in which the algorithm is allowed to "actively query" the oracle. In this case, the algorithm may either sample $z \sim \mathcal{P}$ or choose a desired z and receive an oracle answer for that z. Our lower bounds hold for either type of oracle. Most optimization algorithms only use the fully stochastic oracle, but some require more powerful active queries.

We capture the structure of a parallel optimization algorithm with a directed, acyclic **oracle graph** $\mathcal{G}$. Its depth, D, is the length of the longest directed path, and the size, N, is the number of nodes. Each node in the graph represents a single stochastic oracle access, and the edges in the graph indicate where the results of that oracle access may be used: only the oracle accesses from *ancestors* of each node are available when issuing a new query. These limitations might arise e.g. due to parallel computation delays or the expense of communicating between disparate machines.

Let $\mathcal{Q}$ be the set of possible oracle queries, with the exact form of queries (e.g., $q = x$ vs. $q = (x, \beta, z)$) depending on the context. Formally, a randomized optimization algorithm that accesses the stochastic oracle $\mathcal{O}$ as prescribed by the graph $\mathcal{G}$ is specified by associating with each node v_t a query rule $R_t : (\mathcal{Q}, \mathcal{O}(\mathcal{Q}))^* \times \Xi \to \mathcal{Q}$, plus a single output rule $\hat{X} : (\mathcal{Q}, \mathcal{O}(\mathcal{Q}))^* \times \Xi \to \mathcal{X}$. We grant all of the nodes access to a source of shared randomness $\xi \in \Xi$ (e.g. an infinite stream of random bits). The mapping R_t selects a query q_t to make at node v_t using the set of queries and oracle responses in ancestors of v_t, namely

$$q_t = R_t\big((q_i, \mathcal{O}(q_i) \,:\, i \in \text{Ancestors}(v_t)), \xi\big) \tag{5}$$

Similarly, the output rule $\hat{X}$ maps from all of the queries and oracle responses to the algorithm's output as $\hat{x} = \hat{X}((q_i, \mathcal{O}(q_i) : i \in [N]), \xi)$. The essential question is: for a class of optimization problems $(\mathcal{G}, \mathcal{O}, \mathcal{F})$ specified by a dependency graph $\mathcal{G}$, a stochastic oracle $\mathcal{O}$, and a function class $\mathcal{F}$, what is the best possible guarantee on the expected suboptimality of an algorithm's output, i.e.

$$\inf_{\left(R_1,\ldots,R_N,\hat{X}\right)} \sup_{f \in \mathcal{F}} \mathbb{E}_{\hat{x},z}\left[f(\hat{x}; z)\right] - \min_x \mathbb{E}_z\left[f(x; z)\right] \tag{6}$$

In this paper, we consider optimization problems $(\mathcal{G}, \mathcal{O}, \mathcal{F}_{L,H,B})$ where $\mathcal{F}_{L,H,B}$ is the class of convex, L-Lipschitz, and H-smooth functions on the domain $\{x \in \mathbb{R}^m : \|x\| \leq B\}$ and parametrized by z, and $\mathcal{O}$ is either a stochastic gradient oracle $\mathcal{O}_{\text{grad}}$ (2) or a stochastic prox oracle $\mathcal{O}_{\text{prox}}$ (3). We consider this function class to contain Lipschitz but non-smooth functions too, which corresponds to $H = \infty$. Our function class does not bound the dimension m of the problem, as we seek to understand the best possible guarantees in terms of Lipschitz and smoothness constants that hold in any dimension. Indeed, there are (typically impractical) algorithms such as center-of-mass methods, which might use the dimension in order to significantly reduce the oracle complexity, but at a potentially huge computational cost. Nemirovski [19] studied non-smooth optimization in the case that the dimension is bounded, proving lower bounds in this setting that scale with the $1/3$-power of the dimension but have only logarithmic dependence on the suboptimality. We do not analyze strongly convex functions, but the situation is similar and lower bounds can be established via reduction [27].

3 Lower bounds

We now provide lower bounds for optimization problems $(\mathcal{G}, \mathcal{O}_{\text{grad}}, \mathcal{F}_{L,H,B})$ and $(\mathcal{G}, \mathcal{O}_{\text{prox}}, \mathcal{F}_{L,H,B})$ in terms of L, H, B, and the depth and size of $\mathcal{G}$.

Theorem 1. *Let $L, B \in (0, \infty)$, $H \in [0, \infty]$, $N \geq D \geq 1$, let $\mathcal{G}$ be any oracle graph of depth D and size N and consider the optimization problem $(\mathcal{G}, \mathcal{O}_{\text{grad}}, \mathcal{F}_{L,H,B})$. For any randomized algorithm $\mathcal{A} = (R_1, \ldots, R_N, \hat{X})$, there exists a distribution $\mathcal{P}$ and a convex, L-Lipschitz, and H-smooth function f on a B-bounded domain in $\mathbb{R}^m$ for $m = O\left(\max\left\{N^2, D^3 N\right\} \log(DN)\right)$ such that*

$$\mathbb{E}_{\substack{z \sim \mathcal{P} \\ \hat{X} \sim \mathcal{A}}} \left[f(\hat{X}; z)\right] - \min_x \mathbb{E}_{z \sim \mathcal{P}}\left[f(x; z)\right] \geq \Omega\left(\min\left\{\frac{LB}{\sqrt{D}}, \frac{HB^2}{D^2}\right\} + \frac{LB}{\sqrt{N}}\right)$$

Theorem 2. *Let $L, B \in (0, \infty)$, $H \in [0, \infty]$, $N \geq D \geq 1$, let $\mathcal{G}$ be any oracle graph of depth D and size N and consider the optimization problem $(\mathcal{G}, \mathcal{O}_{\text{prox}}, \mathcal{F}_{L,H,B})$. For any randomized algorithm $\mathcal{A} = (R_1, \ldots, R_N, \hat{X})$, there exists a distribution $\mathcal{P}$ and a convex, L-Lipschitz, and H-smooth function f on a B-bounded domain in $\mathbb{R}^m$ for $m = O\left(\max\left\{N^2, D^3 N\right\} \log(DN)\right)$ such that*

$$\mathbb{E}_{\substack{z \sim \mathcal{P} \\ \hat{X} \sim \mathcal{A}}} \left[f(\hat{X}; z)\right] - \min_x \mathbb{E}_{z \sim \mathcal{P}}\left[f(x; z)\right] \geq \Omega\left(\min\left\{\frac{LB}{D}, \frac{HB^2}{D^2}\right\} + \frac{LB}{\sqrt{N}}\right)$$

These are the tightest possible lower bounds in terms of just the depth and size of $\mathcal{G}$ in the sense that for all D, N there are graphs $\mathcal{G}$ and associated algorithms which match the lower bound. Of course, for specific, mostly degenerate graphs they might not be tight. For instance, our lower bound for the graph consisting of a short sequential chain plus a very large number of disconnected nodes might be quite loose due to the artificial inflation of N. Nevertheless, for many interesting graphs they are tight, as we shall see in Section 4.

Each lower bound has two components: an "optimization" term and a "statistical" term. The statistical term $\Omega(LB/\sqrt{N})$ is well known, although we include a brief proof of this portion of the bound

in Appendix D for completeness. The optimization term depends on the depth D, and indicates, intuitively, the best suboptimality guarantee that can be achieved by an algorithm using unlimited parallelism but only D rounds of communication. Arjevani and Shamir [3] also obtain lower bounds in terms of rounds of communication, which are similar to how our lower bounds depend on depth. However they restricted the type of computations that are allowed to the algorithm to a specific class of operations, while we only limit the number of oracle queries and the dependency structure between them, but allow forming the queries in any arbitrary way.

Similar to Arjevani and Shamir [3], to establish the optimization term in the lower bounds, we construct functions that require multiple rounds of sequential oracle accesses to optimize. In the gradient oracle case, we use a single, deterministic function which resembles a standard construction for first order optimization lower bounds. For the prox case, we construct two functions inspired by previous lower bounds for round-based and finite sum optimization [3, 27]. In order to account for randomized algorithms that might leave the span of gradients or proxs returned by the oracle, we use a technique that was proposed by Woodworth and Srebro [26, 27] and refined by Carmon et al. [8]. For our specific setting, we require a slight modification of the previous analysis, which is detailed in Appendix A.
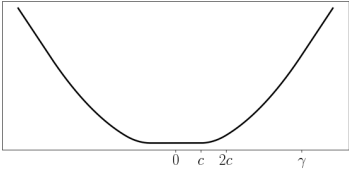
A useful feature of our lower bounds is that they apply when both the Lipschitz constant and smoothness are bounded *concurrently*. Consequently, "non-smooth" in the subsequent discussion can be read as simply identifying the case where the L term achieves the minimum as opposed to the H term (even if $H < \infty$). This is particularly important when studying stochastic parallel optimization, since obtaining non-trivial guarantees in a purely stochastic setting requires some sort of control on the magnitude of the gradients (smoothness by itself is not sufficient), while obtaining parallelization speedups often requires smoothness, and so we would like to ask what is the best that can be done when both Lipschitz and smoothness are controlled. Interestingly, the dependence on both L and H in our bounds is tight, even when the other is constrained, which shows that the optimization term cannot be substantially reduced by using both conditions together.

In the case of the gradient oracle, we "smooth out" a standard non-smooth lower bound construction [20, 26]; previous work has used a similar approach in slightly different settings [2, 12]. For $\ell \leq L$ and $\eta \leq H$, and orthonormal $v_1, \ldots, v_{D+1}$ drawn uniformly at random, we define the ℓ-Lipschitz but non-smooth function $\tilde{f}$, and its ℓ-Lipschitz, η-smooth "η-Moreau envelope" [5]:

$$\tilde{f}(x) = \max_{1 \leq r \leq D+1} \ell \left(v_r^\top x - \frac{r-1}{2(D+1)^{1.5}} \right) \qquad f(x) = \min_y \tilde{f}(y) + \frac{\eta}{2} \|y - x\|^2 \quad (7)$$

This defines a distribution over f's based on the randomness in the draw of $v_1, \ldots, v_{D+1}$, and we apply Yao's minimax principle. In Appendix B, we prove Theorem 1 using this construction.

In the case of the prox oracle, we "straighten out" the smooth construction of Woodworth and Srebro [27]. For fixed constants c, γ, we define the following Lipschitz and smooth scalar function ϕ_c:

$$\phi_c(z) = \begin{cases} 0 & |z| \leq c \\ 2(|z| - c)^2 & c < |z| \leq 2c \\ z^2 - 2c^2 & 2c < |z| \leq \gamma \\ 2\gamma |z| - \gamma^2 - 2c^2 & |z| > \gamma \end{cases} \quad (8)$$

For $\mathcal{P} = \text{Uniform}\{1, 2\}$ and orthonormal $v_1, \ldots, v_{2D}$ drawn uniformly at random, we define

$$f(x; 1) = \frac{\eta}{8} \left(-2a v_1^\top x + \phi_c \left(v_{2D}^\top x \right) + \sum_{r=3,5,7,\ldots}^{2D-1} \phi_c \left(v_{r-1}^\top x - v_r^\top x \right) \right) \quad (9)$$

$$f(x; 2) = \frac{\eta}{8} \left(\sum_{r=2,4,6,\ldots}^{2D} \phi_c \left(v_{r-1}^\top x - v_r^\top x \right) \right) \quad (10)$$

Again, this defines a distribution over f's based on the randomness in the draw of $v_1, \ldots, v_{2D}$ and we apply Yao's minimax principle. In Appendix C, we prove Theorem 2 using this construction.

Graph example	With gradient oracle	With gradient and prox oracle
path(T) (Section 4.1)	$\dfrac{L}{\sqrt{T}}$	
layer(T, M) (Section 4.2)	$\left(\dfrac{L}{\sqrt{T}} \wedge \dfrac{H}{T^2}\right) + \dfrac{L}{\sqrt{MT}}$	$\left(\dfrac{L}{T} \wedge \dfrac{H}{T^2}\right) + \dfrac{L}{\sqrt{MT}}$
delay(T, τ) (Section 4.3)	$\left(\dfrac{L}{\sqrt{T/\tau}} \wedge \dfrac{H\tau^2}{T^2}\right) + \dfrac{L}{\sqrt{T}}$	$\left(\dfrac{L\tau}{T} \wedge \dfrac{H\tau^2}{T^2}\right) + \dfrac{L}{\sqrt{T}}$
intermittent(T, K, M) (Section 4.4)	$\left(\dfrac{L}{\sqrt{KT}} \wedge \dfrac{H}{K^2T^2}\right) + \dfrac{L}{\sqrt{MKT}}$ $\dfrac{L}{\sqrt{KT}} \wedge \left(\dfrac{H}{T^2} + \dfrac{L}{\sqrt{MKT}}\right)$ $\wedge \left(\dfrac{H}{TK} + \dfrac{L}{\sqrt{MKT}}\right) \log\left(\dfrac{MKT}{L}\right)$	$\left(\dfrac{L}{KT} \wedge \dfrac{H}{K^2T^2}\right) + \dfrac{L}{\sqrt{MKT}}$ $\dfrac{L}{\sqrt{KT}} \wedge \left(\left(\dfrac{L}{T} \wedge \dfrac{H}{T^2}\right) + \dfrac{L}{\sqrt{MKT}}\right)$ $\wedge \left(\dfrac{H}{TK} + \dfrac{L}{\sqrt{MKT}}\right) \log\left(\dfrac{MKT}{L}\right)$

Table 1: Summary of upper and lower bounds for stochastic convex optimization of L-Lipschitz and H-smooth functions with T iterations, M machines, and K sequential steps per machine. Green indicates lower bounds matched only by "unnatural" methods, red and blue indicates a gap between the lower and upper bounds.

4 Specific dependency graphs

We now use our framework to study four specific parallelization structures. The main results (tight complexities and gaps between lower and upper bounds) are summarized in Table 1. For simplicity and without loss of generality, we set $B = 1$, i.e. we normalize the optimization domain to be $\{x \in \mathbb{R}^m : \|x\| \le 1\}$. All stated upper and lower bounds are for the expected suboptimality $\mathbb{E}[F(\hat{x})] - F(x^*)$ of the algorithm's output.

4.1 Sequential computation: the path graph
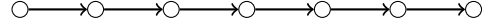
We begin with the simplest model, that of sequential computation captured by the path graph of length T depicted above. The ancestors of each vertex v_i, $i = 1 \ldots T$ are all the preceding vertices $(v_1, \ldots, v_{i-1})$. The sequential model is of course well studied and understood. To see how it fits into our framework: A path graph of length T has a depth of $D = T$ and size of $N = T$, thus with either gradient or prox oracles, the statistical term is dominant in Theorems 1 and 2. These lower bounds are matched by sequential stochastic gradient descent, yielding a tight complexity of $\Theta(L/\sqrt{T})$ and the familiar conclusion that SGD is (worst case) optimal in this setting.

4.2 Simple parallelism: the layer graph
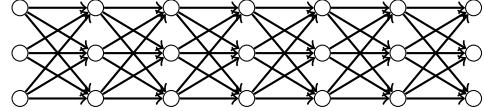
We now turn to a model in which M oracle queries can be made in parallel, and the results are broadcast for use in making the next batch of M queries. This corresponds to synchronized parallelism and fast communication between processors. The model is captured by a layer graph of width M, depicted above for $M = 3$. The graph consists of T layers $i = 1, \ldots, T$ each with M nodes $v_{t,1}, \ldots, v_{t,m}$ whose ancestors include $v_{t',i}$ for all $t' < t$ and $i \in [M]$. The graph has a depth of $D = T$ and size of $N = MT$. With a stochastic gradient oracle, Theorem 1 yields a lower bound of:

$$\Omega\left(\min\left\{\frac{L}{\sqrt{T}}, \frac{H}{T^2}\right\} + \frac{L}{\sqrt{MT}}\right) \tag{11}$$

which is matched by accelerated mini-batch SGD (A-MB-SGD) [9, 14], establishing the optimality of A-MB-SGD in this setting. For sufficiently smooth objectives, the same algorithm is also optimal even if prox access is allowed, since Theorem 2 implies a lower bound of:

$$\Omega\left(\min\left\{\frac{L}{T}, \frac{H}{T^2}\right\} + \frac{L}{\sqrt{MT}}\right). \tag{12}$$

That is, for smooth objectives, having access to a prox oracle does not improve the optimal complexity over just using gradient access. However, for non-smooth or insufficiently smooth objectives, there is a gap between (11) and (12). An optimal algorithm, smoothed A-MB-SGD, uses the prox oracle in order to calculate gradients of the Moreau envelope of $f(x; z)$ (cf. Proposition 12.29 of [5]), and then performs A-MB-SGD on the smoothed objectives. This yields a suboptimality guarantee that precisely matches (12), establishing that the lower bound from Theorem 2 is tight for the layer graph,

and that smoothed A-MB-SGD is optimal. An analysis of the smoothed A-MB-SGD algorithm is provided in Appendix E.1.

4.3 Delayed updates
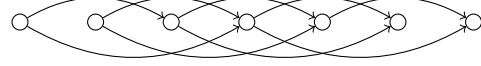
We now turn to a delayed computation model that is typical in many asynchronous parallelization and pipelined computation settings, e.g. when multiple processors or machines are working asynchronously, reading iterates, taking some time to perform the oracle accesses and computation, then communicating the results back (or updating the iterate accordingly) [1, 6, 16, 18, 24]. This is captured by a "delay graph" with T nodes $v_1, \ldots, v_T$ and delays τ_t for the response to the oracle query performed at v_t to become available. Hence, $\text{Ancestors}(v_t) = \{v_s \mid s + \tau_s \leq t\}$. Analysis is typically based on the delays being bounded, i.e. $\tau_t \leq \tau$ for all t. The depiction above corresponds to $\tau_t = 2$; the case $\tau_t = 1$ corresponds to the path graph. With constant delays $\tau_t = \tau$, the delay graph has depth $D \leq T/\tau$ and size $N = T$, so Theorem 1 gives the following lower bound when using a gradient oracle:

$$\Omega\left(\min\left\{ \frac{L}{\sqrt{T/\tau}}, \frac{H}{(T/\tau)^2} \right\} + \frac{L}{\sqrt{T}} \right). \tag{13}$$

Delayed SGD, with updates $x_t \leftarrow x_{t-1} - \eta_t \nabla f(x_{t-\tau_t}; z)$, is a natural algorithm in this setting. Under the bounded delay assumption the best guarantee we are aware of for delayed update SGD is (see [10] improving over [1])

$$O\left(\frac{H}{T/\tau^2} + \frac{L}{\sqrt{T}} \right). \tag{14}$$

This result is significantly worse than the lower bound (13) and quite disappointing. It does not provide for a $1/T^2$ accelerated optimization rate, but even worse, compared to non-accelerated SGD it suffers a slowdown quadratic in the delay, compared to the linear slowdown we would expect. In particular, the guarantee (14) only allows maximum delay of $\tau = O(T^{1/4})$ in order to attain the optimal statistical rate $\Theta(L/\sqrt{T})$, whereas the lower bound allows a delay up to $\tau = O(T^{3/4})$.

This raises the question of whether a different algorithm can match the lower bound (13). The answer is affirmative, but it requires using an "unnatural" algorithm, which simulates a mini-batch approach in what seems an unnecessarily wasteful way. We refer to this as a "wait-and-collect" approach: it works in $T/(2\tau)$ stages, each stage consisting of 2τ iterations (i.e. nodes or oracle accesses). In stage i, τ iterations are used to obtain τ stochastic gradient estimates $\nabla f(x_i; z_{2\tau i+j})$, $j = 1, \ldots, \tau$ at the same point x_i. For the remaining τ iterations, we wait for all the preceding oracle computations to become available and do not even use our allowed oracle access. We can then finally update the x_{i+1} using the minibatch of τ gradient estimates. This approach is also specified formally as Algorithm 2 in Appendix E.2. Using this approach, we can perform $T/(2\tau)$ A-MB-SGD updates with a minibatch size of τ, yielding a suboptimality guarantee that precisely matches the lower bound (13).

Thus (13) indeed represents the tight complexity of the delay graph with a stochastic gradient oracle, and the wait-and-collect approach is optimal. However, this answer is somewhat disappointing and leaves an intriguing open question: can a more natural, and seemingly more efficient (no wasted oracle accesses) delayed update SGD algorithm also match the lower bound? An answer to this question has two parts: first, does the delayed update SGD truly suffer from a τ^2 slowdown as indicated by (14), or does it achieve linear degradation and a speculative guarantee of

$$O\left(\frac{H}{T/\tau} + \frac{L}{\sqrt{T}} \right). \tag{15}$$

Second, can delayed update SGD be accelerated to achieve the optimal rate (13). We note that concurrent with our work there has been progress toward closing this gap: Arjevani et al. [4] showed an improved bound matching the non-accelerated (15) for delayed updates (with a fixed delay) on quadratic objectives. It still remains to generalize the result to smooth non-quadratic objectives, handle non-constant bounded delays, and accelerate the procedure so as to improve the rate to $(\tau/T)^2$.

4.4 Intermittent communication
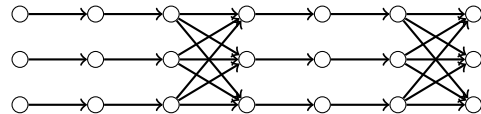
We now turn to a parallel computation model which is relevant especially when parallelizing across disparate machines: in each of T iterations, there are M machines that, instead of just a single oracle access, perform K sequential oracle accesses before broadcasting to all other machines synchronously. This communication pattern is relevant in the realistic scenario where local computation is plentiful relative to communication costs (i.e. K is large). This may be the case with fast processors distributed across different machines, or in the setting of federated learning, where mobile devices collaborate to train a shared model while keeping their respective training datasets local [17].

This is captured by a graph consisting of M parallel chains of length TK, with cross connections between the chains every K nodes. Indexing the nodes as $v_{t,m,k}$, the nodes $v_{t,m,1} \to \cdots \to v_{t,m,K}$ form a chain, and $v_{t,m,K}$ is connected to $v_{t+1,m',1}$ for all $m' = 1..M$. This graph generalizes the layer graph by allowing K sequential oracle queries between each complete synchronization; $K = 1$ recovers the layer graph, and the depiction above corresponds to $K = M = 3$. We refer to the computation between each synchronization step as a (communication) round.

The depth of this graph is $D = TK$ and the size is $N = TKM$. Focusing on the stochastic gradient oracle (the situation is similar for the prox oracle, except with the potential of smoothing a non-smooth objective, as discussed in Section 4.2), Theorem 1 yields the lower bound:

$$\Omega\left(\min\left\{\frac{L}{\sqrt{TK}}, \frac{H}{T^2K^2}\right\} + \frac{L}{\sqrt{TKM}}\right). \tag{16}$$

A natural algorithm for this graph is parallel SGD, where we run an SGD chain on each machine and average iterates during communication rounds, e.g. [17]. The updates are then given by:

$$x_{t,m,0} = \frac{1}{M}\sum_{m'} x_{t,m',K} \tag{17}$$

$$x_{t,m,k} = x_{t,m,k-1} - \eta_t \nabla f(x_{t,m,k-1}; z_{t,m,k}), \ k = 1, \ldots, K$$

(note that $x_{t,m,0}$ does not correspond to any node in the graph, and is included for convenience of presentation). Unfortunately, we are not aware of any satisfying analysis of such a parallel SGD approach. Instead, we consider two other algorithms in an attempt to match the lower bound (16). First, we can combine all KM oracle accesses between communication rounds in order to form a single mini-batch, giving up on the possibility of sequential computation along the "local" K node sub-paths. Using all KM nodes to obtain stochastic gradient estimates at the same point, we can perform T iterations of A-MB-SGD with a mini-batch size of KM, yielding an upper bound of

$$O\left(\frac{H}{T^2} + \frac{L}{\sqrt{TKM}}\right). \tag{18}$$

This is a reasonable and common approach, and it is optimal (up to constant factors) when $KM = O(\frac{L^2}{H^2}T^3)$ so that the statistical term is limiting. However, comparing (18) to the lower bound (16) we see a gap by a factor of K^2 in the optimization term, indicating the possibility for significant gains when K is large (i.e. when we can process a large number of examples on each machine at each round). Improving the optimization term by this K^2 factor would allow statistical optimality as long as $M = O(T^3K^3)$—-this is a very significant difference. In many scenarios we would expect a modest number of machines, but the amount of data on each machine could easily be much more than the number of communication rounds, especially if communication is across a wide area network.

In fact, when K is large, a different approach is preferable: we can ignore all but a single chain and simply execute KT iterations of sequential SGD, offering an upper bound of

$$O\left(\frac{L}{\sqrt{TK}}\right). \tag{19}$$

Although this approach seems extremely wasteful, it actually yields a better guarantee than (18) when $K \geq \Omega(T^3L^2/H)$. This is a realistic regime, e.g. in federated learning when computation is distributed across devices, communication is limited and sporadic and so only a relatively small number of rounds T are possible, but each device already possesses a large amount of data. Furthermore, for non-smooth functions, (19) matches the lower bound (16).

Our upper bound on the complexity is therefore obtained by selecting either A-MB-SGD or single-machine sequential SGD, yielding a combined upper bound of

$$O\left(\min\left\{\frac{L}{\sqrt{TK}}, \frac{H}{T^2}\right\} + \frac{L}{\sqrt{TKM}}.\right) \tag{20}$$

For smooth functions, there is still a significant gap between this upper bound and the lower bound (16). Furthermore, this upper bound is not achieved by a single algorithm, but rather a combination of two separate algorithms, covering two different regimes. This raises the question of whether there is a single, natural algorithm, perhaps an accelerated variant of the parallel SGD updates (17), that at the very least matches (20), and preferably also improves over them in the intermediate regime or even matches the lower bound (16).

Active querying and SVRG All methods discussed so far used fully stochastic oracles, requesting a gradient (or prox computation) with respect to an independently and randomly drawn $z \sim \mathcal{P}$. We now turn to methods that also make active queries, i.e. draw samples from $\mathcal{P}$ and then repeatedly query the oracle, at different points x, but on the same samples z. Recall that all of our lower bounds are valid also in this setting.

With an active query gradient oracle, we can implement SVRG [13, 15] on an intermittent communication graph. More specifically, for an appropriate choice of n and λ, we apply SVRG to the regularized empirical objective:

$$\hat{F}_\lambda(x) = \frac{1}{n} \sum_{i=1}^{n} f(x; z_i) + \frac{\lambda}{2} \|x\|^2 \tag{21}$$

Algorithm 1 SVRG

Parameters: n, S, I
Sample $z_1, \ldots, z_n \sim \mathcal{P}$
Initialize $x_0 = 0$
for $s = 1, 2, \ldots, S = \left\lfloor \frac{T}{\lceil \frac{n}{KM} \rceil + \lceil \frac{I}{K} \rceil} \right\rfloor$ **do**
  $\tilde{x} = x_{s-1}$
  $\tilde{g} = \nabla \hat{F}_\lambda(\tilde{x}) = \frac{1}{n} \sum_{i=1}^{n} \nabla f(\tilde{x}; z_i) + \lambda \tilde{x}$ $(*)$
  $x_s^0 = \tilde{x}$
  for $i = 1, 2, \ldots, I = \frac{H}{\lambda}$ **do**
    Sample $j \sim \text{Uniform} \{1, \ldots, n\}$
    $x_s^i = x_s^{i-1} - \eta \left(\left(\nabla f(x_s^{i-1}; z_j) + \lambda x_s^{i-1} \right) - \left(\nabla f(\tilde{x}; z_j) + \lambda \tilde{x} \right) + \tilde{g} \right)$ $(**)$
  end for
  $x_s = x_s^i$ for $i \sim \text{Uniform} \{1, \ldots, I\}$
end for
Return x_S

To do so, we first pick a sample $\{z_1, \ldots z_n\}$ (without actually querying the oracle). As indicated by Algorithm 1, we then alternate between computing full gradients on $\{z_1, \ldots z_n\}$ in parallel $(*)$, and sequential variance-reduced stochastic gradient updates in between $(**)$. The full gradient $\tilde{g}$ is computed using n active queries to the gradient oracle. Since all of these oracle accesses are made at the same point $\tilde{x}$, this can be fully parallelized across the M parallel chains of length K thus requiring n/KM rounds. The sequential variance-reduced stochastic gradient updates *cannot* be parallelized in this way, and must be performed using queries to the gradient oracle in just one of the M available parallel chains, requiring I/K rounds of synchronization. Consequently, each outer iteration of SVRG requires $\lceil \frac{n}{KM} \rceil + \lceil \frac{I}{K} \rceil$ rounds. We analyze this method using $\lambda = \Theta \left(\frac{L}{\sqrt{n}} \right)$, $I = \Theta \left(\frac{H}{\lambda} \right) = \Theta \left(\frac{H\sqrt{n}}{L} \right)$, and $n = \min \left\{ \Theta \left(\frac{K^2 T^2 L^2}{H^2 \log^2(MKT/L)} \right), \Theta \left(\frac{MKT}{\log(MKT/L)} \right) \right\}$. Using the analysis of Johnson and Zhang [13], SVRG guarantees that, with an appropriate stepsize, we have $\hat{F}_\lambda(x_S) - \min_x \hat{F}_\lambda(x) \le 2^{-S}$; the value of x_S on the empirical objective also generalizes to the population, so $\mathbb{E}\left[f(x_S; z) \right] - \min_x \mathbb{E}\left[f(x; z) \right] \le 2^{-S} + O\left(\frac{L}{\sqrt{n}} \right)$ (see [22]). With our choice of parameters, this implies upper bound (see Appendix E.3)

$$O\left(\left(\frac{H}{TK} + \frac{L}{\sqrt{TKM}} \right) \log \left(\frac{TKM}{L} \right) \right). \tag{22}$$

These guarantees improve over sequential SGD (17) as soon as $M > \log^2(TKM/L)$ and $K > H^2/L^2$, i.e. $L/\sqrt{TK} < L^2/H$. This is a very wide regime: we require only a moderate number

8

of machines, and the second condition will typically hold for a smooth loss. Intuitively, SVRG does roughly the same number (up to a factor of two) of sequential updates as in the sequential SGD approach but it uses better, variance reduced updates. The price we pay is in the smaller total sample size since we keep calling the oracle on the same samples. Nevertheless, since SVRG only needs to calculate the "batch" gradient a logarithmic number of times, this incurs only an additional logarithmic factor.

Comparing (18) and (22), we see that SVRG also improves over A-MB-SGD as soon as $K > T \log(TKM/L)$, that is if the number of points we are processing on each machine each round is slightly more then the total number of rounds, which is also a realistic scenario.

To summarize, the best known upper bound for optimizing with intermittent communication using a pure stochastic oracle is (20), which combines two different algorithms. However, with active oracle accesses, SVRG is also possible and the upper bound becomes:

$$O \left(\min \left\{ \frac{L}{\sqrt{TK}}, \left(\frac{H}{TK} + \frac{L}{\sqrt{TKM}} \right) \log \left(\frac{TKM}{L} \right), \frac{H}{T^2} + \frac{L}{\sqrt{TKM}} \right\} \right) \qquad (23)$$

5 Summary

Our main contributions in this paper are: (1) presenting a precise formal oracle framework for studying parallel stochastic optimization; (2) establishing tight oracle lower bounds in this framework that can then be easily applied to particular instances of parallel optimization; and (3) using the framework to study specific settings, obtaining optimality guarantees, understanding where additional assumptions would be needed to break barriers, and, perhaps most importantly, identifying gaps in our understanding that highlight possibilities for algorithmic improvement.

Specifically, we highlight the following gaps:

- For non-smooth objectives and a stochastic prox oracle, it is possible to improve performance in the layer graph setting using smoothing and acceleration. It is not clear whether one can match the same optimal performance using a more direct algorithm, e.g. one that averages the answers from the prox oracle.

- In the delay graph setting, delayed update SGD's guarantee is not optimal. We suggest an alternative optimal algorithm, but it would be interesting and beneficial to understand the true behavior of delayed update SGD and to improve it as necessary to attain optimality.

- With intermittent communication, we show how different methods are better in different regimes, but even combining these methods does not match our lower bound. This sets the stage for studying the optimal complexity in this setting: is our lower bound achievable? Are current methods optimal? Is the true optimal complexity somewhere in between? Even just obtaining a single method that matches the current best performance in all regimes would be a significant advance here.

- With intermittent communication, active queries allow us to obtain better performance in a certain regime. Can we match this performance using pure stochastic queries or is there a real gap between active and pure stochastic queries?

The investigation into optimizing over $\mathcal{F}_{L,H,B}$ in our framework indicates that there is no advantage to the prox oracle for optimizing (sufficiently) smooth functions. This raises the question of what additional assumptions might allow us to leverage the prox oracle, which is intuitively much stronger as it allows global access to $f(\cdot; z)$. One option is to assume a bound on the variance of the stochastic oracle i.e. $\mathbb{E}_z[\|\nabla f(x; z) - \nabla F(x)\|^2] \leq \sigma^2$ which captures the notion that the functions $f(\cdot; z)$ are somehow related and not arbitrarily different. In particular, if each stochastic oracle access, in each node, is based on a sample of b data points (thus, a prox operation optimizes a sub-problem of size b), we have that $\sigma^2 \leq L^2/b$. Initial investigation into the complexity of optimizing over the restricted class $\mathcal{F}_{L,H,B,\sigma}$ (where we also require the above variance bound), reveals a significant theoretical advantage for the prox oracle over the gradient oracle, even for smooth functions. This is an example of how formalizing the optimization problem gives insight into additional assumptions, in this case low variance, that are necessary for realizing the benefits of a stronger oracle.

References

[1] Alekh Agarwal and John C Duchi. Distributed delayed stochastic optimization. In *Advances in Neural Information Processing Systems*, pages 873–881, 2011.

[2] Naman Agarwal and Elad Hazan. Lower bounds for higher-order convex optimization. *arXiv preprint arXiv:1710.10329*, 2017.

[3] Yossi Arjevani and Ohad Shamir. Communication complexity of distributed convex learning and optimization. In *Advances in Neural Information Processing Systems*, pages 1756–1764, 2015.

[4] Yossi Arjevani, Ohad Shamir, and Nathan Srebro. A tight convergence analysis for stochastic gradient descent with delayed updates. 2018.

[5] Heinz H Bauschke, Patrick L Combettes, et al. *Convex analysis and monotone operator theory in Hilbert spaces*, volume 2011. Springer, 2017.

[6] Dimitri P Bertsekas. *Parallel and distributed computation: numerical methods*, volume 23. Prentice hall Englewood Cliffs, NJ, 1989.

[7] 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. ACM, 2016.

[8] Yair Carmon, John C Duchi, Oliver Hinder, and Aaron Sidford. Lower bounds for finding stationary points i. *arXiv preprint arXiv:1710.11606*, 2017.

[9] Andrew Cotter, Ohad Shamir, Nati Srebro, and Karthik Sridharan. Better mini-batch algorithms via accelerated gradient methods. In *Advances in neural information processing systems*, pages 1647–1655, 2011.

[10] Hamid Reza Feyzmahdavian, Arda Aytekin, and Mikael Johansson. An asynchronous mini-batch algorithm for regularized stochastic optimization. *IEEE Transactions on Automatic Control*, 61(12):3740–3754, 2016.

[11] Ankit Garg, Tengyu Ma, and Huy Nguyen. On communication cost of distributed statistical estimation and dimensionality. In *Advances in Neural Information Processing Systems*, pages 2726–2734, 2014.

[12] Cristóbal Guzmán and Arkadi Nemirovski. On lower complexity bounds for large-scale smooth convex optimization. *Journal of Complexity*, 31(1):1–14, 2015.

[13] Rie Johnson and Tong Zhang. Accelerating stochastic gradient descent using predictive variance reduction. In *Advances in Neural Information Processing Systems*, 2013.

[14] Guanghui Lan. An optimal method for stochastic composite optimization. *Mathematical Programming*, 133(1-2):365–397, 2012.

[15] Jason D Lee, Qihang Lin, Tengyu Ma, and Tianbao Yang. Distributed stochastic variance reduced gradient methods by sampling extra data with replacement. *The Journal of Machine Learning Research*, 18(1):4404–4446, 2017.

[16] Brendan McMahan and Matthew Streeter. Delay-tolerant algorithms for asynchronous distributed online learning. In *Advances in Neural Information Processing Systems*, pages 2915–2923, 2014.

[17] H. Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial Intelligence and Statistics*, 2017.

[18] A Nedić, Dimitri P Bertsekas, and Vivek S Borkar. Distributed asynchronous incremental subgradient methods. *Studies in Computational Mathematics*, 8(C):381–407, 2001.

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

[20] Arkadii Nemirovski, David Borisovich Yudin, and Edgar Ronald Dawson. Problem complexity and method efficiency in optimization. 1983.

[21] Yurii Nesterov. A method of solving a convex programming problem with convergence rate o (1/k2). 1983.

[22] Shai Shalev-Shwartz and Nathan Srebro. Svm optimization: inverse dependence on training set size. In *International Conference on Machine Learning*, pages 928–935, 2008.

[23] Eric V Slud et al. Distribution inequalities for the binomial law. *The Annals of Probability*, 5 (3):404–412, 1977.

[24] Suvrit Sra, Adams Wei Yu, Mu Li, and Alex Smola. Adadelay: Delay adaptive distributed stochastic optimization. In *Artificial Intelligence and Statistics*, pages 957–965, 2016.

[25] Jialei Wang, Weiran Wang, and Nathan Srebro. Memory and communication efficient distributed stochastic optimization with minibatch prox. In *Conference on Learning Theory*, 2017.

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

[27] Blake Woodworth and Nati Srebro. Tight complexity bounds for optimizing composite objectives. In *Advances in Neural Information Processing Systems*, pages 3639–3647, 2016.

[28] Yuchen Zhang, John Duchi, Michael I Jordan, and Martin J Wainwright. Information-theoretic lower bounds for distributed statistical estimation with communication constraints. In *Advances in Neural Information Processing Systems*, pages 2328–2336, 2013.

A Main lower bound lemma

This analysis closely follows that of previous work, specifically the proof of Theorem 1 in [26] and the proof of Lemma 4 in [8]. There are slight differences in the problem setup between this work and that of previous papers, thus we include the following analysis for completeness and to ensure that all of our results can be verified. We do not claim any significant technical novelty within this section.

Let $V = \{v_1, \ldots, v_k\}$ be a uniformly random orthonormal set of vectors in $\mathbb{R}^m$. All of the probabilities referred to in Appendix A will be over the randomness in the selection of V. Let $X = \{x_1, x_2, \ldots, x_N\}$ be a set of vectors in $\mathbb{R}^m$ where $\|x_i\| \leq 1$ for all $i \leq N$. Let these vectors be organized into disjoint subsets $X_1 \cup X_2 \cup \cdots \cup X_k = X$. Furthermore, suppose that for each $t \leq k$, the set X_t is a deterministic function $X_t = X_t(X_{<t}, V)$, so it can also be expressed as $X_t = X_t(V)$.

Let $S_t = X_{\leq t} \cup V_{\leq t}$, let P_t be the projection operator onto the span of S_t and let $P_t^\perp$ be the projection onto the orthogonal complement of the span of S_t. As in [8, 26], define

$$G_t = G_t(V) = \left[\!\!\left[\forall x \in X_t \; \forall j \geq t \; \left| \left\langle \frac{P_{t-1}^\perp x}{\|P_{t-1}^\perp x\|}, \, v_j \right\rangle \right| \leq \alpha \right]\!\!\right] \tag{24}$$

Finally, suppose that for each t, X_t is of the form:

$$X_t(V) = X_t\left(V_{<t}\mathbb{1}_{G_{<t}} + V\mathbb{1}_{\neg G_{<t}}\right) \tag{25}$$

i.e. conditioned on the event $G_{<t}$, it is a deterministic function of $V_{<t}$ only (and not $v_t, \ldots, v_k$). We say that $\mathbb{P}[G_{<1}] = 1$, so X_1 is always independent of V.

First, we connect the events G_t to a more immediately useful condition

Lemma 1. *[cf. Lemma 9 [8], Lemma 1 [26]] For any c, k, N, V, and $\{X_k\}_{t=1}^k$, let $\alpha = \min\left\{\frac{1}{4N}, \frac{c}{2(1+\sqrt{2N})}\right\}$ then for each $t \leq k$*

$$G_{\leq t} \implies G'_{\leq t} := \left[\!\!\left[\forall r \leq t, \; \forall x \in X_r, \; \forall j \geq t \; |\langle x, v_j \rangle| \leq \frac{c}{2} \right]\!\!\right]$$

The proof of Lemma 1 involves straightforward linear algebra, and we defer it to Appendix A.1. By Lemma 1, $G_{<t} \subseteq G'_{<t}$, therefore the property (25) is implied by

$$X_t(V) = X_t\left(V_{<t}\mathbb{1}_{G'_{<t}} + V\mathbb{1}_{\neg G'_{<t}}\right) \tag{26}$$

Now, we state the main result which allows us to prove our lower bounds:

Lemma 2. *[cf. Lemma 4 [8], Lemma 4 [26]] For any $k \geq 1$, $N \geq 1$, $c \in (0,1)$, and dimension*

$$m \geq k + N + \max\left\{32N^2, \frac{8(1+\sqrt{2N})^2}{c^2}\right\} \log\left(2k^2N\right)$$

if the sets $X_1, \ldots, X_k$ satisfy the property (26) then

$$\mathbb{P}\left[\forall t \leq k \; \forall x \in X_t \; \forall j \geq t \; \; |\langle x, v_j \rangle| \leq \frac{c}{2}\right] \geq \frac{1}{2}$$

The proof of Lemma 2 relies upon the following, whose proof we defer to Appendix A.1.

Lemma 3. *[cf. Lemma 11 [8], Lemma 3 [26]] Let R be any rotation operator, $R^\top R = I$, that preserves S_{t-1}, that is $Rw = R^\top w = w$ for any $w \in Span(S_{t-1})$. Then the following conditional densities are equal*

$$p\left(V_{\geq t} \mid G_{<t}, V_{<t}\right) = p\left(RV_{\geq t} \mid G_{<t}, V_{<t}\right)$$

Proof of Lemma 2. This closely follows the proof of Lemma 4 [8] and Lemma 4 [26], with small modifications to account for the different setting.

Set $\alpha = \min\left\{\frac{1}{4N}, \frac{c}{2(1+\sqrt{2N})}\right\}$. Then by Lemma 1, since $X_1, \ldots, X_k$ satisfy the property (26)

$$\mathbb{P}\left[\forall t \leq k \; \forall x \in X_t \; \forall j \geq t \; \; |\langle x, v_j \rangle| \leq \frac{c}{2}\right] \geq \mathbb{P}[G_{\leq k}] = \prod_{t \leq k} \mathbb{P}[G_t \mid G_{<t}] \tag{27}$$

12

Focus on a single term in this product,

$$\mathbb{P}\left[G_t \mid G_{<t}\right] = \mathbb{E}_{V_{<t}}\left[\mathbb{P}\left[G_t \mid G_{<t}, V_{<t}\right]\right] \geq \inf_{V_{<t}} \mathbb{P}\left[G_t \mid G_{<t}, V_{<t}\right] \tag{28}$$

For any particular $V_{<t}$,

$$\mathbb{P}\left[G_t \mid G_{<t}, V_{<t}\right] = \mathbb{P}\left[\forall x \in X_t \; \forall j \geq t \; \left|\left\langle \frac{P_{t-1}^{\perp}x}{\|P_{t-1}^{\perp}x\|}, v_j \right\rangle\right| \leq \alpha \; \middle| \; G_{<t}, V_{<t}\right] \tag{29}$$

$$\geq 1 - \sum_{x \in X_t(V_{<t})} \sum_{j=t}^{k} \mathbb{P}\left[\left|\left\langle \frac{P_{t-1}^{\perp}x}{\|P_{t-1}^{\perp}x\|}, v_j \right\rangle\right| > \alpha \; \middle| \; G_{<t}, V_{<t}\right] \tag{30}$$

$$\geq 1 - \sum_{x \in X_t(V_{<t})} \sum_{j=t}^{k} \mathbb{P}\left[\left|\left\langle \frac{P_{t-1}^{\perp}x}{\|P_{t-1}^{\perp}x\|}, \frac{P_{t-1}^{\perp}v_j}{\|P_{t-1}^{\perp}v_j\|} \right\rangle\right| > \alpha \; \middle| \; G_{<t}, V_{<t}\right] \tag{31}$$

Conditioned on $G_{<t}$ and $V_{<t}$, the set $X_t = X_t(V_{<t})$ is fixed, as is the set S_{t-1} and therefore $P_{t-1}^{\perp}$, so the first term in the inner product is a fixed unit vector. By Lemma 3, the conditional density of $v_j \mid G_{<t}, V_{<t}$ is spherically symmetric within the span onto which $P_{t-1}^{\perp}$ projects. Therefore, $\frac{P_{t-1}^{\perp}v_j}{\|P_{t-1}^{\perp}v_j\|}$ is distributed uniformly on the unit sphere in $\mathrm{Span}\left(S_{t-1}\right)^{\perp}$, which has dimension at least $m' := m - (t-1) - \sum_{r=1}^{t-1}|X_r| \geq m - k + 1 - N$.

The probability of a fixed vector and a uniform random vector on the unit sphere in $\mathbb{R}^{m'}$ having inner product more than α is proportional to the surface area of the "end caps" of the sphere lying above and below circles of radius $\sqrt{1-\alpha^2}$, which is strictly smaller than the surface area of a full sphere of radius $\sqrt{1-\alpha^2}$. Therefore, for a given x, v_j

$$\mathbb{P}\left[\left|\left\langle \frac{P_{t-1}^{\perp}x}{\|P_{t-1}^{\perp}x\|}, \frac{P_{t-1}^{\perp}v_j}{\|P_{t-1}^{\perp}v_j\|} \right\rangle\right| > \alpha \; \middle| \; G_{<t}, V_{<t}\right] < \frac{\mathrm{SurfaceArea}_{m'}(\sqrt{1-\alpha^2})}{\mathrm{SurfaceArea}_{m'}(1)} \tag{32}$$

$$= \left(\sqrt{1-\alpha^2}\right)^{m'-1} \tag{33}$$

$$\leq \exp\left(-\frac{(m'-1)\alpha^2}{2}\right) \tag{34}$$

where we used that $1 - x \leq \exp(-x)$. Finally, this holds for each t, $x \in X_t$, and $j \geq t$, so

$$\mathbb{P}\left[G_{\leq k}\right] \geq \prod_{t \leq k} \inf_{V_{<t}} \mathbb{P}\left[G_t \mid G_{<t}, V_{<t}\right] \tag{35}$$

$$\geq \left(1 - kN \exp\left(-\frac{(m-k-N)\alpha^2}{2}\right)\right)^k \tag{36}$$

$$\geq 1 - k^2 N \exp\left(-\frac{\alpha^2}{2} \max\left\{32N^2, \frac{8(1+\sqrt{2N})^2}{c^2}\right\} \log\left(\frac{1}{2k^2N}\right)\right) \tag{37}$$

$$= \frac{1}{2} \tag{38}$$

Where we used that $m \geq k + N + \max\left\{32N^2, \frac{8(1+\sqrt{2N})^2}{c^2}\right\} \log\left(2k^2N\right)$ for (37). For (38), recall that we chose $\alpha = \min\left\{\frac{1}{4N}, \frac{c}{2(1+\sqrt{2N})}\right\}$ so $\max\left\{32N^2, \frac{8(1+\sqrt{2N})^2}{c^2}\right\} = \frac{2}{\alpha^2}$. $\qquad\square$

A.1 Proof of Lemmas 1 and 3

Lemma 1. *[cf. Lemma 9 [8], Lemma 1 [26]] For any c, k, N, V, and $\{X_k\}_{t=1}^{k}$, let $\alpha = \min\left\{\frac{1}{4N}, \frac{c}{2\left(1+\sqrt{2N}\right)}\right\}$ then for each $t \leq k$*

$$G_{\leq t} \implies G'_{\leq t} := \left[\!\left[\forall r \leq t, \; \forall x \in X_r, \; \forall j \geq t \; |\langle x, v_j \rangle| \leq \frac{c}{2}\right]\!\right]$$

13

Proof. This closely follows the proof of Lemma 9 [8], with slight modification to account for the different problem setup.

For $t \leq k$ assume $G_{\leq t}$. For any $x \in X_t$ and $j \geq t$

$$|\langle x, v_j \rangle| \leq \|x\| \left| \left\langle \frac{x}{\|x\|}, P_{t-1}v_j \right\rangle \right| + \|x\| \left| \left\langle \frac{x}{\|x\|}, P_{t-1}^{\perp}v_j \right\rangle \right| \tag{39}$$

$$\leq \|P_{t-1}v_j\| + \left| \left\langle \frac{P_{t-1}^{\perp}x}{\|x\|}, v_j \right\rangle \right| \tag{40}$$

$$\leq \|P_{t-1}v_j\| + \left| \left\langle \frac{P_{t-1}^{\perp}x}{\|P_{t-1}^{\perp}x\|}, v_j \right\rangle \right| \tag{41}$$

$$\leq \|P_{t-1}v_j\| + \alpha \tag{42}$$

First, we decomposed v_j into its S_{t-1} and $S_{t-1}^{\perp}$ components and applied the triangle inequality. Next we used that $\|x\| \leq 1$ and that the orthogonal projection operator $P_{t-1}^{\perp}$ is self-adjoint. Finally, we used that the projection operator is non-expansive and the definition of G_t.

Next, we prove by induction on t that for all $t \leq k$ and $j \geq t$, the event $G_{\leq t}$ implies that $\|P_{t-1}v_j\|^2 \leq 2\alpha^2 \sum_{r=1}^{t-1} |X_r|$. As a base case ($t = 1$), observe that, trivially, $\|P_{t-1}v_j\|^2 = \|0v_j\|^2 = 0$. For the inductive step, fix any $t \leq k$ and $j \geq t$ and suppose that $G_{\leq t'} \implies \|P_{t'-1}v_j'\|^2 \leq 2\alpha^2 \sum_{r=1}^{t'-1} |X_r|$ for all $t' < t$ and $j' \geq t'$. Let $\hat{P}_t$ project onto $\text{Span}(S_t \cup X_{t+1})$ (this includes X_{t+1} in contrast with P_t) and let $\hat{P}_t^{\perp}$ project onto the orthogonal subspace. Since $\text{Span}(X_1 \cup X_2 \cup \cdots \cup X_{t-1} \cup V_{\leq t-1}) = S_{t-1}$,

$$\left\{ \frac{P_{r-1}^{\perp}x}{\|P_{r-1}^{\perp}x\|} : r \leq t-1, \ x \in X_r \right\} \cup \left\{ \frac{\hat{P}_{r-1}^{\perp}v_r}{\|\hat{P}_{r-1}^{\perp}v_r\|} : r \leq t-1 \right\} \tag{43}$$

is a (potentially over-complete) basis for S_{t-1}. Using the triangle inequality and $G_{<t}$, we can therefore expand

$$\|P_{t-1}v_j\|^2 = \sum_{r=1}^{t-1} \sum_{x \in X_r} \left\langle \frac{P_{r-1}^{\perp}x}{\|P_{r-1}^{\perp}x\|}, v_j \right\rangle^2 + \sum_{r=1}^{t-1} \left\langle \frac{\hat{P}_{r-1}^{\perp}v_r}{\|\hat{P}_{r-1}^{\perp}v_r\|}, v_j \right\rangle^2 \tag{44}$$

$$\leq \alpha^2 \sum_{r=1}^{t-1} |X_r| + \sum_{r=1}^{t-1} \frac{1}{\|\hat{P}_{r-1}^{\perp}v_r\|^2} \left\langle \hat{P}_{r-1}^{\perp}v_r, v_j \right\rangle^2 \tag{45}$$

We must now bound the second term of (45). Focusing on the inner product in the numerator for one particular $r < t$:

$$\left| \left\langle \hat{P}_{r-1}^{\perp}v_r, v_j \right\rangle \right| = \left| \langle v_r, v_j \rangle - \left\langle \hat{P}_{r-1}v_r, v_j \right\rangle \right| \tag{46}$$

$$= \left| \left\langle \hat{P}_{r-1}v_r, v_j \right\rangle \right| \tag{47}$$

$$\leq |\langle P_{r-1}v_r, v_j \rangle| + \sum_{x \in X_r} \left| \left\langle \frac{P_{r-1}^{\perp}x}{\|P_{r-1}^{\perp}x\|}, v_r \right\rangle \left\langle \frac{P_{r-1}^{\perp}x}{\|P_{r-1}^{\perp}x\|}, v_j \right\rangle \right| \tag{48}$$

$$\leq \|P_{r-1}v_r\| \|P_{r-1}v_j\| + |X_r|\alpha^2 \tag{49}$$

$$\leq 2\alpha^2 \sum_{i=1}^{r-1} |X_i| + |X_r|\alpha^2 \tag{50}$$

$$\leq \frac{\alpha}{2} \tag{51}$$

First, we used that $\hat{P}_{r-1}^{\perp} = I - \hat{P}_{r-1}$, then that $v_r \perp v_j$. Next, we applied the definition of $\hat{P}_{r-1}$ and the triangle inequality. To get (49) we use the Cauchy-Schwarz inequality on the first term, and the definition of G_r for the second. Finally, we use the inductive hypothesis and that $\alpha \leq \frac{1}{4N}$.

14

We have now upper bounded the inner products in the second term of (45), and it remains to lower bound the norm in the denominator. We can rewrite

$$\left\|\hat{P}_{r-1}^{\perp}v_r\right\|^2 = \left\langle \hat{P}_{r-1}^{\perp}v_r,\, v_r \right\rangle \tag{52}$$

$$= \langle v_r,\, v_r \rangle - \left\langle \hat{P}_{r-1}v_r,\, v_r \right\rangle \tag{53}$$

$$\geq 1 - \langle P_{r-1}v_r,\, v_r \rangle - \sum_{x \in X_r} \left\langle \frac{P_{r-1}^{\perp}x}{\left\|P_{r-1}^{\perp}x\right\|},\, v_r \right\rangle^2 \tag{54}$$

$$\geq 1 - \|P_{r-1}v_r\|^2 - |X_r|\alpha^2 \tag{55}$$

$$\geq 1 - 2\alpha^2 \sum_{i=1}^{r-1} |X_i| - |X_r|\alpha^2 \tag{56}$$

$$\geq \frac{1}{2} \tag{57}$$

Here we again used $\hat{P}_{r-1}^{\perp} = I - \hat{P}_{r-1}$ followed by an (over)expansion of $\hat{P}_{r-1}$. The remaining steps follow from the inductive hypothesis and fact that $\alpha \leq \frac{1}{4N}$. Combining (57) with (51) and returning to (45), we have that

$$\|P_{t-1}v_j\|^2 \leq \alpha^2 \sum_{r=1}^{t-1} |X_r| + \sum_{r=1}^{t-1} \frac{1}{\left\|\hat{P}_{r-1}^{\perp}v_r\right\|^2} \left\langle \hat{P}_{r-1}^{\perp}v_r,\, v_j \right\rangle^2 \tag{58}$$

$$\leq \alpha^2 \sum_{r=1}^{t-1} |X_r| + \sum_{r=1}^{t-1} \alpha^2 \tag{59}$$

$$\leq 2\alpha^2 \sum_{r=1}^{t-1} |X_r| \tag{60}$$

Therefore, for each $t \leq k$ and $j \geq t$ an upper bound $\|P_{t-1}v_j\|^2 \leq 2\alpha^2 \sum_{r=1}^{t-1} |X_r|$. Returning now to (42), we have that for any $t \leq k$, $x \in X_t$, and $j \geq t$ the event $G_{\leq t}$ implies

$$|\langle x,\, v_j \rangle| \leq \|P_{t-1}v_j\| + \alpha \tag{61}$$

$$\leq \alpha \left(1 + \sqrt{2 \sum_{r=1}^{t-1} |X_r|} \right) \tag{62}$$

$$\leq \frac{c}{2} \tag{63}$$

where we used that $\alpha \leq \frac{c}{2\left(1+\sqrt{2N}\right)}$. $\square$

Lemma 3. *[cf. Lemma 11 [8], Lemma 3 [26]] Let R be any rotation operator, $R^{\top}R = I$, that preserves S_{t-1}, that is $Rw = R^{\top}w = w$ for any $w \in \text{Span}(S_{t-1})$. Then the following conditional densities are equal*

$$p\left(V_{\geq t} \mid G_{<t}, V_{<t}\right) = p\left(RV_{\geq t} \mid G_{<t}, V_{<t}\right)$$

Proof. This closely follows the proof of Lemma 11 [8].

First, we apply Bayes' rule to each density and use the fact that $RV_{<t} = V_{<t}$:

$$p\left(V_{\geq t} \mid G_{<t}, V_{<t}\right) = \frac{\mathbb{P}\left[G_{<t} \mid V\right] p(V)}{\mathbb{P}\left[G_{<t} \mid V_{<t}\right] p(V_{<t})} \tag{64}$$

$$p\left(RV_{\geq t} \mid G_{<t}, V_{<t}\right) = \frac{\mathbb{P}\left[G_{<t} \mid RV\right] p(RV)}{\mathbb{P}\left[G_{<t} \mid V_{<t}\right] p(V_{<t})} \tag{65}$$

Since V has a spherically symmetric marginal distribution, $p(V) = p(RV)$. Therefore, it only remains to show that $\mathbb{P}\left[G_{<t} \mid V\right] = \mathbb{P}\left[G_{<t} \mid RV\right]$. The event $G_{<t}$ is determined by V or by RV, thus both probabilities are either 0 or 1, so it suffices to show $\mathbb{P}\left[G_{<t} \mid V\right] = 1 \iff \mathbb{P}\left[G_{<t} \mid RV\right] = 1$.

Assume first $\mathbb{P}\left[G_{<t} \mid V\right] = 1$. Then for each $r < t$, $x \in X_r$, and $j \geq r$ $\left|\left\langle \frac{P_{r-1}^{\perp}x}{\|P_{r-1}^{\perp}x\|}, v_j \right\rangle\right| \leq \alpha$, and each set X_r is a deterministic function of $V_{<r}$. Also, observe that for any $x \in X_r$ and $j \geq r$,

$$\left|\left\langle \frac{P_{r-1}^{\perp}x}{\|P_{r-1}^{\perp}x\|}, Rv_j \right\rangle\right| = \left|\left\langle \frac{R^{\top}P_{r-1}^{\perp}x}{\|P_{r-1}^{\perp}x\|}, v_j \right\rangle\right| = \left|\left\langle \frac{P_{r-1}^{\perp}x}{\|P_{r-1}^{\perp}x\|}, v_j \right\rangle\right| \leq \alpha \tag{66}$$

where we used that $P_{r-1}^{\perp}x \in \operatorname{Span}(S_r) \subseteq \operatorname{Span}(S_{t-1})$ so $R^{\top}P_{r-1}^{\perp}x = P_{r-1}^{\perp}x$. Therefore, it suffices to show that the sequence $X_1(RV), ..., X_t(RV) = X_1(V), ..., X_t(V)$ when $\mathbb{P}\left[G_{<t} \mid V\right] = 1$. We prove this by induction.

For the base case, by definition $X_1(RV) = X_1 = X_1(V)$. For the inductive step, suppose now that $X_{r'}(RV) = X_{r'}(V)$ for each $r' < r$. This, plus the fact that $\mathbb{P}\left[G_{<t} \mid V\right] = 1 \implies \mathbb{P}\left[G_{<r} \mid V\right] = 1$ together imply that $\mathbb{P}\left[G_{<r} \mid RV\right] = 1$. Thus, $X_r(RV) = X_r(RV_{<r}) = X_r(V_{<r})$. Therefore, we conclude that $\mathbb{P}\left[G_{<t} \mid V\right] = 1 \implies \mathbb{P}\left[G_{<t} \mid RV\right] = 1$, the reverse implication can be proven with a similar argument. □

B Proof of Theorem 1

Theorem 1. *Let $L, B \in (0, \infty)$, $H \in [0, \infty]$, $N \geq D \geq 1$, let $\mathcal{G}$ be any oracle graph of depth D and size N and consider the optimization problem $(\mathcal{G}, \mathcal{O}_{grad}, \mathcal{F}_{L,H,B})$. For any randomized algorithm $\mathcal{A} = (R_1, \ldots, R_N, \hat{X})$, there exists a distribution $\mathcal{P}$ and a convex, L-Lipschitz, and H-smooth function f on a B-bounded domain in $\mathbb{R}^m$ for $m = O\left(\max\left\{N^2, D^3 N\right\}\log(DN)\right)$ such that*

$$\mathbb{E}_{\substack{z \sim \mathcal{P} \\ \hat{X} \sim \mathcal{A}}}\left[f(\hat{X}; z)\right] - \min_x \mathbb{E}_{z \sim \mathcal{P}}\left[f(x; z)\right] \geq \Omega\left(\min\left\{\frac{LB}{\sqrt{D}}, \frac{HB^2}{D^2}\right\} + \frac{LB}{\sqrt{N}}\right)$$

Proof. Assume for now that $B = 1$, the lower bound can be established for other values of B by scaling inputs to our construction. Let

$$\ell = \min\left\{L, \frac{H}{10(D+1)^{1.5}}\right\} \qquad \eta = 10(D+1)^{1.5}\ell \tag{67}$$

and consider the following ℓ-Lipschitz function:

$$\tilde{f}(x) = \max_{1 \leq r \leq D+1} \ell v_r^{\top}x - \frac{5\ell^2(r-1)}{\eta} \tag{68}$$

where the vectors $v_1, \ldots, v_{D+1}$ are an orthonormal set drawn uniformly at random from the unit sphere in $\mathbb{R}^m$. We use the η-Moreau envelope [5] of this function in order to prove our lower bound:

$$f(x) = \inf_y \left\{\tilde{f}(y) + \frac{\eta}{2}\|y - x\|^2\right\} \tag{69}$$

The random draw of V defines a distribution over functions f. We will lower bound the expected suboptimality of any *deterministic* optimization algorithm's output and apply Yao's minimax principle at the end of the proof.

This function has the following properties:

Lemma 4. *The function f is convex, ℓ-Lipschitz, and η-smooth, with $\ell \leq L$ and $\eta \leq H$.*

Furthermore, optimizing f is equivalent to "finding" the vectors $v_1, \ldots, v_{D+1}$. In particular, until a point that has a substantial inner product with all of $v_1, \ldots, v_{D+1}$ is found, the algorithm will remain far from the minimum:

Lemma 5. *For any $H, L > 0$, $D \geq 1$, and orthonormal $v_1, ..., v_{D+1}$, for any x with $\left|v_{D+1}^{\top}x\right| \leq \frac{\ell}{\eta}$*

$$f(x) - \min_{x:\|x\| \leq 1} f(x) \geq \min\left\{\frac{L}{2\sqrt{D+1}}, \frac{H}{20(D+1)^2}\right\}$$

The function also has the property that if x has a small inner product with $v_t, \ldots, v_{D+1}$, then the gradient oracle will reveal little information about f when queried at x:

16

Lemma 6. *For any x with $|\langle x, v_r \rangle| \leq \frac{\ell}{\eta}$ for all $r \geq t$, both the function value $f(x)$ and gradient $\nabla f(x)$ can be calculated from $v_1, \ldots, v_t$ only.*

In Appendix A, we studied the situation where orthonormal $v_1, \ldots, v_{D+1}$ are chosen uniformly at random and a sequence of sets of vectors $X_1, \ldots, X_{D+1}$ are generated as

$$X_t(V) = X_t \left(V_{<t} \mathbb{1}_{G'_{<t}} + V \mathbb{1}_{\neg G'_{<t}} \right) \tag{70}$$

where

$$G'_{<t} = \left[\!\!\left[\forall r < t, \ \forall x \in X_r, \ \forall j \geq r \ |\langle x, v_j \rangle| \leq \frac{c}{2} \right]\!\!\right] \tag{71}$$

Take $c = \frac{2\ell}{\eta}$ and consider the dependency graph. Let X_1 be the set of queries made in vertices at depth 1 in the graph (i.e. they have no parents). Let X_2 be the set of queries made in vertices at depth 2 in the graph (i.e. their parents correspond to the queries in X_1). Continue in this way for each $t \leq D$, and let $X_{D+1} = \{\hat{x}\}$ correpond to the algorithm's output, which is allowed to depend on all queries and oracle responses in the graph, and thus has depth $D + 1$.

Supposing $G'_{<t}$, for all queries $x \in X_1 \cup \cdots \cup X_{t-1}$ and for all $r \geq t-1$ we have $|\langle x, v_r \rangle| \leq \frac{c}{2} = \frac{\ell}{\eta}$. Therefore, by Lemma 6 all of the function evaluations and gradients returned by the stochastic gradient oracle are calculable from $v_1, \ldots, v_{t-1}$ only. Therefore, all of the queries in X_t are a deterministic function of $V_{<t}$ (since we are currently considering only deterministic optimization algorithms), so X_t satisfies the required decomposition property (70). Finally, the queries are required to be in the domain of f, thus they will have norm bounded by 1.

Therefore, by Lemma 2, when the dimension

$$m \geq D + 1 + N + \max\left\{ 32N^2, \ 200 \left(D + 1\right)^3 \left(1 + \sqrt{2N}\right)^2 \right\} \log \left(2(D+1)^2 N \right) \tag{72}$$

with probability $1/2$, all $x \in X_1 \cup \cdots \cup X_{t+1}$ including the algorithm's output $\hat{x}$ satsify $|\langle x, v_{D+1} \rangle| \leq \frac{\ell}{\eta}$ so by Lemma 5

$$f(\hat{x}) - \min_{x:\|x\|\leq 1} f(x) \geq \min\left\{ \frac{L}{2\sqrt{D+1}}, \ \frac{H}{20(D+1)^2} \right\} \tag{73}$$

Therefore, by Yao's minimax principle for any randomized algorithm $\mathcal{A}$

$$\max_V \mathbb{E}_{\hat{X} \sim \mathcal{A}} \left[f(\hat{X}) \right] - \min_{x:\|x\|\leq 1} f(x) \geq \min_{\text{deterministic } \mathcal{A}} \mathbb{E}_V \left[f(\hat{x}) \right] - \min_{x:\|x\|\leq 1} f(x)$$

$$\geq \min\left\{ \frac{L}{4\sqrt{D+1}}, \ \frac{H}{40(D+1)^2} \right\} \tag{74}$$

The statistical term $\frac{L}{8\sqrt{N}}$ follows from Lemma 10. $\qquad\square$

B.1 Deferred proofs

Lemma 4. *The function f is convex, ℓ-Lipschitz, and η-smooth, with $\ell \leq L$ and $\eta \leq H$.*

Proof. Since $\tilde{f}$ is the maximum of ℓ-Lipschitz affine functions, it is convex and ℓ-Lipschitz. Furthermore, by Proposition 12.29 [5], f, the η-Moreau Envelope of $\tilde{f}$ is η-smooth and

$$\nabla f(x) = \eta \left(x - \arg\min_y \tilde{f}(y) + \frac{\eta}{2} \|y - x\|^2 \right) \tag{75}$$

The minimizing y satisfies that $\eta(x - y) \in \partial \tilde{f}(y)$ (where $\partial \tilde{f}(y)$ denotes the set of subgradients of $\tilde{f}$ at y), and since $\tilde{f}$ is ℓ-Lipschitz this implies that $\|\nabla f(x)\| \leq \ell$. $\qquad\square$

Lemma 5. *For any $H, L > 0$, $D \geq 1$, and orthonormal $v_1, ..., v_{D+1}$, for any x with $\left| v_{D+1}^\top x \right| \leq \frac{\ell}{\eta}$*

$$f(x) - \min_{x:\|x\|\leq 1} f(x) \geq \min\left\{ \frac{L}{2\sqrt{D+1}}, \ \frac{H}{20(D+1)^2} \right\}$$

17

Proof. First

$$\min_{x:\|x\|\leq 1} f(x) \leq f\left(-\sum_{r=1}^{D+1} \frac{v_r}{\sqrt{D+1}}\right) \leq \tilde{f}\left(-\sum_{r=1}^{D+1} \frac{v_r}{\sqrt{D+1}}\right) \leq -\frac{\ell}{\sqrt{D+1}} \tag{76}$$

Now, for an arbitrary point x such that $\left|v_{D+1}^\top x\right| \leq \frac{\ell}{\eta} = \frac{1}{10(D+1)^{1.5}}$, consider

$$y^* = \mathrm{prox}_{\tilde{f}}(x,\eta) = \arg\min_y \left\{ \max_{1\leq r\leq D+1}\left(\ell v_r^\top y - \frac{5\ell^2(r-1)}{\eta}\right) + \frac{\eta}{2}\|y-x\|^2 \right\} \tag{77}$$

Since y^* is the minimizer, $\eta(x-y^*) \in \partial\tilde{f}(y^*)$ and since $\tilde{f}$ is ℓ-Lipschitz, $\|x-y^*\| \leq \frac{\ell}{\eta}$. Thus $v_{D+1}^\top y^* \geq -\frac{2\ell}{\eta}$ and

$$f(x) = \tilde{f}(y^*) + \frac{\eta}{2}\|y^*-x\|^2 \tag{78}$$

$$= \max_{1\leq r\leq D+1}\left(\ell v_r^\top y^* - \frac{5\ell^2(r-1)}{\eta}\right) + \frac{\eta}{2}\|y^*-x\|^2 \tag{79}$$

$$\geq \ell v_{D+1}^\top y^* - \frac{5\ell^2 D}{\eta} \tag{80}$$

$$\geq -\frac{2\ell^2}{\eta} - \frac{5\ell^2 D}{\eta} \tag{81}$$

$$\geq -\frac{5\ell^2(D+1)}{\eta} \tag{82}$$

Combining (76) and (82), for any x such that $\left|v_{D+1}^\top x\right| \leq \frac{\ell}{\eta}$

$$f(x) - \min_{x:\|x\|\leq 1} f(x) \geq \frac{\ell}{\sqrt{D+1}} - \frac{5\ell^2(D+1)}{\eta} = \min\left\{\frac{L}{2\sqrt{D+1}}, \frac{H}{20(D+1)^2}\right\} \tag{83}$$

$\square$

Lemma 6. *For any x with $|\langle x, v_r\rangle| \leq \frac{\ell}{\eta}$ for all $r \geq t$, both the function value $f(x)$ and gradient $\nabla f(x)$ can be calculated from $v_1, \ldots, v_t$ only.*

Proof. Let x be a point such that $\left|v_r^\top x\right| \leq \frac{\ell}{\eta}$ for all $r \geq t$. By Proposition 12.29 [5]

$$\nabla f(x) = \eta\left(x - \mathrm{prox}_{\tilde{f}}(x,\eta)\right) \tag{84}$$

Since f is ℓ-Lipschitz (Lemma 4), $\left\|x - \mathrm{prox}_{\tilde{f}}(x,\eta)\right\| \leq \frac{\ell}{\eta}$. Consequently, for $y^* = \mathrm{prox}_{\tilde{f}}(x,\eta)$ we have $\left|v_r^\top y^*\right| \leq \frac{2\ell}{\eta}$ for all $r \geq t$. Furthermore,

$$\nabla f(x) = \eta(x-y^*) \in \mathrm{conv}\left\{\ell v_r : r \in \arg\max_{1\leq r\leq D+1}\left(\ell v_r^\top y^* - \frac{5\ell^2(r-1)}{\eta}\right)\right\} \tag{85}$$

For any $r > t$

$$\ell v_r^\top y^* - \frac{5\ell^2(r-1)}{\eta} \leq \frac{2\ell^2}{\eta} - \frac{5\ell^2(r-1)}{\eta} = -\frac{5\ell^2\left(r-\frac{7}{5}\right)}{\eta} \tag{86}$$

Whereas

$$\ell v_t^\top y^* - \frac{5\ell^2(t-1)}{\eta} \geq -\frac{2\ell^2}{\eta} - \frac{5\ell^2(t-1)}{\eta} = -\frac{2\ell^2\left(t-\frac{3}{5}\right)}{\eta} \tag{87}$$

For any $r > t$ (86) is less than (87), thus no $r > t$ can be in the $\arg\max$ in (85). Therefore, using only $v_1, \ldots, v_t$ we can calculate

$$f(x) = \inf_y\left\{\max_{1\leq r\leq D+1}\left(\ell v_r^\top y - \frac{5\ell^2(r-1)}{\eta}\right) + \frac{\eta}{2}\|y-x\|^2\right\} \tag{88}$$

$$= \inf_y\left\{\max_{1\leq r\leq t}\left(\ell v_r^\top y - \frac{5\ell^2(r-1)}{\eta}\right) + \frac{\eta}{2}\|y-x\|^2\right\} \tag{89}$$

$$\tag{90}$$

and

$$\text{prox}_{\tilde{f}}(x, \eta) = \arg\min_y \left\{ \max_{1 \leq r \leq D+1} \left(\ell v_r^\top y - \frac{5\ell^2(r-1)}{\eta} \right) + \frac{\eta}{2} \|y - x\|^2 \right\} \tag{91}$$

$$= \arg\min_y \left\{ \max_{1 \leq r \leq t} \left(\ell v_r^\top y - \frac{5\ell^2(r-1)}{\eta} \right) + \frac{\eta}{2} \|y - x\|^2 \right\} \tag{92}$$

from which we get $\nabla f(x) = \eta(x - \text{prox}_{\tilde{f}}(x, \eta))$. $\qquad\square$

C Proof of Theorem 2

Theorem 2. *Let $L, B \in (0, \infty)$, $H \in [0, \infty]$, $N \geq D \geq 1$, let $\mathcal{G}$ be any oracle graph of depth D and size N and consider the optimization problem $(\mathcal{G}, \mathcal{O}_{prox}, \mathcal{F}_{L,H,B})$. For any randomized algorithm $\mathcal{A} = (R_1, \ldots, R_N, \hat{X})$, there exists a distribution $\mathcal{P}$ and a convex, L-Lipschitz, and H-smooth function f on a B-bounded domain in $\mathbb{R}^m$ for $m = O\left(\max\{N^2, D^3 N\} \log(DN) \right)$ such that*

$$\mathbb{E}_{\substack{z \sim \mathcal{P} \\ \hat{X} \sim \mathcal{A}}} \left[f(\hat{X}; z) \right] - \min_x \mathbb{E}_{z \sim \mathcal{P}} \left[f(x; z) \right] \geq \Omega \left(\min\left\{ \frac{LB}{D}, \frac{HB^2}{D^2} \right\} + \frac{LB}{\sqrt{N}} \right)$$

Proof. Without loss of generality, assume $B = 1$, the lower bound can be proven for other values of B by scaling inputs to our construction by $1/B$. Let

$$\eta = \min\{H, 2LD\} \qquad \gamma = \frac{4L}{\eta\sqrt{2D}} \qquad a = 2c = \frac{1}{\sqrt{8D^3}} \tag{93}$$

Define the following scalar function

$$\phi_c(z) = \begin{cases} 0 & |z| \leq c \\ 2(|z| - c)^2 & c < |z| \leq 2c \\ z^2 - 2c^2 & 2c < |z| \leq \gamma \\ 2\gamma |z| - \gamma^2 - 2c^2 & |z| > \gamma \end{cases} \tag{94}$$

It is straightforward to confirm that ϕ_c is convex, 2γ-Lipschitz continuous, and 4-smooth. Let $\mathcal{P}$ be the uniform distribution over $\{1, 2\}$. Let $v_1, v_2, \ldots, v_{2D}$ be a set of orthonormal vectors drawn uniformly at random and define

$$f(x; 1) = \frac{\eta}{8} \left(-2a v_1^\top x + \phi_c(v_{2D}^\top x) + \sum_{r=3,5,7,\ldots}^{2D-1} \phi_c(v_{r-1}^\top x - v_r^\top x) \right) \tag{95}$$

$$f(x; 2) = \frac{\eta}{8} \left(\sum_{r=2,4,6,\ldots}^{2D} \phi_c(v_{r-1}^\top x - v_r^\top x) \right) \tag{96}$$

$$F(x) = \mathbb{E}_{z \sim \mathcal{P}} [f(x; z)] = \frac{1}{2} (f(x; 1) + f(x; 2)) \tag{97}$$

$$= \frac{\eta}{16} \left(-2a v_1^\top x + \phi_c(v_{2D}^\top x) + \sum_{r=2}^{2D} \phi_c(v_{r-1}^\top x - v_r^\top x) \right) \tag{98}$$

The random choice of V determines a distribution over functions $f(\cdot; 1)$ and $f(\cdot; 2)$. We will lower bound the expectation (over V) of the suboptimality of any deterministic algorithm's output, and then apply Yao's minimax principle.

First, we show that the functions $f(\cdot; 1)$ and $:= f(\cdot; 2)$ are convex, L-Lipschitz, and H-smooth:

Lemma 7. *For any $H, L \geq 0$, $D \geq 1$, and orthonormal $v_1, \ldots, v_{2D}$, and with η, γ, a, and c chosen as in (93), $f(\cdot; 1)$ and $f(\cdot; 2)$ are convex, L-Lipschitz, and H-smooth.*

Next, we show that optimizing F is equivalent to "finding" a large number of the vectors $v_1, \ldots, v_{2D}$:

Lemma 8. *For any $H, L \geq 0$, $D \geq 1$, and orthonormal $v_1, ..., v_{2D}$, and with $\eta, \gamma, a,$ and c chosen as in (93), for any x such that $\left|v_r^\top x\right| \leq \frac{c}{2}$ for all $r > D$*

$$F(x) - \min_{x:\|x\| \leq 1} F(x) \geq \min\left\{\frac{L}{32D}, \frac{H}{64D^2}\right\}$$

Next, we show that at any point x such that $\left|v_r^\top x\right| \leq \frac{c}{2}$ for all $r \geq t$, the function value, gradient, and prox of $f(\cdot; 1)$ and $f(\cdot; 2)$ at x are calculable using $v_1, \ldots, v_t$ only:

Lemma 9. *For any x such that $\left|v_r^\top x\right| \leq \frac{c}{2}$ for all $r \geq t$, and any $\beta \geq 0$ the function values, gradients, and proxs $f(x; 1)$, $f(x; 2)$, $\nabla f(x; 1)$, $\nabla f(x; 2)$, $\text{prox}_{f(\cdot, 1)}(x, \beta)$, and $\text{prox}_{f(\cdot, 2)}(x, \beta)$ are calculable using $\beta, x, v_1, \ldots, v_t$ only.*

In Appendix A, we studied the situation where orthonormal $v_1, \ldots, v_{2D}$ are chosen uniformly at random and a sequence of sets of vectors $X_1, \ldots, X_{2D}$ are generated as

$$X_t(V) = X_t\left(V_{<t}\mathbb{1}_{G'_{<t}} + V\mathbb{1}_{\neg G'_{<t}}\right) \tag{99}$$

where

$$G'_{<t} = \left[\!\!\left[\forall r < t, \; \forall x \in X_r, \; \forall j \geq r \; |\langle x, v_j\rangle| \leq \frac{c}{2}\right]\!\!\right] \tag{100}$$

Consider the dependency graph, and let X_1 be the set of queries made in vertices at depth 1 in the graph (i.e. they have no parents). Let X_2 be the set of queries made in vertices at depth 2 in the graph (i.e. their parents correspond to the queries in X_1). Continue in this way for each $t \leq D$, and then let $X_{D+1} = \{\hat{x}\}$ correpond to the output of the optimization algorithm, which for now is deterministic.

Suppose $G'_{<t}$. Then for all of the queries $x \in X_1 \cup \cdots \cup X_{t-1}$ and for all $r \geq t-1$ we have $|\langle x, v_r\rangle| \leq \frac{c}{2}$. Therefore, by Lemma 9 the function values, gradients, and proxs of $f(\cdot; 1)$ and $f(\cdot; 2)$ are calculable based only on the query points and $v_1, \ldots, v_{t-1}$. Therefore, all of the queries in X_t are a deterministic function of $V_{<t}$ only so X_t satisfies the required decomposition property (99). Finally, the queries are required to be in the domain of f, thus they will have norm bounded by B.

Therefore, by Lemma 2 for

$$m \geq 2D + N + \max\left\{32N^2, \; 128B^2D^3(1+\sqrt{2N})^2\right\}\log\left(8D^2N\right) \tag{101}$$

with probability $1/2$ for every $x \in X_1 \cup \cdots \cup X_{D+1}$ which includes $\hat{x}$, $|\langle x, v_r\rangle| \leq \frac{c}{2}$ for $r > D$, so by Lemma 8

$$f(\hat{x}) - \min_{x:\|x\| \leq 1} f(x) \geq \min\left\{\frac{L}{32D}, \frac{H}{64D^2}\right\} \tag{102}$$

Therefore,

$$\min_{\text{deterministic } \mathcal{A}} \mathbb{E}_V\left[f(\hat{x}) - \min_{x:\|x\| \leq 1} f(x)\right] \geq \min\left\{\frac{L}{64D}, \frac{H}{128D^2}\right\} \tag{103}$$

so by Yao's minimax principle, for any randomized algorithm $\mathcal{A}$

$$\max_V \mathbb{E}_{\hat{X} \sim \mathcal{A}}\left[f(\hat{X}) - \min_{x:\|x\| \leq 1} f(x)\right] \geq \min\left\{\frac{L}{64D}, \frac{H}{128D^2}\right\} \tag{104}$$

The statistical term $\frac{LB}{8\sqrt{N}}$ follows from Lemma 10. $\qquad\square$

C.1 Deferred proof

Lemma 7. *For any $H, L \geq 0$, $D \geq 1$, and orthonormal $v_1, ..., v_{2D}$, and with $\eta, \gamma, a,$ and c chosen as in (93), $f(\cdot; 1)$ and $f(\cdot; 2)$ are convex, L-Lipschitz, and H-smooth.*

Proof. The functions $f(\cdot; 1)$ and $f(\cdot; 2)$ are a sum of linear functions and ϕ_c, which is convex; therefore both are convex. Also, the scalar function ϕ_c is 2γ-Lipschitz, so

$$\|\nabla f(x; 1)\|^2 = \left\| \frac{\eta}{8} \left(-2av_1 + \phi_c' \left(v_{2D}^\top x \right) v_{2D} + \sum_{r=3,5,7,\ldots}^{2D-1} \phi_c' \left(v_{r-1}^\top x - v_r^\top x \right) (v_{r-1} - v_r) \right) \right\|^2$$

(105)

$$\leq \frac{\eta^2 \left(a^2 + (2D-1)\gamma^2 \right)}{16} \leq \frac{2D\eta^2\gamma^2}{16} = L^2$$

(106)

where we used that $a = \frac{1}{\sqrt{8D^3}} < \gamma = \frac{4L}{\eta\sqrt{2D}}$. Similarly,

$$\|\nabla f(x; 2)\|^2 = \left\| \frac{\eta}{8} \left(\sum_{r=2,4,6,\ldots}^{2D} \phi_c' \left(v_{r-1}^\top x - v_r^\top x \right) (v_{r-1} - v_r) \right) \right\|^2 \leq \frac{2D\eta^2\gamma^2}{16} = L^2$$

(107)

Therefore, $f(\cdot; 1)$ and $f(\cdot; 2)$ are L-Lipschitz. Furthermore, since ϕ_c is 4-smooth,

$$\left| v_i^\top \nabla^2 f(x; 1) v_j \right| \leq \begin{cases} \frac{\eta}{2} & |i-j| \leq 1 \\ 0 & |i-j| > 1 \end{cases} \quad \text{and} \quad \left| v_i^\top \nabla^2 f(x; 2) v_j \right| \leq \begin{cases} \frac{\eta}{2} & |i-j| \leq 1 \\ 0 & |i-j| > 1 \end{cases}$$

(108)

therefore, the maximum eigenvalue of $\nabla^2 f(\cdot; 1)$ and $\nabla^2 f(\cdot; 2)$ is at most $\eta \leq H$. $\square$

Lemma 8. *For any $H, L \geq 0$, $D \geq 1$, and orthonormal $v_1, \ldots, v_{2D}$, and with η, γ, a, and c chosen as in (93), for any x such that $\left| v_r^\top x \right| \leq \frac{c}{2}$ for all $r > D$*

$$F(x) - \min_{x:\|x\| \leq 1} F(x) \geq \min \left\{ \frac{L}{32D}, \frac{H}{64D^2} \right\}$$

Proof. First, we upper bound $\min_{x:\|x\| \leq 1} F(x)$. Recalling that $a = \frac{1}{\sqrt{8D^3}}$, define

$$x^* = a \sum_{r=1}^{2D} (2D + 1 - r)v_r$$

(109)

$$\|x^*\|^2 = \frac{1}{8D^3} \left(\frac{2D(2D+1)(4D+1)}{6} \right) \leq 1$$

(110)

For this x^*, $v_{r-1}^\top x^* - v_r^\top x^* = v_{2D}^\top x^* = a$ and with our choice of parameters (93), $2c = a < \gamma$, so that $\phi_c'(a) = 2a$, thus

$$\nabla F(x^*) = \frac{\eta}{16} \left(-2av_1 + \phi_c' \left(v_{2D}^\top x^* \right) v_{2D} + \sum_{r=2}^{2D} \phi_c' \left(v_{r-1}^\top x^* - v_r^\top x^* \right) (v_{r-1} - v_r) \right)$$

(111)

thus,

$$\nabla F(x^*)^\top v_1 = -2a + \phi_c'(a) = 0$$

(112)

$$\nabla F(x^*)^\top v_r = -\phi_c'(a) + \phi_c'(a) = 0 \qquad \qquad 2 \leq r \leq 2D - 1$$

(113)

$$\nabla F(x^*)^\top v_{2D} = -\phi_c'(a) + \phi_c'(a) = 0$$

(114)

Since $\|x^*\| \leq 1$ and $\nabla F(x^*) = 0$, we conclude

$$\min_{x:\|x\| \leq 1} F(x) = F(x^*) = \frac{\eta}{16} \left(-2Da^2 - 4Dc^2 \right) = -\frac{\eta Da^2}{4} = -\frac{\eta}{32D^2}$$

(115)

Let $X_D = \left\{ x : \|x\| \leq 1, \left| v_r^\top x \right| \leq \frac{c}{2} \; \forall r > D \right\}$. We will now lower bound

$$\min_{x \in X_D} F(x) = \min_{x:\|x\| \leq 1} F(x) \quad \text{s.t.} \quad \left| v_r^\top x \right| \leq \frac{c}{2} \quad \forall r > D$$

(116)

21

Introducing dual variables $\lambda_{D+1}, ..., \lambda_{2D} \geq 0$, solving (116) amounts to finding an $x \in X_D$ and a set of non-negative λs such that $\nabla F(x) = -\sum_{r=D+1}^{2D} \lambda_r \, \text{sign} \left(v_r^\top x\right) v_r$ and such that $\lambda_r \left(v_r^\top x - \frac{c}{2}\right) = 0$ for each r. Let

$$x_D = \sum_{r=1}^{D+1} \left(a\left(D+1-r\right) + \frac{c}{2}\right) v_r, \quad \lambda_{D+1} = 2a, \quad \lambda_{D+2} = \cdots = \lambda_k = 0 \tag{117}$$

Since $a\left(D+1-r\right) + \frac{c}{2} < a\left(2D+1-r\right)$ for $r \leq D+1$ and $\|x^*\| \leq 1$ it follows that $\|x_D\| \leq 1$. Furthermore, since $v_{r-1}^\top x_D - v_r^\top x_D = a$ for $2 \leq r \leq D+1$ and $2c = a < \gamma$, the gradient

$$\nabla F(x_D)^\top v_1 = -2a + \phi_c'\left(a\right) = 0 \tag{118}$$

$$\nabla F(x_D)^\top v_r = -\phi_c'\left(a\right) + \phi_c'\left(a\right) = 0 \qquad\qquad 2 \leq r \leq D \tag{119}$$

$$\nabla F(x_D)^\top v_{D+1} = -\phi_c'\left(a\right) + \phi_c'\left(\frac{c}{2}\right) = -2a = -\lambda_{D+1} \tag{120}$$

$$\nabla F(x_D)^\top v_r = 0 = -\lambda_r \qquad\qquad D+2 \leq r \leq 2D \tag{121}$$

Therefore,

$$\min_{x \in X_D} F(x) = F(x_D) = \frac{\eta}{16}\left(-Da^2 - ac - 2Dc^2\right) = -\frac{\eta(3D+1)a^2}{32} = -\frac{\eta(3D+1)}{256D^3} \tag{122}$$

Combining (115) and (122) we have that

$$\min_{x \in X_D} F(x) - \min_{x:\|x\| \leq 1} F(x) = F(x_D) - F(x^*)$$

$$= \frac{\eta}{32D^2} - \frac{\eta(3D+1)}{256D^3} \geq \frac{\eta}{32D^2} - \frac{\eta}{64D^2} = \min\left\{\frac{L}{32D}, \frac{H}{64D^2}\right\} \tag{123}$$

$\square$

Lemma 9. *For any x such that $\left|v_r^\top x\right| \leq \frac{c}{2}$ for all $r \geq t$, and any $\beta \geq 0$ the function values, gradients, and proxs $f(x;1)$, $f(x;2)$, $\nabla f(x;1)$, $\nabla f(x;2)$, $\text{prox}_{f(\cdot,1)}(x,\beta)$, and $\text{prox}_{f(\cdot,2)}(x,\beta)$ are calculable using $\beta, x, v_1, \ldots, v_t$ only.*

Proof. Suppose that x is a point such that $\left|v_r^\top x\right| \leq \frac{c}{2}$ for all $r \geq t$, and $\beta \geq 0$. Therefore, $\phi_c\left(v_{r-1}^\top x - v_r^\top x\right) = 0$ for $r > t$ so

$$f(x;1) = \frac{\eta}{8}\left(-2av_1^\top x + \phi_c\left(v_{2D}^\top x\right) + \sum_{r=3,5,7,\ldots}^{2D-1} \phi_c\left(v_{r-1}^\top x - v_r^\top x\right)\right) \tag{124}$$

$$= \frac{\eta}{8}\left(-2av_1^\top x + \sum_{r=3,5,7,\ldots}^{t} \phi_c\left(v_{r-1}^\top x - v_r^\top x\right)\right) \tag{125}$$

$$f(x;2) = \frac{\eta}{8}\left(\sum_{r=2,4,6,\ldots}^{2D} \phi_c\left(v_{r-1}^\top x - v_r^\top x\right)\right) \tag{126}$$

$$= \frac{\eta}{8}\left(\sum_{r=2,4,6,\ldots}^{t} \phi_c\left(v_{r-1}^\top x - v_r^\top x\right)\right) \tag{127}$$

Thus both $f(x;1)$ and $f(x;2)$ can be calculated from $x, v_1, \ldots, v_t$ only. Similarly, $\phi'_c \left(v^\top_{r-1} x - v^\top_r x \right) = 0$ for $r > t$ so

$$\nabla f(x;1) = \frac{\eta}{8} \left(-2av_1 + \phi'_c \left(v^\top_{2D} x \right) v_{2D} + \sum_{r=3,5,7,\ldots}^{2D-1} \phi'_c \left(v^\top_{r-1} x - v^\top_r x \right) (v_{r-1} - v_r) \right) \quad (128)$$

$$= \frac{\eta}{8} \left(-2av_1 + \sum_{r=3,5,7,\ldots}^{t} \phi'_c \left(v^\top_{r-1} x - v^\top_r x \right) (v_{r-1} - v_r) \right) \quad (129)$$

$$\nabla f(x;2) = \frac{\eta}{8} \left(\sum_{r=2,4,6,\ldots}^{2D} \phi'_c \left(v^\top_{r-1} x - v^\top_r x \right) (v_{r-1} - v_r) \right) \quad (130)$$

$$= \frac{\eta}{8} \left(\sum_{r=2,4,6,\ldots}^{t} \phi'_c \left(v^\top_{r-1} x - v^\top_r x \right) (v_{r-1} - v_r) \right) \quad (131)$$

Thus $\nabla f(x;1)$ and $\nabla f(x;2)$ can also be calculated from $x, v_1, \ldots, v_t$ only.

Now, we consider the proxs at such a point x. Let $t' = t$ if t is odd, and $t' = t - 1$ if t is even. Let P be the projection operator onto $S = \text{Span}(v_1, \ldots, v_{t'})$ and let $P^\perp$ be the projection onto the orthogonal subspace, $S^\perp$. Then, since $f(x;1) = f(Px;1) + f(P^\perp x;1)$, we can decompose the prox:

$$\text{prox}_{f(\cdot;1)}(x, \beta)$$

$$= \arg\min_y f(y;1) + \frac{\beta}{2} \|y - x\|^2 \quad (132)$$

$$= \arg\min_{y_1 \in S, y_2 \in S^\perp} f(y_1;1) + f(y_2;1) + \frac{\beta}{2} \left(\|y_1 - Px\|^2 + \|y_2 - P^\perp x\|^2 \right) \quad (133)$$

$$= \arg\min_{y_1 \in S} \frac{\eta}{8} \left(-2av^\top_1 y_1 + \sum_{r=3,5,7,\ldots}^{t'} \phi_c \left(v^\top_{r-1} y_1 - v^\top_r y_1 \right) \right) + \frac{\beta}{2} \|y_1 - Px\|^2 \quad (134)$$

$$+ \arg\min_{y_2 \in S^\perp} \frac{\eta}{8} \left(\phi_c \left(v^\top_{2D} y_2 \right) + \sum_{r=t'+2,t'+4,\ldots}^{2D-1} \phi_c \left(v^\top_{r-1} y_2 - v^\top_r y_2 \right) \right) + \frac{\beta}{2} \|y_2 - P^\perp x\|^2 \quad (135)$$

$$= P^\perp x + \arg\min_{y_1 \in S} \frac{\eta}{8} \left(-2av^\top_1 y_1 + \sum_{r=3,5,7,\ldots}^{t'} \phi_c \left(v^\top_{r-1} y_1 - v^\top_r y_1 \right) \right) + \frac{\beta}{2} \|y_1 - Px\|^2 \quad (136)$$

Where we used that $\left| v^\top_r P^\perp x \right| = \left| v^\top_r x \right| \leq \frac{c}{2}$ for all $r > t'$, so setting $y_2 = P^\perp x$ achieves the minimum since every term in the expression is zero and function is non-negative. The vector $P^\perp x$ is calculable from $x, v_1, \ldots, v_{t'} \subseteq x, v_1, \ldots, v_t$, and similarly the second term is a minimization depends only on $\beta, x, v_1, \ldots, v_{t'} \subseteq \beta, x, v_1, \ldots, v_t$. A nearly identical argument shows that $\text{prox}_{f(\cdot;2)}(x, \beta)$ has the same property. $\qquad\square$

D Statistical term

Lemma 10. *For any $L, B > 0$, there exists a distribution $\mathcal{P}$, and an L-Lipschitz, 0-smooth function f defined on $[-B, B]$ such that the output $\hat{x}$ of any potentially randomized optimization algorithm which accesses a stochastic gradient or prox oracle at most N times satisfies*

$$\mathbb{E}_{\hat{X} \sim \mathcal{A}} \left[\mathbb{E}_{z \sim \mathcal{P}} \left[f(\hat{X}; z) \right] \right] - \min_{|x| \leq B} \mathbb{E}_{z \sim \mathcal{P}} \left[f(x; z) \right] \geq \frac{LB}{8\sqrt{N}}$$

Proof. Let $\epsilon > 0$ and $p \sim \text{Uniform} \{p_1, p_{-1}\}$ where $p_1 = \frac{1+\epsilon}{2}$ and $p_{-1} = \frac{1-\epsilon}{2}$. Define $\mathcal{P}_p$ as

$$\mathbb{P}_{\mathcal{P}_p} [Z = 1] = 1 - \mathbb{P}_{\mathcal{P}_p} [Z = -1] = p \quad (137)$$

Then, let $f(x; z) = zLx$, so $\mathbb{E}_{z \sim \mathcal{P}_p}[f(x; z)] = (2p - 1)Lx$. When $p = p_1$, $(2p - 1) > 0$ so the minimizer is $x = -B$ with value $-LB(2p - 1) = -LB\epsilon$, and when $p = p_{-1}$, $(2p - 1) < 0$ so the minimizer is $x = B$, also with value $-LB\epsilon$. Furthermore, if $p = p_1$ and $x \geq 0$ then it is at least $LB\epsilon$-suboptimal, and if $p = p_2$ and $x \leq 0$ then it is also at least $LB\epsilon$-suboptimal.

Now consider any deterministic optimization algorithm which accesses the gradient or prox oracle N times. Each gradient or prox oracle response can be simulated using a single $z \sim \mathcal{P}_p$, so the algorithm's output is $\hat{x} = \hat{x}(z_1, \ldots, z_N) \in [-B, B]$. Consider

$$\mathbb{E}_{p \sim \text{Uniform}\{p_1, p_{-1}\}, z \sim \mathcal{P}_p}[(2p - 1)L\hat{x}(z_1, \ldots, z_N) \,|\, z_1, \ldots, z_N]$$
$$\geq LB\epsilon \mathbb{P}_{p \sim \text{Uniform}\{p_1, p_{-1}\}, z \sim \mathcal{P}_p}[\text{sign}(\hat{x}(z_1, \ldots, z_N)) \neq \text{sign}(2p - 1) \,|\, z_1, \ldots, z_N] \quad (138)$$

Furthermore, the Bayes optimal estimate $\hat{x}$ of $\text{sign}(2p - 1)$ is

$$\hat{x}(z_1, \ldots, z_N) = \begin{cases} 1 & \frac{1}{N} \sum_{i=1}^N z_i \geq 0 \\ -1 & \frac{1}{N} \sum_{i=1}^N z_i < 0 \end{cases} \quad (139)$$

so

$$\mathbb{P}_{p \sim \text{Uniform}\{p_1, p_{-1}\}, z \sim \mathcal{P}_p}\left[\text{sign}(\hat{X}(z_1, \ldots, z_N)) \neq \text{sign}(2p - 1) \,\Big|\, z_1, \ldots, z_N\right]$$

$$\geq \mathbb{P}_{p \sim \text{Uniform}\{p_1, p_{-1}\}, z \sim \mathcal{P}_p}\left[\left|\frac{1}{N} \sum_{i=1}^N z_i - (2p - 1)\right| \geq \epsilon\right] \quad (140)$$

$$= \mathbb{P}_{z \sim \mathcal{P}_{p_{-1}}}\left[\left|\frac{1}{N} \sum_{i=1}^N z_i - \epsilon\right| \geq \epsilon\right] \quad (141)$$

This simply requires lower bounding the tail of the $\text{Binomial}\left(N, \frac{1-\epsilon}{2}\right)$ distribution. Using Theorem 2.1 in [23],

$$\mathbb{P}_{z \sim \mathcal{P}_{p_{-1}}}\left[\left|\frac{1}{N} \sum_{i=1}^N z_i - \epsilon\right| \geq \epsilon\right] \geq 1 - \Phi\left(\frac{\epsilon N}{\sqrt{N(1 + \epsilon)(1 - \epsilon)}}\right) = 1 - \Phi\left(\frac{\epsilon \sqrt{N}}{\sqrt{1 - \epsilon^2}}\right) \quad (142)$$

where Φ is the distribution function of the standard normal. Let $\epsilon = \frac{1}{2\sqrt{N}}$, then $\frac{\epsilon \sqrt{N}}{\sqrt{1-\epsilon^2}} < \frac{3}{5}$ and

$$\mathbb{P}_{z \sim \mathcal{P}_{p_{-1}}}\left[\left|\frac{1}{N} \sum_{i=1}^N z_i - \epsilon\right| \geq \epsilon\right] \geq 1 - \Phi\left(\frac{3}{5}\right) \geq \frac{1}{4} \quad (143)$$

Therefore, we conclude that

$$\mathbb{E}_{p \sim \text{Uniform}\{p_1, p_{-1}\}, z \sim \mathcal{P}_p}[(2p - 1)L\hat{x}(z_1, \ldots, z_N) \,|\, z_1, \ldots, z_N] \geq \frac{LB\epsilon}{4} = \frac{LB}{8\sqrt{N}} \quad (144)$$

Therefore, by Yao's minimax principle, for any randomized algorithm $\mathcal{A}$

$$\max_{p \in \{p_1, p_{-1}\}} \mathbb{E}_{\hat{X} \sim \mathcal{A}}\left[\mathbb{E}_{z \sim \mathcal{P}_p}\left[f(\hat{X}; z)\right] - \min_x \mathbb{E}_{z \sim \mathcal{P}_p}[f(x; z)]\right] \geq \frac{LB}{8\sqrt{N}} \quad (145)$$

$\square$

E Supplement to Section 4

E.1 Smoothed accelerated mini-batch SGD

Smoothed accelerated mini-batch SGD is the composition of two ingredients. First, we approximate the non-smooth f with a smooth surrogate, and then perform accelerated mini-batch SGD on the surrogate [9, 14]. In particular, we use the β-Moreau envelope $f^{(\beta)}$ of f:

$$f^{(\beta)}(x; z) = \inf_y f(y; z) + \frac{\beta}{2} \|y - x\|^2 \quad (146)$$

Since f is L-Lipschitz, $f^{(\beta)}$ has the following properties (Proposition 12.29 [5]):

1. $f^{(\beta)}$ is β-smooth

2. $\nabla f^{(\beta)}(x; z) = \beta(x - \mathrm{prox}_{f(\cdot; z)}(x, \beta))$

3. $f^{(\beta)}(x; z) \le f(x; z) \le f^{(\beta)}(x; z) + \frac{L^2}{2\beta}$ for all x

We use the prox oracle to execute A-MB-SGD on the L-Lipschitz and β-smooth $f^{(\beta)}$, with updates

$$w_t = \alpha y_t + (1 - \alpha)x_t \tag{147}$$

$$y_{t+1} = w_t - \frac{\eta}{M} \sum_{i=1}^{M} \beta \left(w_t - \mathrm{prox}_{f(\cdot; z_i)}(w_t, \beta) \right) \tag{148}$$

$$x_{t+1} = \alpha y_{t+1} + (1 - \alpha)x_t \tag{149}$$

The A-MB-SGD algorithm will converge on $f^{(\beta)}$ at a rate (see [9, 14])

$$\mathbb{E}\left[f(x_T; z) \right] - \min_x \mathbb{E}\left[f(x; z) \right] = O \left(\min \left\{ \frac{L}{\sqrt{T}}, \frac{\beta}{T^2} \right\} + \frac{L}{\sqrt{MT}} \right) \tag{150}$$

Choosing $\beta = \min \{LT, H\}$ the conclude

$$\mathbb{E}\left[f(x_T; z) \right] - \min_x \mathbb{E}\left[f(x; z) \right] \le \mathbb{E}\left[f^{(\beta)}(x_T; z) \right] + \frac{L}{2T} - \min_x \mathbb{E}\left[f^{(\beta)}(x; z) \right] \tag{151}$$

$$= O \left(\min \left\{ \frac{L}{\sqrt{T}}, \frac{\min\{LT, H\}}{T^2} \right\} + \frac{L}{\sqrt{MT}} + \frac{L}{T} \right) \tag{152}$$

$$= O \left(\min \left\{ \frac{L}{T}, \frac{H}{T^2} \right\} + \frac{L}{\sqrt{MT}} \right) \tag{153}$$

which matches the lower bound in Theorem 2.

E.2   Wait-and-collect accelerated mini-batch SGD

Algorithm 2 "Wait-and-collect" accelerated minibatch SGD

Initialize $\hat{x} = \tilde{x} = x_0 = 0$,, parameter α.
for $t = 1, 2, \ldots, T$ **do**
  if $\mathrm{mod}\,(t, 2\tau + 1) \le \tau$ **then**
    Query stochastic gradient at $\tilde{x}$.
    Update $x_t \leftarrow x_{t-1}, \tilde{g} = 0$.
  else if $\mathrm{mod}\,(t, 2\tau + 1) > \tau$ **and** $\mathrm{mod}\,(t, 2\tau + 1) \le 2\tau$ **then**
    Update $x_t \leftarrow x_{t-1}$.
    Receive noisy gradient $g_{t-1-\tau}$, update $\tilde{g} \leftarrow \tilde{g} + (1/\tau) * g_{t-1-\tau}$
  else if $\mathrm{mod}\,(t, 2\tau + 1) = 0$ **then**
    Update $x_t \leftarrow \tilde{x} - \eta \tilde{g}$.
    Update $\hat{x} \leftarrow \alpha \hat{x} + (1 - \alpha)x_t, \tilde{x} \leftarrow \alpha \hat{x} + (1 - \alpha)x_t$.
  end if
end for

E.3   Analysis of technical results in Section 4.4

Applying SVRG under intermittent synchronization graph   To apply SVRG method to solve stochastic convex optimization problems under intermittent synchronization graph. We adopt the approach by [15, 25], first we sample n instances $\{z_1, ..., z_n\}$ and solve a regularized empirical risk minimization problem based on $\{z_1, ..., z_n\}$:

$$\min_x \hat{F}_\lambda(x) := \frac{1}{n} \sum_{i=1}^{n} f(x; z_i) + \frac{\lambda}{2} \|x\|^2 , \tag{154}$$

where λ is the regularization parameter will specified later. We will apply SVRG algorithm on the intermittent synchronization graph to solve above empirical objective (154) to certain sub-optimality.

The SVRG method works in stages, at each stage, we first use n/KM communication rounds to calculate the full gradient of (154) at a reference point $\tilde{x}$, and then using a single chain to perform stochastic gradient updates, equipped with $\nabla \hat{F}(\tilde{w})$ to reduce the variance. We choose $\lambda \asymp L/(\sqrt{n}B)$, which will makes the objective (154) to be at least $L/(\sqrt{n}B)$-strongly convex, thus the condition number of (154) will be bounded by $O(H/(L/(\sqrt{n}B))) = O(H\sqrt{n}B/L)$. The SVRG analysis [13] requires the number of stochastic gradient updates to be scales as the condition number, so here we will use $O(H\sqrt{n}B/(LK))$ communication rounds to perform the stochastic updates, since one chain within each communication round has length K. Let $\hat{x}^* = \arg\min_x \hat{F}_\lambda(x)$, and let $\hat{x}_s$ to be the iterate after running the SVRG algorithm for s-stages. By the standard results of SVRG (Theorem 1 in [13]), we have

$$\mathbb{E}\left[\hat{F}_\lambda(\hat{x}_s)\right] - \hat{F}_\lambda(\hat{x}^*) \leq \left(\frac{1}{2}\right)^s.$$

By standard estimation-optimization error decomposition (e.g. Section 4 in [22]), we have

$$\mathbb{E}\left[F(\hat{x}_s)\right] - F(x^*) \leq 2\mathbb{E}\left[\hat{F}_\lambda(\hat{x}_s) - \hat{F}_\lambda(\hat{x}^*)\right] + \frac{\lambda B^2}{2} + O\left(\frac{L^2}{\lambda n}\right)$$
$$\leq \left(\frac{1}{2}\right)^s + O\left(\frac{LB}{\sqrt{n}}\right) = O\left(\frac{LB}{\sqrt{n}}\right), \tag{155}$$

given $s \asymp \log(n/(LB))$. Thus to implement SVRG successfully, we need to choose n such that the following two conditions are satisfied:

$$\frac{n}{KM} * s \leq T, \quad \text{and} \quad \frac{H\sqrt{n}B}{LK} * s \leq T.$$

Thus we know by choosing n below will satisfy above condition:

$$n \asymp \min\left\{\frac{K^2 T^2 L^2}{H^2 B^2 \log^2(MKT/L)}, \frac{MKT}{\log(MKT/L)}\right\},$$

substitute the scale of n to (155) we get

$$\mathbb{E}\left[F(\hat{x}_s)\right] - F(x^*) \leq O\left(\frac{HB^2}{KT}\log\left(\frac{MKT}{L}\right) + \frac{LB}{\sqrt{MKT}}\left(\log\left(\frac{MKT}{L}\right)\right)^{1/2}\right)$$
$$\leq O\left(\left(\frac{HB^2}{KT} + \frac{LB}{\sqrt{MKT}}\right)\log\left(\frac{MKT}{L}\right)\right),$$

and we obtain the desired result.