Estimation, Optimization, and Parallelism when Data is Sparse or Highly Varying

John C. Duchi   Michael I. Jordan   H. Brendan McMahan

November 10, 2013

Abstract

We study stochastic optimization problems when the *data* is sparse, which is in a sense dual to the current understanding of high-dimensional statistical learning and optimization. We highlight both the difficulties—in terms of increased sample complexity that sparse data necessitates—and the potential benefits, in terms of allowing parallelism and asynchrony in the design of algorithms. Concretely, we derive matching upper and lower bounds on the minimax rate for optimization and learning with sparse data, and we exhibit algorithms achieving these rates. We also show how leveraging sparsity leads to (still minimax optimal) parallel and asynchronous algorithms, providing experimental evidence complementing our theoretical results on several medium to large-scale learning tasks.

1 Introduction and problem setting

In this paper, we investigate stochastic optimization problems in which the *data* is sparse. Formally, let $\{F(\cdot;\xi), \xi \in \Xi\}$ be a collection of real-valued convex functions, each of whose domains contains the convex set $\mathcal{X} \subset \mathbb{R}^d$. For a probability distribution P on Ξ, we consider the following optimization problem:

$$\operatorname*{minimize}_{x \in \mathcal{X}} \; f(x) := \mathbb{E}[F(x;\xi)] = \int_{\Xi} F(x;\xi) dP(\xi). \tag{1}$$

By data sparsity, we mean that the sampled data ξ is sparse: samples ξ are assumed to lie in $\mathbb{R}^d$, and if we define the support $\operatorname{supp}(x)$ of a vector x to the set of indices of its non-zero components, we assume that

$$\operatorname{supp} \nabla F(x;\xi) \subset \operatorname{supp} \xi. \tag{2}$$

The sparsity condition (2) means that $F(x;\xi)$ does not "depend" on the values of x_j for indices j such that $\xi_j = 0$.[1] This type of data sparsity is prevalent in statistical optimization problems and machine learning applications, though in spite of its prevalence, study of such problems has been somewhat limited.

As a motivating example, consider a text classification problem: data $\xi \in \mathbb{R}^d$ represents words appearing in a document, and we wish to minimize a logistic loss $F(x;\xi) = \log(1 + \exp(\langle \xi, x \rangle))$ on the data (we encode the label implicitly with the sign of ξ). Such generalized linear models satisfy the sparsity condition (2), and while instances are of very high dimension, in any given instance,

[1] Formally, if we define π_ξ as the coordinate projection that zeros all indices j of its argument where $\xi_j = 0$, then $F(\pi_\xi(x);\xi) = F(x;\xi)$ for all x, ξ. This is implied by first order conditions for convexity [6, Chapter VI.2]

very few entries of ξ are non-zero [8]. From a modelling perspective, it thus makes sense to allow a *dense* predictor x: any non-zero entry of ξ is potentially relevant and important. In a sense, this is dual to the standard approaches to high-dimensional problems; one usually assumes that the data ξ may be dense, but there are only a few relevant features, and thus a parsimonious model x is desirous [2]. So while such sparse data problems are prevalent—natural language processing, information retrieval, and other large data settings all have significant data sparsity—they do not appear to have attracted as much study as their high-dimensional "duals" of dense data and sparse predictors.

In this paper, we investigate algorithms and their inherent limitations for solving problem (1) under natural conditions on the data generating distribution. Recent work in the optimization and machine learning communities has shown that data sparsity can be leveraged to develop parallel optimization algorithms [12, 13, 14], but the authors do not study the statistical effects of data sparsity. In recent work, Duchi et al. [4] and McMahan and Streeter [9] develop "adaptive" stochastic gradient algorithms designed to address problems in sparse data regimes (2). These algorithms exhibit excellent practical performance and have theoretical guarantees on their convergence, but it is not clear if they are optimal—in the sense that no algorithm can attain better statistical performance—or whether they can leverage parallel computing as in the papers [12, 14].

In this paper, we take a two-pronged approach. First, we investigate the fundamental limits of optimization and learning algorithms in sparse data regimes. In doing so, we derive lower bounds on the optimization error of *any* algorithm for problems of the form (1) with sparsity condition (2). These results have two main implications. They show that in some scenarios, learning with sparse data is quite difficult, as essentially each coordinate $j \in [d]$ can be relevant and must be optimized for. In spite of this seemingly negative result, we are also able to show that the AdaGrad algorithms of [4, 9] are optimal, and we show examples in which their dependence on the dimension d can be made exponentially better than standard gradient methods.

As the second facet of our two-pronged approach, we study how sparsity may be leveraged in parallel computing frameworks to give substantially faster algorithms that still achieve optimal sample complexity in terms of the number of samples ξ used. We develop two new algorithms, asynchronous dual averaging (AsyncDA) and asynchronous AdaGrad (AsyncAdaGrad), which allow asynchronous parallel solution of the problem (1) for general convex f and $\mathcal{X}$. Combining insights of Niu et al.'s Hogwild! [12] with a new analysis, we prove our algorithms can achieve linear speedup in the number of processors while maintaining optimal statistical guarantees. We also give experiments on text-classification and web-advertising tasks to illustrate the benefits of the new algorithms.

Notation For a convex function $x \mapsto f(x)$, we let $\partial f(x)$ denote its subgradient set at x (if f has two arguments, we say $\partial_x f(x, y)$ is the subgradient w.r.t. x). For a positive semi-definite matrix A, we let $\|\cdot\|_A$ be the (semi)norm defined by $\|v\|_A^2 := \langle v, Av \rangle$, where $\langle \cdot, \cdot \rangle$ is the standard inner product. We let $\mathbf{1}\{\cdot\}$ be the indicator function, which is 1 when its argument is true and 0 otherwise.

2 Minimax rates for sparse optimization

We begin our study of sparse optimization problems by establishing their fundamental statistical and optimization-theoretic properties. To do this, we derive bounds on the minimax convergence rate of any algorithm for such problems. Formally, let $\widehat{x}$ denote any estimator for a minimizer of the objective (1). We define the optimality gap ϵ_N for the estimator $\widehat{x}$ based on N samples $\xi^1, \ldots, \xi^N$ from the distribution P as

$$\epsilon_N(\widehat{x}, F, \mathcal{X}, P) := f(\widehat{x}) - \inf_{x \in \mathcal{X}} f(x) = \mathbb{E}_P\left[F(\widehat{x}; \xi)\right] - \inf_{x \in \mathcal{X}} \mathbb{E}_P\left[F(x; \xi)\right].$$

This quantity is a random variable, since $\widehat{x}$ is a random variable (it is a function of $\xi^1, \ldots, \xi^N$). To define the minimax error, we thus take expectations of the quantity ϵ_N, though we require a bit more than simply $\mathbb{E}[\epsilon_N]$. We let $\mathcal{P}$ denote a collection of probability distributions, and we consider a collection of loss functions F specified by a collection $\mathcal{F}$ of convex losses $F : \mathcal{X} \times \xi \to \mathbb{R}$. We can then define the minimax error for the family of losses $\mathcal{F}$ and distributions $\mathcal{P}$ as

$$\epsilon_N^*(\mathcal{X}, \mathcal{P}, \mathcal{F}) := \inf_{\widehat{x}} \sup_{P \in \mathcal{P}} \sup_{F \in \mathcal{F}} \mathbb{E}_P[\epsilon_N(\widehat{x}, F, \mathcal{X}, P)], \tag{3}$$

where the infimum is taken over all possible estimators (optimization schemes) $\widehat{x}$.

2.1 Minimax lower bounds

Let us now give a more precise characterization of the (natural) set of sparse optimization problems we consider to provide the lower bound. For the next proposition, we let $\mathcal{P}$ consist distributions supported on $\Xi = \{-1, 0, 1\}^d$, and we let $p_j := P(\xi_j \neq 0)$ be the marginal probability of appearance of feature j ($j \in \{1, \ldots, d\}$). For our class of functions, we set $\mathcal{F}$ to consist of functions F satisfying the sparsity condition (2) and with the additional constraint that for $g \in \partial_x F(x; \xi)$, we have that the jth coordinate $|g_j| \leq M_j$ for a constant $M_j < \infty$. We obtain

Proposition 1. *Let the conditions of the preceding paragraph hold. Let R be a constant such that $\mathcal{X} \supset [-R, R]^d$. Then*

$$\epsilon_N^*(\mathcal{X}, \mathcal{P}, \mathcal{F}) \geq \frac{1}{8} R \sum_{j=1}^{d} M_j \min\left\{p_j, \frac{\sqrt{p_j}}{\sqrt{N \log 3}}\right\}.$$

We provide the proof of Proposition 1 in Appendix A.1, providing a few remarks here. We begin by giving a corollary to Proposition 1 that follows when the data ξ obeys a type of power law: let $p_0 \in [0, 1]$, and assume that $P(\xi_j \neq 0) = p_0 j^{-\alpha}$. We have

Corollary 2. *Let $\alpha \geq 0$. Let the conditions of Proposition 1 hold with $M_j \equiv M$ for all j, and assume the power law condition $P(\xi_j \neq 0) = p_0 j^{-\alpha}$ on coordinate appearance probabilities. Then*

(1) If $d > (p_0 N)^{1/\alpha}$,

$$\epsilon_N^*(\mathcal{X}, \mathcal{P}, \mathcal{F}) \geq \frac{MR}{8}\left[\frac{2}{2-\alpha}\sqrt{\frac{p_0}{N}}\left((p_0 N)^{\frac{2-\alpha}{2\alpha}} - 1\right) + \frac{p_0}{1-\alpha}\left(d^{1-\alpha} - (p_0 N)^{\frac{1-\alpha}{\alpha}}\right)\right].$$

(2) If $d \leq (p_0 N)^{1/\alpha}$,

$$\epsilon_N^*(\mathcal{X}, \mathcal{P}, \mathcal{F}) \geq \frac{MR}{8}\sqrt{\frac{p_0}{N}}\left(\frac{1}{1-\alpha/2} d^{1-\frac{\alpha}{2}} - \frac{1}{1-\alpha/2}\right).$$

For simplicity assume that the features are not too extraordinarily sparse, say, that $\alpha \in [0, 2]$, and that number of samples is large enough that $d \le (p_0 N)^{1/\alpha}$. Then we find ourselves in regime (2) of Corollary 2, so that the lower bound on optimization error is of order

$$MR\sqrt{\frac{p_0}{N}} d^{1-\frac{\alpha}{2}} \text{ when } \alpha < 2, \quad MR\sqrt{\frac{p_0}{N}} \log d \text{ when } \alpha \to 2, \quad \text{and} \quad MR\sqrt{\frac{p_0}{N}} \text{ when } \alpha > 2. \tag{4}$$

These results beg the question of tightness: are they improvable? As we see presently, they are not.

2.2 Algorithms for attaining the minimax rate

The lower bounds specified by Proposition 1 and the subsequent specializations are sharp, meaning that they are unimprovable by more than constant factors. To show this, we review a few stochastic gradient algorithms. We first recall stochastic gradient descent, after which we review the dual averaging methods and an extension of both.

We begin with stochastic gradient descent (SGD): for this algorithm, we repeatedly sample $\xi \sim P$, compute $g \in \partial_x F(x; \xi)$, then perform the update $x \leftarrow \Pi_{\mathcal{X}}(x - \eta g)$, where η is a stepsize parameter and $\Pi_{\mathcal{X}}$ denotes Euclidean projection onto $\mathcal{X}$. Then standard analyses of stochastic gradient descent (e.g. [10]) show that after N samples ξ, in our setting the SGD estimator $\widehat{x}(N)$ satisfies

$$\mathbb{E}[f(\widehat{x}(N))] - \inf_{x \in \mathcal{X}} f(x) \le \mathcal{O}(1) \frac{R_2 M \sqrt{\sum_{j=1}^d p_j}}{\sqrt{N}}, \tag{5}$$

where R_2 denotes the ℓ_2-radius of $\mathcal{X}$. Dual averaging, due to Nesterov [11] and referred to as "follow the regularized leader" in the machine learning literature (see, e.g., the survey article by Hazan [5]) is somewhat more complex. In dual averaging, one again samples $g \in \partial_x F(x; \xi)$, but instead of updating the parameter vector x one updates a dual vector z by $z \leftarrow z + g$, then computes

$$x \leftarrow \operatorname*{argmin}_{x \in \mathcal{X}} \left\{ \langle z, x \rangle + \frac{1}{\eta} \psi(x) \right\},$$

where $\psi(x)$ is a strongly convex function defined over $\mathcal{X}$ (often one takes $\psi(x) = \frac{1}{2}\|x\|_2^2$). The dual averaging algorithm, as we shall see, is somewhat more natural in asynchronous and parallel computing environments, and it enjoys the same type of convergence guarantees (5) as SGD.

The AdaGrad algorithm [4, 9] is a slightly more complicated extension of the preceding stochastic gradient methods. It maintains a diagonal matrix S, where upon receiving a new sample ξ, AdaGrad performs the following: it computes $g \in \partial_x F(x; \xi)$, then updates

$$S_j \leftarrow S_j + g_j^2 \text{ for } j \in [d].$$

Depending on whether the dual averaging or stochastic gradient descent (SGD) variant is being used, AdaGrad performs one of two updates. In the dual averaging case, it maintains the dual vector z, which is updated by $z \leftarrow z + g$; in the SGD case, the parameter x is maintained. The updates for the two cases are then

$$x \leftarrow \operatorname*{argmin}_{x' \in \mathcal{X}} \left\{ \langle g, x' \rangle + \frac{1}{2\eta} \left\langle x' - x, S^{\frac{1}{2}}(x' - x) \right\rangle \right\}$$

for stochastic gradient descent and

$$x \leftarrow \operatorname*{argmin}_{x' \in \mathcal{X}} \left\{ \left\langle z, x' \right\rangle + \frac{1}{2\eta} \left\langle x', S^{\frac{1}{2}} x' \right\rangle \right\}$$

for dual averaging, where η is a stepsize. Then appropriate choice of η shows that after N samples ξ, the averaged parameter $\widehat{x}(N)$ AdaGrad returns satisfies

$$\mathbb{E}[f(\widehat{x}(N))] - \inf_{x \in \mathcal{X}} f(x) \leq \mathcal{O}(1) \frac{R_\infty M}{\sqrt{N}} \sum_{j=1}^{d} \sqrt{p_j}, \tag{6}$$

where R_∞ denotes the ℓ_∞-radius of $\mathcal{X}$ (e.g. [4, Section 1.3 and Theorem 5], where one takes $\eta \approx R_\infty$). By inspection, the AdaGrad rate (6) matches the lower bound in Proposition 1 and is thus optimal. It is interesting to note, though, that in the power law setting of Corollary 2 (recall the error order (4)), a calculation shows that the multiplier for the SGD guarantee (5) becomes $R_\infty \sqrt{d} \max\{d^{(1-\alpha)/2}, 1\}$, while AdaGrad attains rate at worst $R_\infty \max\{d^{1-\alpha/2}, \log d\}$ (by evaluation of $\sum_j \sqrt{p_j}$). Thus for $\alpha > 1$, the AdaGrad rate is no worse, and for $\alpha \geq 2$, is more than $\sqrt{d}/\log d$ better than SGD—an exponential improvement in the dimension.

3 Parallel and asynchronous optimization with sparsity

As we note in the introduction, recent works [12, 14] have suggested that sparsity can yield benefits in our ability to *parallelize* stochastic gradient-type algorithms. Given the optimality of AdaGrad-type algorithms, it is natural to focus on their parallelization in the hope that we can leverage their ability to "adapt" to sparsity in the data. To provide the setting for our further algorithms, we first revisit Niu et al.'s Hogwild!.

The Hogwild! algorithm of Niu et al. [12] is an asynchronous (parallelized) stochastic gradient algorithm that proceeds as follows. To apply Hogwild!, we must assume the domain $\mathcal{X}$ in problem (1) is a product space, meaning that it decomposes as $\mathcal{X} = \mathcal{X}_1 \times \cdots \times \mathcal{X}_d$, where $\mathcal{X}_j \subset \mathbb{R}$. Fix a stepsize $\eta > 0$. Then a pool of processors, each running independently, performs the following updates asynchronously to a centralized vector x:

1. Sample $\xi \sim P$
2. Read x and compute $g \in \partial_x F(x; \xi)$
3. For each j s.t. $g_j \neq 0$, update $x_j \leftarrow \Pi_{\mathcal{X}_j}(x_j - \eta g_j)$

Here $\Pi_{\mathcal{X}_j}$ denotes projection onto the jth coordinate of the domain $\mathcal{X}$. The key of Hogwild! is that in step 2, the parameter x at which g is calculated may be somewhat inconsistent—it may have received partial gradient updates from many processors—though for appropriate problems, this inconsistency is negligible. Indeed, Niu et al. [12] show a linear speedup in optimization time as the number of independent processors grow; they show this empirically in many scenarios, providing a proof under the somewhat restrictive assumption that there is at most one non-zero entry in any gradient g.

3.1 Asynchronous dual averaging

One of the weaknesses of Hogwild! is that, as written it appears to only be applicable to problems for which the domain $\mathcal{X}$ is a product space, and the known analysis assumes that $\|g\|_0 = 1$ for all gradients g. In effort to alleviate these difficulties, we now develop and present our asynchronous dual averaging algorithm, AsyncDA. In AsyncDA, instead of asynchronously updating a centralized parameter vector x, we maintain a centralized dual vector z. A pool of processors performs asynchronous additive updates to z, where each processor repeatedly performs the following updates:

1. Read z and compute $x := \mathrm{argmin}_{x \in \mathcal{X}} \left\{ \langle z, x \rangle + \frac{1}{\eta}\psi(x) \right\}$ // *Implicitly increment "time" counter t and let $x(t) = x$*
2. Sample $\xi \sim P$ and let $g \in \partial_x F(x; \xi)$ // *Let $g(t) = g$.*
3. For $j \in [d]$ such that $g_j \neq 0$, update $z_j \leftarrow z_j + g_j$

Because the actual computation of the vector x in asynchronous dual averaging (AsyncDA) is performed locally on each processor in step 1 of the algorithm, the algorithm can be *executed* with any proximal function ψ and domain $\mathcal{X}$. The only communication point between any of the processors is the addition operation in step 3. As noted by Niu et al. [12], this operation can often be performed atomically on modern processors.

In our analysis of AsyncDA, and in our subsequent analysis of the adaptive methods, we require a measurement of time elapsed. With that in mind, we let t denote a time index that exists (roughly) behind-the-scenes. We let $x(t)$ denote the vector $x \in \mathcal{X}$ computed in the tth step 1 of the AsyncDA algorithm, that is, whichever is the tth x actually computed by any of the processors. We note that this quantity exists and is recoverable from the algorithm, and that it is possible to track the running sum $\sum_{\tau=1}^{t} x(\tau)$.

Additionally, we require two assumptions encapsulating the conditions underlying our analysis.

Assumption A. *There is an upper bound m on the delay of any processor. In addition, for each $j \in [d]$ there is a constant $p_j \in [0, 1]$ such that $P(\xi_j \neq 0) \leq p_j$.*

We also require an assumption about the continuity (Lipschitzian) properties of the loss functions being minimized; the assumption amounts to a second moment constraint on the sub-gradients of the instantaneous F along with a rough measure of the sparsity of the gradients.

Assumption B. *There exist constants M and $(M_j)_{j=1}^{d}$ such that the following bounds hold for all $x \in \mathcal{X}$: $\mathbb{E}[\|\partial_x F(x; \xi)\|_2^2] \leq \mathsf{M}^2$ and for each $j \in [d]$ we have $\mathbb{E}[|\partial_{x_j} F(x; \xi)|] \leq p_j M_j$.*

With these definitions, we have the following theorem, which captures the convergence behavior of AsyncDA under the assumption that $\mathcal{X}$ is a Cartesian product, meaning that $\mathcal{X} = \mathcal{X}_1 \times \cdots \times \mathcal{X}_d$, where $\mathcal{X}_j \subset \mathbb{R}$, and that $\psi(x) = \frac{1}{2}\|x\|_2^2$. Note the algorithm itself can still be efficiently parallelized for more general convex $\mathcal{X}$, even if the theorem does not apply.

Theorem 3. *Let Assumptions A and B and the conditions in the preceding paragraph hold. Then*

$$\mathbb{E}\left[\sum_{t=1}^{T} F(x(t); \xi^t) - F(x^*; \xi^t)\right] \leq \frac{1}{2\eta}\|x^*\|_2^2 + \frac{\eta}{2}T\mathsf{M}^2 + \eta T m \sum_{j=1}^{d} p_j^2 M_j^2.$$

We provide the proof of Theorem 3 in Appendix B.

As stated, the theorem is somewhat unwieldy, so we provide a corollary and a few remarks to explain and simplify the result. Under a more stringent condition that $|\partial_{x_j} F(x;\xi)| \le M_j$, Assumption A implies $\mathbb{E}[\|\partial_x F(x;\xi)\|_2^2] \le \sum_{j=1}^d p_j M_j^2$. Thus, without loss of generality for the remainder of this section we take $\mathsf{M}^2 = \sum_{j=1}^d p_j M_j^2$, which serves as an upper bound on the Lipschitz continuity constant of the objective function f. We then obtain the following corollary.

Corollary 4. *Define $\widehat{x}(T) = \frac{1}{T}\sum_{t=1}^T x(t)$, and set $\eta = \|x^*\|_2 / \mathsf{M}\sqrt{T}$. Then*

$$\mathbb{E}[f(\widehat{x}(T)) - f(x^*)] \le \frac{\mathsf{M}\,\|x^*\|_2}{\sqrt{T}} + m\frac{\|x^*\|_2}{2\mathsf{M}\sqrt{T}}\sum_{j=1}^d p_j^2 M_j^2$$

Corollary 4 is almost immediate. To see the result, note that since ξ^t is independent of $x(t)$, we have $\mathbb{E}[F(x(t);\xi^t) \mid x(t)] = f(x(t))$; applying Jensen's inequality to $f(\widehat{x}(T))$ and performing an algebraic manipulation give the corollary.

If the data is suitably "sparse," meaning that $p_j \le 1/m$ (which may also occur if the data is of relatively high variance in Assumption B) the bound in Corollary 4 simplifies to

$$\mathbb{E}[f(\widehat{x}(T)) - f(x^*)] \le \frac{3}{2}\frac{\mathsf{M}\,\|x^*\|_2}{\sqrt{T}} = \frac{3}{2}\frac{\sqrt{\sum_{j=1}^d p_j M_j^2}\,\|x^*\|_2}{\sqrt{T}} \tag{7}$$

which is the convergence rate of stochastic gradient descent (and dual averaging) even in non-asynchronous situations (5). In non-sparse cases, setting $\eta \propto \|x^*\|_2 / \sqrt{m\mathsf{M}^2 T}$ in Theorem 3 recovers the bound

$$\mathbb{E}[f(\widehat{x}(T)) - f(x^*)] \le \mathcal{O}(1)\sqrt{m}\cdot\frac{\mathsf{M}\,\|x^*\|_2}{\sqrt{T}}.$$

The convergence guarantee (7) shows that after T timesteps, we have error scaling $1/\sqrt{T}$; however, if we have k processors, then updates can occur roughly k times as quickly, as all updates are asynchronous. Thus in time scaling as n/k, we can evaluate n gradient samples: a linear speedup.

3.2 Asynchronous AdaGrad

We now turn to extending AdaGrad to asynchronous settings, developing AsyncAdaGrad (asynchronous AdaGrad). As in the AsyncDA algorithm, AsyncAdaGrad maintains a shared dual vector z among the processors, which is the sum of gradients observed; AsyncAdaGrad also maintains the matrix S, which is the diagonal sum of squares of gradient entries (recall Section 2.2). The matrix S is initialized as diag(δ^2), where $\delta_j \ge 0$ is an initial value. Each processor asynchronously performs the following iterations:

1. Read S and z and set $G = S^{\frac{1}{2}}$. Compute $x := \text{argmin}_{x\in\mathcal{X}}\left\{\langle z, x\rangle + \frac{1}{2\eta}\langle x, Gx\rangle\right\}$ // *Implicitly increment "time" counter t and let $x(t) = x$, $S(t) = S$*

2. Sample $\xi \sim P$ and let $g \in \partial F(x;\xi)$

3. For $j \in [d]$ such that $g_j \ne 0$, update $S_j \leftarrow S_j + g_j^2$ and $z_j \leftarrow z_j + g_j$

As in the description of AsyncDA, we note that $x(t)$ is the vector $x \in \mathcal{X}$ computed in the tth "step" of the algorithm (step 1), and similarly associate ξ^t with $x(t)$.

To analyze AsyncAdaGrad, we make a somewhat stronger assumption on the sparsity properties of the losses F than Assumption B.

Assumption C. *There exist constants $(M_j)_{j=1}^d$ such that for any $x \in \mathcal{X}$ and $\xi \in \Xi$, we have $\mathbb{E}[(\partial_{x_j} F(x;\xi))^2 \mid \xi_j \neq 0] \le M_j^2$.*

Indeed, taking $\mathsf{M}^2 = \sum_j p_j M_j^2$ shows that Assumption C implies Assumption B with specific constants. We then have the following convergence result, whose proof we provide defer to Appendix C.

Theorem 5. *In addition to the conditions of Theorem 3, let Assumption C hold. Assume that $\delta^2 \ge M_j^2 m$ for all j and that $\mathcal{X} \subset [-R_\infty, R_\infty]^d$. Then*

$$\sum_{t=1}^T \mathbb{E}\left[F(x(t);\xi^t) - F(x^*;\xi^t)\right] \\ \le \sum_{j=1}^d \min\left\{\frac{1}{\eta} R_\infty^2 \mathbb{E}\left[\left(\delta^2 + \sum_{t=1}^T g_j(t)^2\right)^{\frac{1}{2}}\right] + \eta \mathbb{E}\left[\left(\sum_{t=1}^T g_j(t)^2\right)^{\frac{1}{2}}\right](1 + p_j m), M_j R_\infty p_j T\right\}.$$

We can also relax the condition on the initial constant diagonal term δ slightly, which gives a qualitatively similar result (see Appendix C.3).

Corollary 6. *Under the conditions of Theorem 5, assume that $\delta^2 \ge M_j^2 \min\{m, 6\max\{\log T, mp_j\}\}$ for all j. Then*

$$\sum_{t=1}^T \mathbb{E}\left[F(x(t);\xi^t) - F(x^*;\xi^t)\right] \\ \le \sum_{j=1}^d \min\left\{\frac{1}{\eta} R_\infty^2 \mathbb{E}\left[\left(\delta^2 + \sum_{t=1}^T g_j(t)^2\right)^{\frac{1}{2}}\right] + \frac{3}{2}\eta \mathbb{E}\left[\sum_{t=1}^T g_j(t)^2\right]^{\frac{1}{2}}(1 + p_j m), M_j R_\infty p_j T\right\}.$$

It is natural to ask in which situations the bound Theorem 5 and Corollary 6 provides is optimal. We note that, as in the case with Theorem 3, we may take an expectation with respect to ξ^t and obtain a convergence rate for $f(\widehat{x}(T)) - f(x^*)$, where $\widehat{x}(T) = \frac{1}{T}\sum_{t=1}^T x(t)$. By Jensen's inequality, we have for any δ that

$$\mathbb{E}\left[\left(\delta^2 + \sum_{t=1}^T g_j(t)^2\right)^{\frac{1}{2}}\right] \le \left(\delta^2 + \sum_{t=1}^T \mathbb{E}[g_j(t)^2]\right)^{\frac{1}{2}} \le \sqrt{\delta^2 + T p_j M_j^2}.$$

For interpretation, let us now make a few assumptions on the probabilities p_j. If we assume that $p_j \le c/m$ for a universal (numerical) constant c, then Theorem 5 guarantees that

$$\mathbb{E}[f(\widehat{x}(T)) - f(x^*)] \le \mathcal{O}(1)\left[\frac{1}{\eta}R_\infty^2 + \eta\right] \sum_{j=1}^d M_j \min\left\{\frac{\sqrt{\log(T)/T + p_j}}{\sqrt{T}}, p_j\right\}, \tag{8}$$

which is the convergence rate of AdaGrad except for a small factor of $\min\{\sqrt{\log T}/T, p_j\}$ in addition to the usual $\sqrt{p_j/T}$ rate. In particular, optimizing by choosing $\eta = R_\infty$, and assuming $p_j \gtrsim \frac{1}{T}\log T$, we have convergence guarantee

$$\mathbb{E}[f(\widehat{x}(T)) - f(x^*)] \le \mathcal{O}(1) R_\infty \sum_{j=1}^{d} M_j \min\left\{\frac{\sqrt{p_j}}{\sqrt{T}}, p_j\right\},$$

which is minimax-optimal by Proposition 1. In fact, however, the bounds of Theorem 5 and Corollary 6 are somewhat stronger: they provide bounds using the *expectation* of the squared gradients $g_j(t)$ rather than the maximal value M_j, though the bounds are perhaps clearer in the form (8). We note also that our analysis applies to more adversarial settings than stochastic optimization (e.g. to online convex optimization [5]). Specifically, an adversary may choose an arbitrary sequence of functions subject to the random data sparsity constraint (2), and our results provide an expected regret bound, which is strictly stronger than the stochastic convergence guarantees provided (and guarantees high-probability convergence in stochastic settings [3]). Moreover, our comments in Section 2 about the relative optimality of AdaGrad versus standard gradient methods apply. When the data is sparse, we indeed should use asynchronous algorithms, but using adaptive methods yields even more improvement than simple gradient-based methods.

4 Experiments

In this section, we give experimental validation of our theoretical results on AsyncAdaGrad and AsyncDA, giving results on two datasets selected for their high-dimensional sparsity.[2]

4.1 Malicious URL detection

For our first set of experiments, we consider the speedup attainable by applying AsyncAdaGrad and AsyncDA, investigating the performance of each algorithm on a malicious URL prediction task [7]. The dataset in this case consists of an anonymized collection of URLs labeled as malicious (e.g. spam, phishing, etc.) or benign over a span of 120 days. The data in this case consists of $2.4 \cdot 10^6$ examples with dimension $d = 3.2 \cdot 10^6$ (sparse) features. We perform several experiments, randomly dividing the dataset into $1.2 \cdot 10^6$ training and test samples for each experiment.

In Figure 1 and we compare the performance of AsyncAdaGrad and AsyncDA after doing after single pass through the training dataset. (For each algorithm, we choose the stepsize η for optimal training set performance.) We perform the experiments on a single machine running Ubuntu Linux with 6 cores (with two-way hyperthreading) and 32Gb of RAM. From the left-most plot in Fig. 1, we see that up to 6 processors, both AsyncDA and AsyncAdaGrad enjoy the expected linear speedup, and from 6 to 12, they continue to enjoy a speedup that is linear in the number of processors though at a lesser slope (this is the effect of hyperthreading). For more than 12 processors, there is no further benefit to parallelism on this machine.

The two right plots in Figure 1 plot performance of the different methods (with standard errors) versus the number of worker threads used. Both are essentially flat; increasing the amount of parallelism does nothing to the average training loss or the test error rate for either method. It is clear, however, that for this dataset, the adaptive AsyncAdaGrad algorithm provides substantial performance benefits over AsyncDA.

[2] We also performed experiments using Hogwild! instead of AsyncDA; the results are similar.

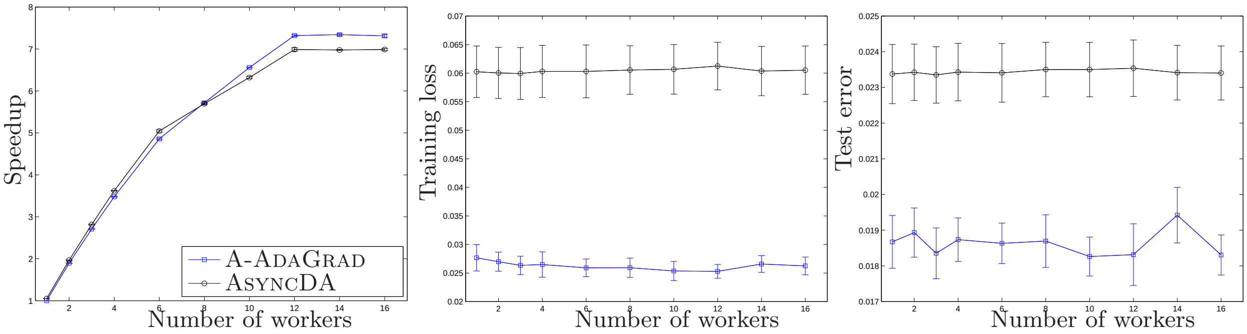

Figure 1. Experiments with URL data. Left: speedup relative to 1 processor. Middle: training dataset loss versus number of processors. Right: test set error rate versus number of processors. A-AdaGrad abbreviates AsyncAdaGrad.

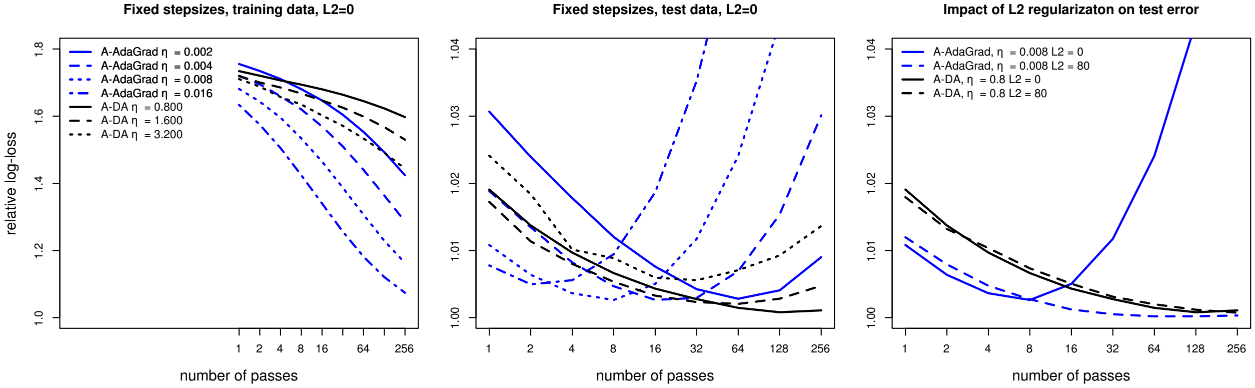

Figure 2. Relative accuracy for various stepsize choices on an click-through rate prediction dataset.

4.2 Click-through-rate prediction experiments

We also experimented on a proprietary datasets consisting of search ad impressions. Each example corresponds to showing a search-engine user a particular text ad in response to a query string. From this, we construct a very sparse feature vector based on the text of the ad displayed and the query string (no user-specific data was used). The target label is 1 if the user clicked the ad, and -1 otherwise. We fit logistic regression models using both AsyncDA and AsyncAdaGrad. Rather than running few experiments on a large dataset, we ran extensive experiments on a moderate-sized dataset (about 10^7 examples, split evenly between training and testing). This allowed us to thoroughly investigate the impact of the stepsize η, the number of training passes,[3] and L_2 regularization on accuracy. Section 4.1 shows that AsyncAdaGrad achieves a similar speedup to AsyncDA, so for these experiments we used 32 threads on 16 core machines for each run.

On this dataset, AsyncAdaGrad typically achieves an effective *additional* speedup over AsyncDA of 4× or more. That is, to reach a given level of accuracy, AsyncDA generally needs four times

[3] Here "number of passes" more precisely means the expected number of times each example in the dataset is trained on. That is, each worker thread randomly selects a training example from the dataset for each update, and we continued making updates until (dataset size) × (number of passes) updates have been processed.

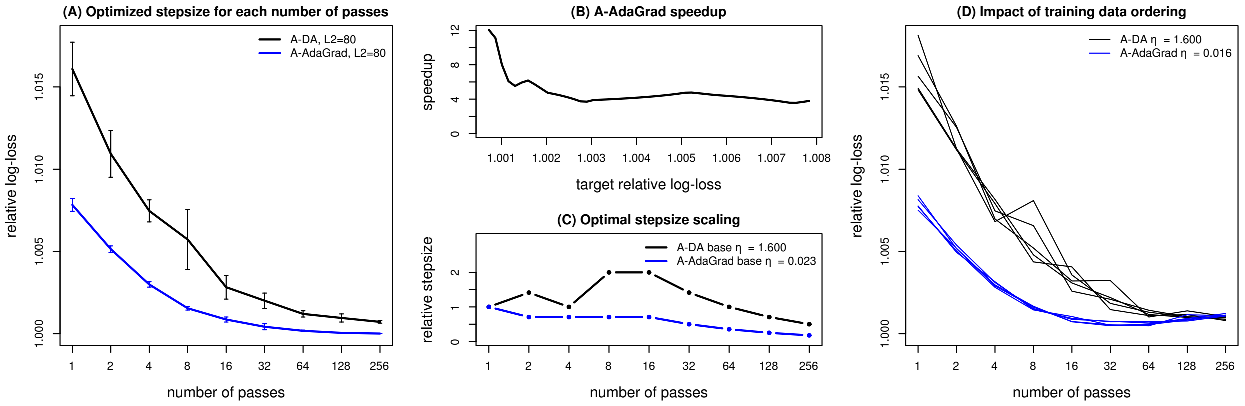

Figure 3. (A) Relative test-set log-loss for AsyncDA and AsyncAdaGrad, choosing the best stepsize (within a factor of about 1.4×) individually for each number of passes. (B) Effective speedup for AsyncAdaGrad. (C) The best stepsize η, expressed as a scaling factor on the stepsize used for one pass. (D) Five runs with different random seeds for each algorithm (with $L_2 = 80$).

as many effective passes over the dataset. We measure accuracy with log-loss (the logistic loss) averaged over 5 runs using different random seeds (which control the order in which the algorithms sample examples during training). We report relative values in Figures 2 and 3, that is, the ratio of the mean loss for the given datapoint to the lowest (best) mean loss obtained. Our results are not particularly sensitive to the choice of relative log-loss as the metric of interest; we also considered AUC (the area under the ROC curve) and observed similar results.

Figure 2 (A–B) shows relative log-loss as a function of the number of training passes for various stepsizes. Without regularization, we see that AsyncAdaGrad is prone to overfitting: it achieves significantly higher accuracy on the training data 2(A), but unless the step-size is tuned carefully to the number of passes, it will overfit and predict poorly on test data 2(B). Fortunately, the addition of L_2 regularization largely solves this problem. Figure 2(C) shows that adding an L_2 penalty of 80 has very little impact on Hogwild!, but effectively prevents the overfitting of AsyncAdaGrad.[4]

Fixing $L_2 = 80$, for each number of passes and for each algorithm, we varied the stepsize η over a multiplicative grid with resolution $\sqrt{2}$. Figure 3 reports the results obtained by selecting the best stepsize in terms of test set log-loss for each number of passes. Figure 3(A) shows relative log-loss of the best stepsize for each algorithm; 3(B) is based on the same data, but considers on the x-axis relative losses between the 256-pass AsyncDA loss (about 1.001) and the 1-pass AsyncAdaGrad loss (about 1.008). For these values, we can take the linear interpolation shown in 3(A), and look at the ratio of the number of passes the two algorithms needed to achieve a fixed relative log-loss. This gives an estimate of the relative speedup obtained by using AsyncAdaGrad over a range of different target accuracies; speedups range from 3.6× to 12×. Figure 3(C) shows the optimal stepsizes as a function of the best setting for one pass. The optimal stepsize decreases moderately for AsyncAdaGrad, but are somewhat noisy for Hogwild!.

It is interesting to note that AsyncAdaGrad's accuracy is largely independent of the ordering of the training data, while Hogwild! shows significant variability. This can be seen both in the

[4] For both algorithms, this is accomplished by adding the term $\eta 80 \left\|x\right\|_2^2$ to the ψ function. We could have achieved slightly better results for AsyncAdaGrad by varying the L_2 penalty with the number of passes (with more passes benefiting from more regularization).

error bars on Figure 3(A), and explicitly in Figure 3(D), where we plot one line for each of the 5 random seeds used. Thus, while on the one hand Hogwild! requires somewhat less tuning of the stepsize and L_2 parameter to control overfitting, tuning AsyncAdaGrad is much easier because of the predictable response.

References

[1] P. Auer and C. Gentile. Adaptive and self-confident online learning algorithms. In *Proceedings of the Thirteenth Annual Conference on Computational Learning Theory*, 2000.

[2] P. Bühlmann and S. van de Geer. *Statistics for High-Dimensional Data: Methods, Theory and Applications*. Springer, 2011.

[3] N. Cesa-Bianchi, A. Conconi, and C. Gentile. On the generalization ability of on-line learning algorithms. *IEEE Transactions on Information Theory*, 50(9):2050–2057, September 2004.

[4] J. C. Duchi, E. Hazan, and Y. Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12:2121–2159, 2011.

[5] E. Hazan. The convex optimization approach to regret minimization. In *Optimization for Machine Learning*, chapter 10. MIT Press, 2012.

[6] J. Hiriart-Urruty and C. Lemaréchal. *Convex Analysis and Minimization Algorithms I & II*. Springer, New York, 1996.

[7] J. Ma, L. K. Saul, S. Savage, and G. M. Voelker. Identifying malicious urls: An application of large-scale online learning. In *Proceedings of the 26th International Conference on Machine Learning*, 2009.

[8] C. Manning and H. Schütze. *Foundations of Statistical Natural Language Processing*. MIT Press, 1999.

[9] B. McMahan and M. Streeter. Adaptive bound optimization for online convex optimization. In *Proceedings of the Twenty Third Annual Conference on Computational Learning Theory*, 2010.

[10] A. Nemirovski, A. Juditsky, G. Lan, and A. Shapiro. Robust stochastic approximation approach to stochastic programming. *SIAM Journal on Optimization*, 19(4):1574–1609, 2009.

[11] Y. Nesterov. Primal-dual subgradient methods for convex problems. *Mathematical Programming*, 120(1):261–283, 2009.

[12] F. Niu, B. Recht, C. Ré, and S. Wright. Hogwild: a lock-free approach to parallelizing stochastic gradient descent. In *Advances in Neural Information Processing Systems 24*, 2011.

[13] P. Richtárik and M. Takáč. Parallel coordinate descent methods for big data optimization. *arXiv:1212.0873 [math.OC]*, 2012. URL `http://arxiv.org/abs/1212.0873`.

[14] M. Takáč, A. Bijral, P. Richtárik, and N. Srebro. Mini-batch primal and dual methods for SVMs. In *Proceedings of the 30th International Conference on Machine Learning*, 2013.

A Proof of Proposition 1 and related results

In this section we provide the proofs of Proposition 1 and its subsequent corollaries.

A.1 Proof of Proposition 1

Our proof proceeds in a few steps: we first define our family of loss functions, after which we perform an essentially standard reduction of the estimation (optimization) problem to testing. Following this step, we carefully lower bound the probabilities of error in our multiple hypothesis testing problem (in a manner similar to Assouad's lemma) to obtain the desired statement of the proposition.

Defining the loss functions We begin by describing the family of loss functions we use to prove the result. Given $\xi \in \{-1, 0, 1\}^d$, let

$$F(x; \xi) := M \sum_{j=1}^{d} |\xi_j| |x_j - R\xi_j|.$$

By inspection F satisfies the conditions of the proposition. Letting $p_j = P(\xi_j \neq 0)$, $p_j^+ = P(\xi_j = 1)$, and $p_j^- = P(\xi_j = 1)$, we thus find that

$$f(x) = M \sum_{j=1}^{d} \left(p_j^+ |x_j - R| + p_j^- |x_j + R| \right),$$

so the objective f behaves like a weighted 1-norm type of quantity and its minimizer is a multi-dimensional median.

From estimation to testing Now we proceed through a reduction of estimation to testing. Fix $\delta_j > 0$ for $j \in \{1, \ldots, d\}$ (we optimize these choices later). Let $\mathcal{V} = \{-1, 1\}^d$, and for a fixed $v \in \mathcal{V}$ let P_v be the distribution specified by

$$P_v(\xi_j \neq 0) = p_j, \quad P_v(\xi_j = 1 \mid \xi_j \neq 0) = \frac{1 + \delta_j v_j}{2}, \quad P_v(\xi_j = -1 \mid \xi_j \neq 0) = \frac{1 - \delta_j v_j}{2}. \tag{9}$$

With this choice of distribution, we claim that for any estimator $\widehat{x}$,

$$\sup_{P \in \mathcal{P}} \mathbb{E}_P[\epsilon_N(\widehat{x}, F, \mathcal{X}, P)] \geq \max_{v \in \mathcal{V}} \mathbb{E}_{P_v}\left[\epsilon_N(\widehat{x}, F, \mathcal{X}, P_v)\right] \geq MR \max_{v \in \mathcal{V}} \sum_{j=1}^{d} p_j \delta_j P_v(\text{sign}(\widehat{x}_j(\xi^{1:N}) \neq v_j), \tag{10}$$

where the last probability distribution is the product P_v^N over the samples $\xi^{1:N}$. Indeed, define

$$x_v^* = \operatorname*{argmin}_{x \in \mathcal{X}} \mathbb{E}_{P_v}[F(x; \xi)] = Rv,$$

the last inequality following by inspection of the loss. We then have

$$\begin{aligned} &\epsilon_N(\widehat{x}, F, \mathcal{X}, P_v) \\ &= M \sum_{j=1}^{N} p_j \left[\frac{1 + \delta_j}{2} |\widehat{x}_j - Rv_j| - \frac{1 - \delta_j}{2} |\widehat{x}_j + Rv_j| \frac{1 + \delta_j}{2} \left|x_{v,j}^* - Rv_j\right| - \frac{1 - \delta_j}{2} \left|x_{v,j}^* + Rv_j\right| \right]. \end{aligned}$$

By inspecting the cases for the possible values of $\text{sign}(\widehat{x}_j)$, we have

$$\frac{1 + \delta_j}{2} |\widehat{x}_j - Rv_j| - \frac{1 - \delta_j}{2} |\widehat{x}_j + Rv_j| + \frac{1 + \delta_j}{2} \left|x_{v,j}^* - Rv_j\right| - \frac{1 - \delta_j}{2} \left|x_{v,j}^* + Rv_j\right| \geq R\delta_j \mathbf{1}\left\{\text{sign}(\widehat{x}_j) \neq v_j\right\}.$$

Taking expectations of this quantity gives the result (10).

Bounding the test error Now we transform our testing problem to a randomized testing problem, averaging over $v \in \mathcal{V}$, which allows us to give a lower bound by controlling the error probability of d distinct simple hypothesis tests. From inequality (10) we have that

$$\max_{v\in\mathcal{V}} \sum_{j=1}^d p_j \delta_j P_v(\text{sign}(\widehat{x}_j(\xi^{1:N})) \neq v_j) \geq \sum_{j=1}^d p_j \delta_j \frac{1}{|\mathcal{V}|} \sum_{v\in\mathcal{V}} P_v(\text{sign}(\widehat{x}_j(\xi^{1:N})) \neq v_j) \tag{11}$$
$$= \sum_{j=1}^d p_j \delta_j \frac{1}{|\mathcal{V}|} \Bigg(\sum_{v:v_j=1} P_v(\text{sign}(\widehat{x}_j(\xi^{1:N})) \neq v_j) + \sum_{v:v_j=-1} P_v(\text{sign}(\widehat{x}_j(\xi^{1:N})) \neq v_j) \Bigg).$$

Recall that for any distributions P, Q, the variational representations of total variation distance imply that

$$\|P - Q\|_{\text{TV}} = \frac{1}{2} \sup_{f:\|f\|_\infty \leq 1} \int f(dP - dQ) = 1 - \inf \left\{ \int f dP + \int g dQ, \;\; f \geq 0, g \geq 0, f + g \geq 1 \right\}.$$

Now, let $P_{v,j}$ be the distribution (9), except that we fix $v_j = 1$ (and let $P_{v,-j}$ similarly have $v_j = -1$ fixed). With this notation, we have

$$P_{v,j}(\text{sign}(\widehat{x}_j) \neq v_j) + P_{v,-j}(\text{sign}(\widehat{x}_j) \neq v_j) = \mathbb{E}_{P_{v,j}} \left[\mathbf{1}\left\{\text{sign}(\widehat{x}_j) \neq 1\right\}\right] + \mathbb{E}_{P_{v,-j}} \left[\mathbf{1}\left\{\text{sign}(\widehat{x}_j) \neq -1\right\}\right].$$

Since $\mathbf{1}\left\{\text{sign}(\widehat{x}_j) \neq 1\right\} + \mathbf{1}\left\{\text{sign}(\widehat{x}_j) \neq -1\right\} \geq 1$, we find that

$$P_{v,j}(\text{sign}(\widehat{x}_j(\xi^{1:N})) \neq v_j) + P_{v,-j}(\text{sign}(\widehat{x}_j(\xi^{1:N})) \neq v_j) \geq 1 - \left\|P_{v,j}^N - P_{v,-j}^N\right\|_{\text{TV}},$$

where we recall that $P_{v,j}^N$ denotes the n-fold product of $P_{v,j}$. Thus, our lower bound (11) becomes the following (essentially a variant of Assouad's lemma):

$$\max_{v\in\mathcal{V}} \sum_{j=1}^d p_j \delta_j P_v(\text{sign}(\widehat{x}_j(\xi^{1:N})) \neq v_j) \geq \sum_{j=1}^d p_j \delta_j \frac{1}{2|\mathcal{V}|} \sum_{v\in\mathcal{V}} \left(1 - \left\|P_{v,j}^N - P_{v,-j}^N\right\|_{\text{TV}}\right). \tag{12}$$

Simple hypothesis tests For the majority of the remainder of the proof, we derive bounds on $\|P_{v,j}^N - P_{v,-j}^N\|_{\text{TV}}$ to apply inequality (12). Using Pinsker's inequality, we have

$$\left\|P_{v,j}^N - P_{v,-j}^N\right\|_{\text{TV}}^2 \leq \frac{1}{2} D_{\text{kl}}\left(P_{v,j}^N \| P_{v,-j}^N\right) \leq \frac{N}{2} D_{\text{kl}}\left(P_{v,j} \| P_{v,-j}\right).$$

Noting that P_v is a product distribution over the coordinates of the samples ξ (recall the construction (9)), we have the equality

$$D_{\text{kl}}\left(P_{v,j} \| P_{v,-j}\right) = p_j \left[\frac{1+\delta_j}{2} \log \frac{1+\delta_j}{1-\delta_j} + \frac{1-\delta_j}{2} \log \frac{1-\delta_j}{1+\delta_j}\right] = p_j \left[\delta_j \log \frac{1+\delta_j}{1-\delta_j}\right].$$

Now we use the fact that $\delta \log \frac{1+\delta}{1-\delta} \leq 2\log(3)\delta^2$ for $\delta \leq 1/2$, so

$$\left\|P_{v,j}^N - P_{v,-j}^N\right\|_{\text{TV}}^2 \leq N p_j \delta_j^2 \log(3) \;\; \text{for } \delta_j \in [0, 1/2]. \tag{13}$$

Combining inequalities (10), (12) and (13), we find

$$\frac{1}{MR}\sup_{P\in\mathcal{P}}\mathbb{E}_P\left[\epsilon_N(\widehat{x},F,\mathcal{X},P)\right] \ge \frac{1}{2}\sum_{j=1}^d p_j\delta_j\frac{1}{|\mathcal{V}|}\sum_{v\in\mathcal{V}}\left[1-\delta_j\sqrt{Np_j\log(3)}\right]$$

$$= \frac{1}{2}\sum_{j=1}^d p_j\delta_j\left[1-\delta_j\sqrt{Np_j\log(3)}\right]. \tag{14}$$

Inequality (14) holds for all $\delta_j\in[0,1/2]$, so we may maximize over such δ_j. By setting

$$\delta_j=\min\left\{\frac{1}{2},\frac{1}{2\sqrt{Np_j\log(3)}}\right\},$$

we have

$$p_j\delta_j\left[1-\delta_j\sqrt{Np_j\log(3)}\right]\ge p_j\min\left\{\frac{1}{4},\frac{1}{4\sqrt{\log 3}}\frac{1}{\sqrt{Np_j}}\right\}.$$

In particular, our simplified Assouad analogue (14) implies

$$\sup_{P\in\mathcal{P}}\mathbb{E}_P\left[\epsilon_N(\widehat{x},F,\mathcal{X},P)\right]\ge\frac{MR}{8}\sum_{j=1}^d\min\left\{p_j,\frac{\sqrt{p_j}}{\sqrt{N\log 3}}\right\},$$

which is the desired statement of the proposition. □

A.2 Proof of Corollary 2

The proof follows by a simple integration argument.

B Proof of Theorem 3

Our proof proceeds in a series of lemmas, which we present shortly. To connect the result to more standard analyses of dual averaging, and because the lemmas we present form the basis of our subsequent analysis for AsyncAdaGrad, we prove our results in somewhat more generality than that presented in the main text. In full generality, we allow $\{\psi_t\}_t$ be a sequence of functions, each strongly convex with respect to the norm $\|\cdot\|_{\psi_t}$. Then dual averaging [11, 4] can be viewed as repeatedly computing

$$x(t) := \operatorname*{argmin}_{x\in\mathcal{X}}\left\{\langle z(t),x\rangle+\psi_t(x)\right\}, \tag{15}$$

where the proximal function ψ_t may or may not vary with time. For the theoretical development, we define the conjugate to ψ_t and associated dual norm

$$\psi_t^*(z):=\sup_{x\in\mathcal{X}}\left\{\langle z,x\rangle-\psi_t(x)\right\} \quad\text{and}\quad \|z\|_{\psi_t^*}:=\sup_x\left\{\langle z,x\rangle \mid \|x\|_{\psi_t}\le 1\right\}.$$

We have the following standard lemma (see, e.g. the book of Hiriart-Urruty and Lemaréchal [6, Chapter X]), which is essential to the analysis of the dual averaging method (15). The result says that since ψ_t is strongly convex with respect to the norm $\|\cdot\|_{\psi_t}$, its dual is smoothly differentiable and, more strongly, has Lipschitz derivative with respect to the dual norm $\|\cdot\|_{\psi_t^*}$.

Lemma 7. *The function ψ_t^* is 1-strongly smooth with respect to $\|\cdot\|_{\psi^*}$, meaning that*

$$\left\|\nabla\psi_t^*(z) - \nabla\psi_t^*(z')\right\|_{\psi_t} \le \left\|z - z'\right\|_{\psi_t^*},$$

and moreover $\nabla\psi_t^*(-z(t)) = x(t)$.

We also recall the standard fact [e.g. 6] that if a function h has Lipschitz continuous derivative with respect to a norm $\|\cdot\|$, then $h(y) \le h(x) + \langle\nabla h(x), y - x\rangle + (1/2)\|x - y\|^2$ for all $x, y \in \operatorname{dom} h$.

With Lemma 7 in place, we can provide a general result on the convergence of asynchronous dual averaging-type algorithms. In this lemma, we use the time indexing established in Section 3.1. In particular, we have that $x(t)$ is computed using the proximal function ψ_t, that is, as in the update (15). This result shows that the performance of AsyncDA, and other algorithms like it, is essentially identical to that of the correct dual averaging algorithm, plus a correction penalty that relates the performance of the correct algorithm to what is actually executed.

Lemma 8. *Define the "corrected" point sequence*

$$\widetilde{x}(t) := \nabla\psi_t^*\Big(-\sum_{\tau=1}^{t-1} g(\tau)\Big) = \operatorname*{argmin}_{x\in\mathcal{X}}\Big\{\sum_{\tau=1}^{t-1}\langle g(\tau), x\rangle + \psi_t(x)\Big\}.$$

For any sequence of samples ξ^t,

$$\sum_{t=1}^T \left[F(x(t);\xi^t) - F(x^*;\xi^t)\right] \le \sum_{t=1}^T\left[\psi_t^*\Big(-\sum_{\tau=1}^{t-1} g(\tau)\Big) - \psi_{t-1}^*\Big(-\sum_{\tau=1}^{t-1} g(\tau)\Big)\right] + \frac{1}{2}\sum_{t=1}^T \|g(t)\|_{\psi_t^*}^2 + \sum_{t=1}^T \langle g(t), x(t) - \widetilde{x}(t)\rangle + \psi_T(x^*)$$

We remark that in the non-asynchronous case, one has $\psi_t^* \le \psi_{t-1}^*$ (either in adaptive settings such as AdaGrad [4] or in standard dual averaging) and $x(t) = \widetilde{x}(t)$, so Lemma 8 reduces to the bound $(1/2)\sum_{t=1}^T \|g(t)\|_{\psi_t^*}^2 + \psi_T(x^*)$, which is standard [11].

Deferring the proof of Lemma 8 to Section B.1, we note that in the context of Theorem 3, it has immediate implications. Since $\psi_t(x) = \frac{1}{2\eta}\|x\|_2^2$ in this case, Lemma 8 immediately implies

$$\sum_{t=1}^T \left[F(x(t);\xi^t) - F(x^*;\xi^t)\right] \le \frac{1}{2\eta}\|x^*\|_2^2 + \frac{\eta}{2}\sum_{t=1}^T \|g(t)\|_2^2 + \sum_{t=1}^T \langle g(t), x(t) - \widetilde{x}(t)\rangle \tag{16}$$

since $\psi_t^* = \psi_{t-1}^*$ and $\psi^*(0) \le 0$, and for any v

$$\|v\|_\psi^2 = \frac{1}{\eta}\|v\|_2^2 \quad\text{and}\quad \|v\|_{\psi^*}^2 = \eta\|v\|_2^2.$$

Now we return to the proof of Theorem 3. Each of the terms present in Theorem 3 is present in Eq. (16) except for the last, since

$$\mathbb{E}[\|g(t)\|_2^2] \le \mathsf{M}^2.$$

For the final term in the bound in the theorem, we note that by assumption that $\mathcal{X}$ is a product domain,

$$\langle g(t), x(t) - \widetilde{x}(t) \rangle \le \sum_{j=1}^{d} |g_j(t)||x_j(t) - \widetilde{x}_j(t)| \le \sum_{j=1}^{d} \eta |g_j(t)| \bigg| \sum_{\tau=1}^{t-1} g_j(\tau) - z_j(t) \bigg|.$$

For the final inequality we have used that by the definition of the x update (recall Lemma 7),

$$|\widetilde{x}_j(t) - x_j(t)| = \bigg| \nabla_j \psi^* \bigg(- \sum_{\tau=1}^{t-1} g(\tau) \bigg) - \nabla_j \psi^*(-z(t)) \bigg| \le \eta \bigg| \sum_{\tau=1}^{t-1} g_j(\tau) - z_j(\tau) \bigg|.$$

Conditioned on the σ-field $\mathcal{F}_{t-1}$ of $\{\xi^\tau\}_{\tau=1}^{t-1}$, we have $\mathbb{E}[|g_j(t)| \mid \mathcal{F}_{t-1}] \le p_j M_j$ by assumption (since ξ^t is independent of ξ^τ for $\tau < t$). Moreover, we have $\mathbb{E}[|\sum_{\tau=1}^{t-1} g_j(\tau) - z_j(t)|] \le m p_j M_j$ because the delay in each processor is assumed to be at most m and $\mathbb{E}[|g_j(\tau)|] \le p_j M_j$. Thus we find

$$\mathbb{E}[\langle g(t), x(t) - \widetilde{x}(t) \rangle] \le \eta \sum_{j=1}^{d} \mathbb{E}\left[\mathbb{E}[|g_j(t)| \mid \mathcal{F}_{t-1}] \bigg| \sum_{\tau=1}^{t-1} g_j(\tau) - z_j(t) \bigg| \right] \le \eta \sum_{j=1}^{d} p_j^2 M_j^2 m.$$

This completes the proof. □

B.1 Proof of Lemma 8

The proof is similar to other analyses of dual averaging (e.g. [11, 4]), but we track the changing time indices. For shorthand throughout this proof, we define the running sum $g(1{:}t) := \sum_{\tau=1}^{t} g(\tau)$. First, we note by convexity and the definition of $g(t) \in \partial F(x; \xi^t)$ that

$$\sum_{t=1}^{T} \left[F(x(t); \xi^t) - F(x^*; \xi^t) \right] \le \sum_{t=1}^{T} \langle g(t), x(t) - x^* \rangle . \tag{17}$$

By definition of ψ_T and the conjugate ψ_T^*, we find that

$$\begin{aligned}\sum_{t=1}^{T} \langle g(t), x(t) - x^* \rangle &= \sum_{t=1}^{T} \langle g(t), x(t) \rangle + \sum_{t=1}^{T} \langle -g(t), x^* \rangle + \psi_T(x^*) - \psi_T(x^*) \\ &\le \psi_T(x^*) + \psi_T^*(-g(1{:}T)) + \sum_{t=1}^{T} \langle g(t), x(t) \rangle . \end{aligned} \tag{18}$$

Now, by applying Lemma 7 and the definition of 1-strongly-smooth, we have that

$$\psi_t^*(-g(1{:}t)) \le \psi_t^*(-g(1{:}t-1)) + \langle -g(t), \nabla \psi_t^*(-g(1{:}t-1)) \rangle + \frac{1}{2} \|g(t)\|_{\psi_t^*}^2 .$$

By construction of x and $\widetilde{x}$, we have $x(t) = \nabla \psi_t^*(-z(t))$ and $\widetilde{x}(t) = \nabla \psi_t^*(-g(1{:}t-1))$. Thus, rearranging the preceding display, we have

$$0 \le \langle -g(t), \widetilde{x}(t) \rangle - \psi_t^*(-g(1{:}t))) + \psi_t^*(-g(1{:}t-1)) + \frac{1}{2} \|g(t)\|_{\psi_t^*}^2 ,$$

and adding $\langle g(t), x(t) \rangle$ to both sides of the above expression gives

$$\langle g(t), x(t) \rangle \le \langle g(t), x(t) - \widetilde{x}(t) \rangle - \psi_t^*(-g(1{:}t)) + \psi_t^*(-g(1{:}t-1)) + \frac{1}{2} \left\| g(t) \right\|_{\psi_t^*}^2 . \tag{19}$$

Thus we obtain the inequalities

$$\begin{aligned}
&\sum_{t=1}^T \langle g(t), x(t) - x^* \rangle \\
&\overset{(i)}{\le} \psi_T(x^*) + \psi_T^*(-g(1{:}T))) + \sum_{t=1}^T \langle g(t), x(t) \rangle \\
&\overset{(ii)}{\le} \psi_T(x^*) + \psi_T^*(-g(1{:}T)) + \sum_{t=1}^T \Big[\langle g(t), x(t) - \widetilde{x}(t) \rangle - \psi_t^*(-g(1{:}t)) + \psi_t^*(-g(1{:}t-1)) + \frac{1}{2} \left\| g(t) \right\|_{\psi_t^*}^2 \Big] \\
&= \psi_T(x^*) + \sum_{t=1}^T \Big[\langle g(t), x(t) - \widetilde{x}(t) \rangle + \psi_t^*(-g(1{:}t-1)) - \psi_{t-1}^*(-g(1{:}t-1)) + \frac{1}{2} \left\| g(t) \right\|_{\psi_t^*}^2 \Big],
\end{aligned}$$

where for step (i) we have applied inequality (18), step (ii) follows from the bound (19), and the last equality follows by re-indexing terms in the sum. Combining the above sum with the first-order convexity inequality (17) proves the lemma. □

C Proof of Theorem 5

Before proving the theorem proper, we recall a now standard lemma that is essential to proving rates of convergence for adaptive algorithms.

Lemma 9 (Auer and Gentile [1], Duchi et al. [4] Lemma 4, McMahan and Streeter [9]). *For any non-negative sequence $\{a_t\}_t$, where we define $0/\sqrt{0} = 0$, we have*

$$\sum_{t=1}^T \frac{a_t}{\sqrt{\sum_{\tau=1}^t a_\tau}} \le 2 \bigg(\sum_{t=1}^T a_t \bigg)^{\frac{1}{2}}.$$

The proof of this theorem follows from the general bound of Lemma 8 applied with a particular choice of the proximal functions ψ_t. To apply the lemma, we require a few definitions that apply throughout the proof. We recall the definitions of $z(t)$ and $S(t)$ to be the values *read* in the computation of step 1 to construct the vector $x(t)$ in the definition of AsyncAdaGrad. In addition, we define the two temporal inequalities $\prec_{S_j}$ and $\prec_{z_j}$ to capture the order in which the updates are applied in the AsyncAdaGrad algorithm. We say that $\tau \prec_{S_j} t$ if the gradient term $g_j(\tau)^2$ has been incorporated into the matrix coordinate S_j at the instant S_j is read in step (1) of the AsyncAdaGrad algorithm to compute $x(t)$, and similarly, we say $\tau \prec_{z_j} t$ if the gradient term $g_j(\tau)$ has been incorporated in the dual vector coordinate z_j.

Before actually applying the general bound of Lemma 8, we note that by convexity,

$$\sum_{t=1}^T \left[F(x(t); \xi^t) - F(x^*; \xi^t) \right] \le \sum_{t=1}^T \langle g(t), x(t) - x^* \rangle .$$

Considering a particular coordinate j, we have

$$\sum_{t=1}^{T} \mathbb{E}\left[g_j(t)(x_j(t) - x_j^*)\right] \le R_\infty \sum_{t=1}^{T} \mathbb{E}[|g_j(t)|] \le R_\infty T M_j p_j \tag{20}$$

where we have used the compactness assumption on $\mathcal{X}$. The remainder of our proof bounds the regret-like term $\sum_{t=1}^{T} \langle g(t), x(t) - x^* \rangle$ in a per-coordinate way, and thus for each coordinate we always have the bound (20), giving the $\min\{\cdot, p_j\}$ terms in the theorem statement. It remains to show the bound that applies when p_j is large.

We now re-state the general bound of Lemma 8 with some minor modifications in notation. AsyncAdaGrad is dual averaging with the choice $\psi_t(x) := \frac{1}{2\eta}\langle x, G(t)x \rangle$ for the proximal function. With this choice, the norm and dual norm $\|\cdot\|_{\psi_t}$ and $\|\cdot\|_{\psi_t^*}$ defined for vectors $v \in \mathbb{R}^d$ are

$$\|v\|_{\psi_t}^2 := \frac{1}{\eta}\|v\|_{G(t)}^2 \quad \text{and} \quad \|v\|_{\psi_t^*}^2 := \eta \|v\|_{G(t)^{-1}}^2 .$$

Rewriting Lemma 8, we thus have

$$\begin{aligned}\sum_{t=1}^{T}\left[F(x(t);\xi^t) - F(x^*;\xi^t)\right] \le \sum_{t=1}^{T}\left[\psi_t^*\Big(-\sum_{\tau=1}^{t-1} g(\tau)\Big) - \psi_{t-1}^*\Big(-\sum_{\tau=1}^{t-1} g(\tau)\Big)\right] + \frac{\eta}{2}\sum_{t=1}^{T}\|g(t)\|_{G(t)^{-1}}^2 \\ + \sum_{t=1}^{T}\langle g(t), x(t) - \widetilde{x}(t)\rangle + \frac{1}{2\eta}\|x^*\|_{G(T)}^2 \end{aligned} \tag{21}$$

for any sequence ξ^t; as in Lemma 8 we define the "corrected" sequences $\widetilde{x}(t) = \nabla\psi_t^*(-g(1\!:\!t-1))$ where $g(1\!:\!t) = \sum_{\tau=1}^{t-1} g(\tau)$. Note that the corrected sequences still use the proximal functions ψ_t^* from the *actual* run of the algorithm.

We focus on bounding each of the terms in the sums (21) in turn, beginning with the summed conjugate differences.

Lemma 10. *Define the matrix $G(T_+)$ to be diagonal with jth diagonal entry $(\delta^2 + \sum_{t=1}^{T} g_j(t)^2)^{\frac{1}{2}}$. For any sequence ξ^t*

$$\sum_{t=1}^{T}\left[\psi_t^*\Big(-\sum_{\tau=1}^{t-1} g(\tau)\Big) - \psi_{t-1}^*\Big(-\sum_{\tau=1}^{t-1} g(\tau)\Big)\right] \le \frac{R_\infty^2}{2\eta}\operatorname{tr}(G(T_+)).$$

We defer the proof of Lemma 10 to Section C.1, noting that the proof follows by carefully considering the conditions under which $\psi_t^* \ge \psi_{t-1}^*$, which may only occur when updates to S (and hence G) are out of order, a rearrangement of the sum to put the updates to S in the correct order, and an application of the AdaGrad Lemma 9.

To complete the proof of the theorem, we must bound the two summed gradient quantities in expression (21). For shorthand, let us define

$$\mathcal{T}_1 := \sum_{t=1}^{T}\|g(t)\|_{G(t)^{-1}}^2 \quad \text{and} \quad \mathcal{T}_2 := \sum_{t=1}^{T}\langle g(t), x(t) - \widetilde{x}(t)\rangle \tag{22}$$

We provide the proof under the assumption that $\delta^2 \ge mM_j^2$ for all j. At the end of the proof, we show how to weaken this assumption while retaining the main conclusions of the theorem.

Recalling the temporal ordering notation $\prec_{S_j}$, we see that

$$\mathcal{T}_1 = \sum_{j=1}^{d} \sum_{t=1}^{T} \frac{g_j(t)^2}{\sqrt{\delta^2 + \sum_{\tau \prec_{S_j} t} g_j(\tau)^2}}.$$

Now, by our assumption that processors are *at most* m steps out of date and $\delta^2 \geq m M_j^2$, we have

$$\frac{g_j(t)^2}{\sqrt{\delta^2 + \sum_{\tau \prec_{S_j} t} g_j(\tau)^2}} \leq \frac{g_j(t)^2}{\sqrt{\sum_{\tau=1}^{t} g_j(\tau)^2}},$$

and thus the standard AdaGrad result Lemma 9 implies

$$\mathcal{T}_1 \leq \sum_{j=1}^{d} \sum_{t=1}^{T} \frac{g_j(t)^2}{\sqrt{\sum_{\tau=1}^{t} g_j(\tau)^2}} \leq \sum_{j=1}^{d} 2 \bigg(\sum_{t=1}^{T} g_j(t)^2 \bigg)^{\frac{1}{2}}. \tag{23}$$

Thus we turn to $\mathcal{T}_2$ as defined in expression (22). We focus on a per-coordinate version of $\mathcal{T}_2$, stating the following lemma:

Lemma 11. *Under the conditions of Theorem 5,*

$$\frac{1}{\eta} \sum_{t=1}^{T} \mathbb{E}[g_j(t)(x_j(t) - \widetilde{x}_j(t))] \leq 2 p_j m \mathbb{E}\left[\bigg(\sum_{t=1}^{T} g_j(t)^2 \bigg)^{\frac{1}{2}} \right] \leq 2 p_j m M_j \sqrt{p_j T}.$$

The proof of the lemma is technical, so we defer it to Section C.2.

Applying the result of Lemma 11, we obtain the following bound on $\mathcal{T}_2$:

$$\mathbb{E}[\mathcal{T}_2] \leq 2\eta \sum_{j=1}^{d} p_j m \mathbb{E}\left[\bigg(\sum_{t=1}^{T} g_j(t)^2 \bigg)^{\frac{1}{2}} \right].$$

Combining Lemma 10 with our bounds (23) on $\mathcal{T}_1$ and the preceding bound on $\mathcal{T}_2$, the basic inequality (21) implies

$$\begin{aligned} &\mathbb{E}\bigg[\sum_{t=1}^{T} \big(F(x(t); \xi^t) - F(x^*; \xi^t) \big) \bigg] \\ &\leq \frac{1}{2\eta} \mathbb{E}\left[\|x^*\|_{G(T)}^2 + R_\infty^2 \operatorname{tr}(G(T_+)) \right] + \eta \sum_{j=1}^{d} \mathbb{E}\left[\bigg(\sum_{t=1}^{T} g_j(t)^2 \bigg)^{\frac{1}{2}} \right] (1 + 2 p_j m). \end{aligned}$$

Noting that

$$\|x^*\|_{G(T)}^2 \leq R_\infty^2 \operatorname{tr}(G(T)) \leq R_\infty^2 \sum_{j=1}^{d} \bigg(\delta^2 + \sum_{t=1}^{T} g_j(t)^2 \bigg)^{\frac{1}{2}}$$

completes the proof of Theorem 5 under the assumption that $\delta^2 \geq m M_j^2$ for all j. □

C.1 Proof of Lemma 10

Since the domain $\mathcal{X} = \mathcal{X}_1 \times \ldots \times \mathcal{X}_d$ is assumed Cartesian and the matrices S and $G = S^{\frac{1}{2}}$ are diagonal, we focus on the individual coordinate terms of ψ_t^*. With that in mind, consider the difference

$$\sup_{x_j \in \mathcal{X}_j} \bigg\{ -\sum_{\tau=1}^{t-1} g_j(\tau) x_j - \frac{1}{2\eta} x_j G_j(t) x_j \bigg\} - \sup_{x_j \in \mathcal{X}_j} \bigg\{ -\sum_{\tau=1}^{t-1} g_j(\tau) x_j - \frac{1}{2\eta} x_j G_j(t-1) x_j \bigg\}. \tag{24}$$

To understand the difference of the terms (24), we recall the temporal ordering $\prec_{S_j}$ defined in the beginning of the proof of Theorem 5 (we say $\tau \succeq_{S_j} t$ if and only if $\tau \not\prec_{S_j} t$). Though throughout the algorithm, the matrix S (and the matrix G) is always increasing—only positive updates are applied to S—when indexed by update time, we may have $G_j(t-1) \lessgtr G_j(t)$. The term (24), however, may be positive only when $G_j(t) < G_j(t-1)$, and this is possible only if

$$\left\{\tau \in \mathbb{N} \mid \tau \prec_{S_j} t-1\right\} \supsetneq \left\{\tau \in \mathbb{N} \mid \tau \prec_{S_j} t\right\}.$$

Finally, we note that for any matrices A, B and vector z, that if we define

$$x(A) := \operatorname*{argmax}_{x \in \mathcal{X}} \left\{ \langle z, x \rangle - \langle x, Ax \rangle \right\}$$

then

$$\begin{aligned}&\sup_{x \in \mathcal{X}} \left\{ \langle z, x \rangle - \langle x, Ax \rangle \right\} - \sup_{x \in \mathcal{X}} \left\{ \langle z, x \rangle - \langle x, Bx \rangle \right\} \\ &\quad \le \langle z, x(A) \rangle - \langle x(A), Ax(A) \rangle - \langle z, x(A) \rangle + \langle x(A), Bx(A) \rangle \le \sup_{x \in \mathcal{X}} \left\{ \langle x, (B-A)x \rangle \right\}.\end{aligned}$$

By considering expression (24), we have

$$\begin{aligned}\psi_t^*\bigg(-\sum_{\tau=1}^{t-1} g(\tau)\bigg) - \psi_{t-1}^*\bigg(-\sum_{\tau=1}^{t-1} g(\tau)\bigg) &\le \frac{1}{2\eta} \sum_{j=1}^{d} \sup_{x_j \in \mathcal{X}_j} \left\{ x_j^2 (G_j(t-1) - G_j(t)) \right\} \\ &\le \frac{R_\infty^2}{2\eta} \sum_{j=1}^{d} |G_j(t) - G_j(t-1)| \, \mathbf{1}\left\{ \{\tau \prec_{S_j} t-1\} \supsetneq \{\tau \prec_{S_j} t\} \right\}.\end{aligned} \tag{25}$$

It thus remains to bound the sum, over all t, of the terms (25). To that end, we note by concavity of $\sqrt{\cdot}$ that for any $a, b \ge 0$, we have $\sqrt{a+b} - \sqrt{a} \le b/2\sqrt{a}$. Thus we find that

$$\begin{aligned}&|G_j(t) - G_j(t-1)| \, \mathbf{1}\left\{ \{\tau \prec_{S_j} t-1\} \supsetneq \{\tau \prec_{S_j} t\} \right\} \\ &= \left| \bigg(\delta^2 + \sum_{\tau \prec_{S_j} t} g_j(\tau)^2\bigg)^{\frac{1}{2}} - \bigg(\delta^2 + \sum_{\tau \prec_{S_j} t-1} g_j(\tau)^2\bigg)^{\frac{1}{2}} \right| \mathbf{1}\left\{ \{\tau \prec_{S_j} t-1\} \supsetneq \{\tau \prec_{S_j} t\} \right\} \\ &\le \frac{\sum_{\tau \prec_{S_j} t-1, \tau \succeq_{S_j} t} g_j(\tau)^2}{2\sqrt{\delta^2 + \sum_{\tau \prec_{S_j} t} g_j(\tau)^2}}.\end{aligned}$$

We note the following: the sequence of update sets $\Delta_t := \{\tau \in \mathbb{N} : \tau \prec_{S_j} t-1, \tau \succeq_{S_j} t\}$ satisfies $\cup_{t=1}^T \Delta_t \subset [T]$, and since the incremental updates to S occur only once, we have $\Delta_t \cap \Delta_{t'} = \emptyset$ for all $t \neq t'$. That is, if $\tau \in \Delta_t$ for some t, then $\tau \notin \Delta_{t'}$ for any $t' \neq t$. Using the assumption that updates may be off by at most m time steps, we thus see that there must exist some permutation $\{u_t\}_{t=1}^T$ of $[T]$ such that

$$\sum_{t=1}^T \frac{\sum_{\tau \in \Delta_t} g_j(\tau)^2}{\sqrt{\delta^2 + \sum_{\tau \prec_{S_j} t} g_j(\tau)^2}} \le \sum_{t=1}^T \frac{g_j(u_t)^2}{\sqrt{\delta^2 + \sum_{\tau \le t-m} g_j(u_\tau)^2}}. \tag{26}$$

For our last step, we use our assumption that $\delta^2 \ge m M_j^2$ and the standard AdaGrad result (Lemma 9) to obtain

$$\sum_{t=1}^T \frac{\sum_{\tau \prec_{S_j} t-1, \tau \succeq_{S_j} t} g_j(\tau)^2}{\sqrt{\delta^2 + \sum_{\tau \prec_{S_j} t} g_j(\tau)^2}} \le \sum_{t=1}^T \frac{g_j(u_t)^2}{\sqrt{\sum_{\tau=1}^t g_j(u_\tau)^2}} \le 2\sqrt{\sum_{t=1}^T g_j(t)^2}.$$

Recalling inequality (25), we have

$$\sum_{t=1}^T \left[\psi_t^*\Big(-\sum_{\tau=1}^{t-1} g(\tau)\Big) - \psi_{t-1}^*\Big(-\sum_{\tau=1}^{t-1} g(\tau)\Big)\right] \le \frac{R_\infty^2}{2\eta} \sum_{j=1}^d \sqrt{\sum_{t=1}^T g_j(t)^2},$$

which gives the statement of the lemma. □

C.2 Proof of Lemma 11

Let us provide a bit of notation before proving the lemma. We define the batch of "outstanding" updates for coordinate j at time t as $\mathcal{B}_j(t) := \{\tau : t-1 \ge \tau \succeq_{z_j} t\}$, and we define the quantity that we wish to bound in expectation in Lemma 11 as

$$\mathcal{T}^j := \frac{1}{\eta} \sum_{t=1}^T g_j(t)(x_j(t) - \widetilde{x}_j(t)).$$

Turning to the proof of the lemma proper, we first note that $z(t)$ does not include any gradient terms $g(\tau)$ for any $\tau \ge t$ by the definition of the AsyncAdaGrad algorithm. Thus

$$\sum_{\tau=1}^{t-1} g_j(\tau) - z_j(t) = \sum_{\tau \in \mathcal{B}_j(t)} g_j(\tau).$$

For brief notational convenience define $\kappa_t = \eta(\delta^2 + \sum_{\tau \prec_{S_j} t} g_j(\tau)^2)^{\frac{1}{2}}$. Applying the definition of the AsyncAdaGrad updates and Young's inequality, we see that

$$\begin{aligned} g_j(t) \cdot (x_j(t) - \widetilde{x}_j(t)) &\le \kappa_t |g_j(t)| \, |g_j(1{:}t-1) - z_j(t)| \\ &= \kappa_t |g_j(t)| \Big| \sum_{\tau \in \mathcal{B}_j(t)} g_j(\tau) \Big| \le \frac{1}{2}\kappa_t |g_j(t)|^2 \mathbf{1}\left\{\xi_j^t \neq 0\right\} + \frac{1}{2}\kappa_t \mathbf{1}\left\{\xi_j^t \neq 0\right\} \Big(\sum_{\tau \in \mathcal{B}_j(t)} g_j(\tau) \Big)^2. \end{aligned}$$

As a consequence, we find that

$$2\mathbb{E}[\mathcal{T}^j] \le \sum_{t=1}^T \mathbb{E}\left[\frac{\mathrm{card}(\{\tau \in \mathcal{B}_j(t) : \xi_j^\tau \neq 0\}) g_j(t)^2}{\sqrt{\delta^2 + \sum_{\tau \prec_{S_j} t} g_j(\tau)^2}} + \frac{\mathbf{1}\left\{\xi_j^t \neq 0\right\} \sum_{\tau \in \mathcal{B}_j(t)} g_j(\tau)^2}{\sqrt{\delta^2 + \sum_{\tau \prec_{S_j} t} g_j(\tau)^2}}\right]. \tag{27}$$

Looking at the first term in the bound (27), we note that $\mathcal{B}_j(t)$ consists of time indices $t \preceq_{z_j} \tau \le t-1$, which consequently have not been incorporated into any vectors used in the computation of $g_j(t)$. Thus, if we let $\mathcal{F}_{t,j}$ denote the σ-field containing ξ_j^t and ξ_j^τ for $\tau \prec_{z_j} t$, we have $g_j(\tau) \in \mathcal{F}_{t,j}$ for any $\tau \prec_{S_j} t$, $g_j(t) \in \mathcal{F}_{t,j}$, and we also have that ξ_j^τ is independent of $\mathcal{F}_{t,j}$ for $\tau \in \mathcal{B}_j(t)$. Thus, iterating expectations, we find

$$\begin{aligned}
\mathbb{E}\left[\frac{\mathrm{card}(\{\tau \in \mathcal{B}_j(t) : \xi_j^\tau \neq 0\}) g_j(t)^2}{\sqrt{\delta^2 + \sum_{\tau \prec_{S_j} t} g_j(\tau)^2}}\right] &= \mathbb{E}\left[\frac{\mathbb{E}[\mathrm{card}(\{\tau \in \mathcal{B}_j(t) : \xi_j^\tau \neq 0\}) g_j(t)^2 \mid \mathcal{F}_{t,j}]}{\sqrt{\delta^2 + \sum_{\tau \prec_{S_j} t} g_j(\tau)^2}}\right] \\
&\le p_j m \mathbb{E}\left[\frac{g_j(t)^2}{\sqrt{\delta^2 + \sum_{\tau \prec_{S_j} t} g_j(\tau)^2}}\right],
\end{aligned}$$

since $\mathbb{E}[\mathrm{card}(\{\tau \in \mathcal{B}_j(t) : \xi_j^\tau \neq 0\})] \le p_j m$ because $|\mathcal{B}_j(t)| \le m$ by assumption. A similar iteration of expectation—since ξ_j^t is independent of any $g_j(\tau)$ for $\tau \in \mathcal{B}_j(t)$—yields

$$\mathbb{E}\Bigg[\mathbf{1}\left\{\xi_j^t \neq 0\right\} \sum_{\tau \in \mathcal{B}_j(t)} g_j(\tau)^2\Bigg] \le p_j \mathbb{E}\Bigg[\sum_{\tau \in \mathcal{B}_j(t)} g_j(\tau)^2\Bigg].$$

We replace the relevant terms in the expectation (27) with the preceding bounds to obtain

$$2\mathbb{E}[\mathcal{T}^j] \le p_j m \sum_{t=1}^T \mathbb{E}\left[\frac{g_j(t)^2}{\sqrt{\delta^2 + \sum_{\tau \prec_{S_j} t} g_j(\tau)^2}}\right] + p_j \sum_{t=1}^T \mathbb{E}\left[\frac{\sum_{\tau \in \mathcal{B}_j(t)} g_j(\tau)^2}{\sqrt{\delta^2 + \sum_{\tau \prec_{S_j} t} g_j(\tau)^2}}\right].$$

For the second term, note each $g_j(\tau)$ can occur in at most m of the sets $\mathcal{B}_j(t)$, and the maximum delay is also at most m. Thus, following the same argument as (26), there must exist a permutation $\{u_t\}$ of the indices $[T]$ such that

$$\begin{aligned}
\sum_{t=1}^T \frac{\sum_{\tau \in \mathcal{B}_j(t)} g_j(\tau)^2}{\sqrt{\delta^2 + \sum_{\tau \prec_{S_j} t} g_j(\tau)^2}} &\le \sum_{t=1}^T \frac{m g_j(u_t)^2}{\sqrt{\delta^2 + \sum_{\tau=1}^{t-m} g_j(u_\tau)^2}} \\
&\le \sum_{t=1}^T \frac{m g_j(u_t)^2}{\sqrt{\sum_{\tau=1}^t g_j(u_\tau)^2}} \le 2m\Bigg(\sum_{t=1}^T g_j(t)^2\Bigg)^{\frac{1}{2}},
\end{aligned}$$

where we have used the fact that $\delta^2 \ge m M_j^2$ and Lemma 9. With this, we immediately find that

$$2\mathbb{E}[\mathcal{T}^j] \le p_j m \sum_{t=1}^T \mathbb{E}\left[\frac{g_j(t)^2}{\sqrt{\sum_{\tau=1}^t g_j(\tau)^2}}\right] + p_j \sum_{t=1}^T \mathbb{E}\left[\frac{\sum_{\tau=t-m}^{t-1} g_j(\tau)^2}{\sqrt{\sum_{\tau=1}^t g_j(\tau)^2}}\right] \le 4 p_j m \mathbb{E}\left[\Bigg(\sum_{t=1}^T g_j(t)^2\Bigg)^{\frac{1}{2}}\right].$$

By inspection, this completes the proof of the lemma. □

C.3 Sharpening the analysis (proof of Corollary 6)

We now demonstrate how to sharpen the analysis in the proof of Theorem 5 to allow the initial matrix δ^2 to be smaller than mM_j^2. Roughly, we argue that for a smaller setting of δ^2, we can have $\delta^2 \ge \sum_{\tau=t-m+1}^{t} g_j(\tau)^2$ for all t with high probability, in which case all the previous arguments go through verbatim. In particular, we show how the terms $\mathcal{T}_1$ and $\mathcal{T}_2$, defined in expression (22), may be bounded under the weaker assumptions on δ^2 specified in Corollary 6.

For this argument, we focus on $\mathcal{T}_1$, as the argument for $\mathcal{T}_2$ is identical. We begin by defining the event $\mathcal{E}$ to occur if $\delta^2 \ge \sum_{\tau=t-m+1}^{t} g_j(\tau)^2$ for all t. We then have

$$\mathbf{1}\{\mathcal{E}\} \sum_{t=1}^{T} \frac{g_j(t)^2}{\sqrt{\delta^2 + \sum_{\tau \prec_{S_j} t} g_j(\tau)^2}} \le \mathbf{1}\{\mathcal{E}\} \sum_{t=1}^{T} \frac{g_j(t)^2}{\sum_{\tau=1}^{t} g_j(\tau)^2} \le 2\Big(\sum_{t=1}^{T} g_j(t)^2\Big)^{\frac{1}{2}}$$

by Lemma 9. On the other hand, on $\mathcal{E}^c$, we have by our assumption that $\delta^2 \ge M_j^2$ that

$$\mathbf{1}\{\mathcal{E}^c\} \sum_{t=1}^{T} \frac{g_j(t)^2}{\sqrt{\delta^2 + \sum_{\tau \prec_{S_j} t} g_j(\tau)^2}} \le \mathbf{1}\{\mathcal{E}^c\} \sum_{t=1}^{T} |g_j(t)|,$$

so if we can show that $\mathcal{E}^c$ has sufficiently low probability, then we still obtain our desired results. Indeed, by Hölder's inequality we have

$$\begin{aligned}
\mathbb{E}\Big[\sum_{t=1}^{T} \frac{g_j(t)^2}{\sqrt{\delta^2 + \sum_{\tau \prec_{S_j} t} g_j(\tau)^2}}\Big] &\le 2\mathbb{E}\left[\Big(\sum_{t=1}^{T} g_j(t)^2\Big)^{\frac{1}{2}}\right] + \mathbb{E}\Big[\mathbf{1}\{\mathcal{E}^c\} \sum_{t=1}^{T} \frac{g_j(t)^2}{\sqrt{\delta^2 + \sum_{\tau \prec_{S_j} t} g_j(\tau)^2}}\Big] \\
&\le 2\mathbb{E}\left[\Big(\sum_{t=1}^{T} g_j(t)^2\Big)^{\frac{1}{2}}\right] + \mathbb{E}[\mathbf{1}\{\mathcal{E}^c\}]^{\frac{1}{2}} \mathbb{E}\left[\Big(\sum_{t=1}^{T} |g_j(t)|\Big)^2\right]^{\frac{1}{2}}. \qquad (28)
\end{aligned}$$

It thus remains to argue that $\mathbb{P}(\mathcal{E}^c)$ is very small, since

$$\mathbb{E}\left[\Big(\sum_{t=1}^{T} |g_j(t)|\Big)^2\right] \le T\mathbb{E}\left[\sum_{t=1}^{T} g_j(t)^2\right]$$

by Jensen's inequality. Now, we note that $g_j(t)^2 \le M_j^2$ and that the ξ_j^t are i.i.d., so we can define the sequence $X_t = \mathbf{1}\left\{\xi_j^t \ne 0\right\}$ and we have

$$\mathbb{P}(\mathcal{E}^c) = \mathbb{P}\Big(\exists\, t \in [T] : \sum_{\tau=t}^{t+m-1} g_j(\tau)^2 > \delta^2\Big) \le \mathbb{P}\Big(\exists\, t \in [T] : \sum_{\tau=t}^{t+m-1} X_t > \delta^2/M_j^2\Big).$$

Definine $\gamma = \delta^2/M_j^2$, and let $p = p_j$ for shorthand. Since $X_t \le 1$, $\mathbb{E}[X_t] \le p$, and $\mathrm{Var}(X_t) \le p(1-p)$, Bernstein's inequality implies that for any fixed t and any $\epsilon \ge 0$

$$\mathbb{P}\Big(\sum_{\tau=t}^{t+m-1} X_t \ge pm + \epsilon\Big) \le \exp\left(-\frac{\epsilon^2}{2mp(1-p) + 2\epsilon/3}\right). \qquad (29)$$

By solving a quadratic, we find that if

$$\epsilon \geq \frac{1}{3} \log \frac{1}{\delta} + \sqrt{\frac{1}{9} \log^2 \frac{1}{\delta} + 2mp(1-p) \log \frac{1}{\delta}}$$

then the quantity (29) is bounded by δ. By a union bound (and minor simplification), we find

$$\epsilon \geq \frac{2}{3} \log \frac{1}{\delta} + \sqrt{2mp(1-p) \log \frac{1}{\delta}} \quad \text{implies} \quad \mathbb{P}(\mathcal{E}^c) \leq T\delta.$$

Setting $\delta = T^{-2}$ means that $\mathbb{P}(\mathcal{E}^c) \leq 1/T$, which in turn implies that

$$\mathbb{E}[\mathbf{1}\{\mathcal{E}^c\}]^{\frac{1}{2}} \mathbb{E}\left[\left(\sum_{t=1}^T |g_j(t)|\right)^2\right]^{\frac{1}{2}} \leq \frac{1}{\sqrt{T}} \sqrt{T} \mathbb{E}\left[\sum_{t=1}^T g_j(t)^2\right]^{\frac{1}{2}}.$$

Combining the preceding display with inequality (28), we find that the term $\mathcal{T}_1$ from expression (22) is bounded by

$$\mathbb{E}[\mathcal{T}_1] \leq \sum_{j=1}^d \left(2\mathbb{E}\left[\left(\sum_{t=1}^T g_j(t)^2\right)^{\frac{1}{2}}\right] + \mathbb{E}\left[\sum_{t=1}^T g_j(t)^2\right]^{\frac{1}{2}}\right)$$

whenever $\delta^2 \geq \frac{4}{3} \log T + 2\sqrt{mp_j(1-p_j) \log T}$ for all $j \in [d]$. This completes the sharper proof for the bound on $\mathcal{T}_1$. To provide a similar bound for $\mathcal{T}_2$ in analogy to Lemma 11, we recall the bound (27). Then following the above steps, *mutatis mutandis*, gives the desired result. □