Adaptive Sampling Probabilities for Non-Smooth Optimization

Hongseok Namkoong 1  Aman Sinha 2  Steve Yadlowsky 2  John C. Duchi 2,3

Abstract

Standard forms of coordinate and stochastic gradient methods do not adapt to structure in data; their good behavior under random sampling is predicated on uniformity in data. When gradients in certain blocks of features (for coordinate descent) or examples (for SGD) are larger than others, there is a natural structure that can be exploited for quicker convergence. Yet adaptive variants often suffer nontrivial computational overhead. We present a framework that discovers and leverages such structural properties at a low computational cost. We employ a bandit optimization procedure that "learns" probabilities for sampling coordinates or examples in (non-smooth) optimization problems, allowing us to guarantee performance close to that of the optimal stationary sampling distribution. When such structures exist, our algorithms achieve tighter convergence guarantees than their non-adaptive counterparts, and we complement our analysis with experiments on several datasets.

1. Introduction

Identifying and adapting to structural aspects of problem data can often improve performance of optimization algorithms. In this paper, we study two forms of such structure: variance in the relative importance of different features and observations (as well as blocks thereof). As a motivating concrete example, consider the \( \ell_p \) regression problem

\[
\min_x \ \left\{ f(x) := \|Ax - b\|_p^p = \sum_{i=1}^n |a_i^T x - b_i|^p \right\}, \tag{1}
\]

where \( a_i \) denote the rows of \( A \in \mathbb{R}^{n \times d} \). When the columns (features) of \( A \) have highly varying norms—say because certain features are infrequent—we wish to leverage this during optimization. Likewise, when rows \( a_i \) have disparate norms, "heavy" rows of \( A \) influence the objective more than others. We develop optimization algorithms that automatically adapt to such irregularities for general non-smooth convex optimization problems.

Standard (stochastic) subgradient methods (Nemirovski et al., 2009), as well as more recent accelerated variants for smooth, strongly convex incremental optimization problems (e.g. Johnson and Zhang, 2013; Defazio et al., 2014), follow deterministic or random procedures that choose data to use to compute updates in ways that are oblivious to conditioning and structure. As our experiments demonstrate, choosing blocks of features or observations—for instance, all examples belonging to a particular class in classification problems—can be advantageous. Adapting to such structure can lead to substantial gains, and we propose a method that adaptively updates the sampling probabilities from which it draws blocks of features/observations (columns/rows in problem (1)) as it performs subgradient updates. Our method applies to both coordinate descent (feature/column sampling) and mirror descent (observation/row sampling). Heuristically, our algorithm learns to sample informative features/observations using their gradient values and requires overhead only logarithmic in the number of blocks over which it samples. We show that our method optimizes a particular bound on convergence, roughly sampling from the optimal stationary probability distribution in hindsight, and leading to substantial improvements when the data has pronounced irregularity.

When the objective \( f(\cdot) \) is smooth and the desired solution accuracy is reasonably low, (block) coordinate descent methods are attractive because of their tractability (Nesterov, 2012; Necoara et al., 2011; Beck and Tetruashvili, 2013; Lee and Sidford, 2013; Richtárik and Takáč, 2014; Lu and Xiao, 2015). In this paper, we consider potentially non-smooth functions and present an adaptive block coordinate descent method, which iterates over \( b \) blocks of coordinates, reminiscent of AdaGrad (Duchi et al., 2011). Choosing a good sampling distribution for coordinates in coordinate descent procedures is nontrivial (Lee and Sidford, 2013; Necoara et al., 2011; Shalev-Shwartz and Zhang, 2012; Richtárik and Takáč, 2015; Csiba et al., 2015; Allen-Zhu and Yuan, 2015). Most work focuses on choos-
Adaptive Sampling Probabilities for Non-Smooth Optimization

ing a good stationary distribution using problem-specific knowledge, which may not be feasible; this motivates automatically adapting to individual problem instances. For example, Csiba et al. (2015) provide an updating scheme for the probabilities in stochastic dual ascent. However, the update requires $O(b)$ time per iteration, making it impractical for large-scale problems. Similarly, Nutini et al. (2015) observe that the Gauss-Southwell rule (choosing the coordinate with maximum gradient value) achieves better performance, but this also requires $O(b)$ time per iteration. Our method roughly emulates this behavior via careful adaptive sampling and bandit optimization, and we are able to provide a number of *a posteriori* optimality guarantees.

In addition to coordinate descent methods, we also consider the finite-sum minimization problem

$$\min_{x \in \mathcal{X}} \frac{1}{n} \sum_{i=1}^{n} f_i(x),$$

where the $f_i$ are convex and may be non-smooth. Variance-reduction techniques for finite-sum problems often yield substantial gains (Johnson and Zhang, 2013; Defazio et al., 2014), but they generally require smoothness. More broadly, importance sampling estimates (Strohmer and Vershynin, 2009; Needell et al., 2014; Zhao and Zhang, 2014; 2015; Csiba and Richtárik, 2016) can yield improved convergence, but the only work that allows online, problem-specific adaptation of sampling probabilities of which we are aware is Gopal (2016). However, these updates require $O(b)$ computation and do not have optimality guarantees.

We develop these ideas in the coming sections, focusing first in Section 2 on adaptive procedures for (non-smooth) coordinate descent methods and developing the necessary bandit optimization and adaptivity machinery. In Section 3, we translate our development into convergence results for finite-sum convex optimization problems. Complementing our theoretical results, we provide a number of experiments in Section 4 that show the importance of our algorithmic development and the advantages of exploiting block structures in problem data.

2. Adaptive sampling for coordinate descent

We begin with the convex optimization problem

$$\min_{x \in \mathcal{X}} f(x),$$

where $\mathcal{X} = \mathcal{X}_1 \times \cdots \times \mathcal{X}_b \subset \mathbb{R}^d$ is a Cartesian product of closed convex sets $\mathcal{X}_j \subset \mathbb{R}^{d_j}$ with $\sum_j d_j = d$, and $f$ is convex and Lipschitz. When there is a natural block structure in the problem, some blocks have larger gradient norms than others, and we wish to sample these blocks more often in the coordinate descent algorithm. To that end, we develop an adaptive procedure that exploits variability in block “importance” online. In the coming sections, we show that we obtain certain near-optimal guarantees, and that the computational overhead over a simple random choice of block $j \in [b]$ is at most $O(\log b)$. In addition, under some natural structural assumptions on the blocks and problem data, we show how our adaptive sampling scheme provides convergence guarantees polynomially better in the dimension than those of naive uniform sampling or gradient descent.

**Notation for coordinate descent** Without loss of generality we assume that the first $d_1$ coordinates of $x \in \mathbb{R}^d$ correspond to $\mathcal{X}_1$, the second $d_2$ to $\mathcal{X}_2$, and so on. We let $U_j \in \{0,1\}^{d \times d_j}$ be the matrix identifying the $j$th block, so that $I_d = U_1 \cdots U_d$. We define the projected subgradient vectors for each block $j$ by

$$G_j(x) = U_j U_j^T f'(x) \in \mathbb{R}^{d_j},$$

where $f'(x) \in \partial f(x)$ is a fixed element of the subdifferential $\partial f(x)$. Define $x_{[j]} := U_j^T x \in \mathbb{R}^{d_j}$ and $G_{[j]}(x) = U_j^T G_j(x) = U_j^T f'(x) \in \mathbb{R}^{d_j}$. Let $\psi_j$ denote a differentiable 1-strongly convex function on $\mathcal{X}_j$ with respect to the norm $\|\cdot\|_j$, meaning for all $\Delta \in \mathbb{R}^{d_j}$ we have

$$\psi_j(x_{[j]} + \Delta) \geq \psi_j(x_{[j]}) + \nabla \psi_j(x_{[j]})^T \Delta + \frac{1}{2} \|\Delta\|_j^2,$$

and let $\|\cdot\|_{j,*}$ be the dual norm of $\|\cdot\|_j$. Let $B_j(u,v) = \psi_j(u) - \psi_j(v) - \nabla \psi_j(v)^T (u-v)$ be the Bregman divergence associated with $\psi_j$, and define the tensorized divergence $B(x,y) := \sum_{j=1}^{b} B_j(x_{[j]},y_{[j]})$. Throughout the paper, we assume the following.

**Assumption 1.** For all $x, y \in \mathcal{X}$, we have $B(x,y) \leq R^2$ and $\|G_{[j]}(x)\|_{j,*}^2 \leq L^2 / b$ for $j = 1, \ldots, b$.

2.1. Coordinate descent for non-smooth functions

The starting point of our analysis is the simple observation that if a coordinate $J \in [b]$ is chosen according to a probability vector $p > 0$, then the importance sampling estimator

$$G_J(x)/p_J \text{ satisfies } \mathbb{E}_p[G_J(x)/p_J] = f'(x) \in \partial f(x).$$

Thus the randomized coordinate subgradient method of Algorithm 1 is essentially a stochastic mirror descent method (Nemirovski and Yudin, 1983; Beck and Teboulle, 2003; Nemirovski et al., 2009), and as long as $\sup_{x \in \mathcal{X}} \mathbb{E}[\|p_{[j]}^{-1} G_J(x)\|^2] < \infty$ it converges at rate $O(1/\sqrt{T})$. With this insight, a variant of standard stochastic mirror descent analysis yields the following convergence guarantee for Algorithm 1 with non-stationary probabilities (cf. Dang and Lan (2015), who do not quite as carefully track dependence on the sampling distribution...
Maintaining the running sum there is some Shwartz In the the upper bound (2.2. Optimal stepsizes by doubling so that each gradient block fits in local memory. In practice, coordinate descent procedures can significantly O the same accuracy within scent with the same constraints and divergence solves problem (5.3) we consider sampling from the uniform dis-

As an immediate consequence, if \( p^j \geq p_{\text{min}} > 0 \) and \( \alpha_x = \frac{\sqrt{2 \min} \cdot 2 T}{R} \), then \( S(f, \bar{x}_T) \leq R L \sqrt{\frac{2}{T p_{\text{min}}} \cdots} \). To make this more concrete, we consider sampling from the uniform distribution \( p^j = \frac{1}{b} \) so that \( p_{\text{min}} = 1/b \), and assume homogeneous block sizes \( d_j = d/b \) for simplicity. Algorithm 1 solves problem (2) to \( \epsilon \)-accuracy within \( O(b R^2 L^2 / \epsilon^2) \) iterations, where each iteration approximately costs \( O(d/b) \) plus the cost of projecting into \( X_j \). In contrast, mirror descent with the same constraints and divergence \( B \) achieves the same accuracy within \( O(R^2 L^2 / \epsilon^2) \) iterations, taking \( O(d) \) time plus the cost of projecting to \( X \) per iteration. As the projection costs are linear in the number \( b \) of blocks, the two algorithms are comparable.

In practice, coordinate descent procedures can significantly outperform full gradient updates through efficient memory usage. For huge problems, coordinate descent methods can leverage data locality by choosing appropriate block sizes so that each gradient block fits in local memory.

### 2.2. Optimal stepsizes by doubling

In the upper bound (3), we wish to choose the optimal stepsizes \( \alpha_x \) that minimizes this bound. However, the term \( \sum_{t=1}^T \mathbb{E} \left[ \sum_{j=1}^b \frac{\|G_{[j]}(x^t)\|_{J_{t,j}}^2}{p_j^t} \right] \) is unknown a priori. We circumvent this issue by using the doubling trick (e.g. Shalev-Shwartz, 2012, Section 2.3.1) to achieve the best possible rate in hindsight. To simplify our analysis, we assume that there is some \( p_{\text{min}} > 0 \) such that

\[
p^j \in \Delta_b := \{ p \in \mathbb{R}_+^b : p^\top 1 = 1, p \geq p_{\text{min}} \}.
\]

Maintaining the running sum \( \sum_{t=1}^T \bar{p}^t J_{t,j} \|G_{[j]}(x^t)\|_{J_{t,j}}^2 \)

### Algorithm 1 Non-smooth Coordinate Descent

**Input:** Stepsize \( \alpha_x > 0 \), Probabilities \( p^1, \ldots, p^T \).
**Initialize:** \( x^1 = x \)
for \( t \leftarrow 1, \ldots, T \)
Sample \( J_t \sim p^t \)
Update:

\[
x^t_{[j,t]} \leftarrow \arg \min_{x \in X_j} \left\{ \frac{G_{[j]}(x^t)}{p_{[j,t]^t}} x \right\} + \frac{1}{\alpha_x} B_{J_t}(x, x^t_{[j,t]}) \}
\]
**return** \( \bar{x}_T \leftarrow \frac{1}{T} \sum_{t=1}^T x^t \)

### Algorithm 2 Stepsize Doubling Coordinate Descent

**Initialize:** \( x^1 = x, p^1 = p, k = 1 \)
while \( t \leq T \)
while \( \sum_{t=1}^t (p_{J_t})^{-2} \left\| G_{[j]}(x^t) \right\|_{J_{t,j}}^2 \leq 4^k, t \leq T \) do
Run inner loop of Algorithm 1 with \( \alpha_x = k \cdot \sqrt{2 R \left( 4^k + \frac{L^2}{b p_{\text{min}}^2} \right)^{-\frac{1}{2}}} \)
\( t \leftarrow t + 1 \)
\( k \leftarrow k + 1 \)
**return** \( \bar{x}_T \leftarrow \frac{1}{T} \sum_{t=1}^T x^t \)

requires incremental time \( O(d_{J_t}) \) at each iteration \( t \), choosing the stepsizes adaptively via Algorithm 2 only requires a constant factor of extra computation over using a fixed step size. The below result shows that the doubling trick in Algorithm 2 achieves (up to log factors) the performance of the optimal stepsize that minimizes the regret bound (3).

### Proposition 2. Under Assumption 1, Algorithm 2 achieves

\[
S(f, \bar{x}_T) \leq \frac{R}{T} \left( \sum_{t=1}^T \mathbb{E} \left[ \frac{b}{p_j^t} \sum_{j=1}^b \frac{\|G_{[j]}(x^t)\|_{J_{t,j}}^2}{p_j^t} \right] \right)^{\frac{1}{2}} + \sqrt{\frac{2}{3 b p_{\text{min}} T \log 4 \left( \frac{4 b T L^2}{p_{\text{min}}} \right)}}
\]

where \( S(f, \bar{x}_T) = \mathbb{E}[f(\bar{x}_T)] - \mathbb{E}[f(x)] \).

### 2.3. Adaptive probabilities

We now present an adaptive updating scheme for \( p^j \), the sampling probabilities. From Proposition 2, the stationary distribution achieving the smallest regret upper bound minimizes the criterion

\[
\sum_{t=1}^T \mathbb{E} \left[ \sum_{j=1}^b \frac{\|G_{[j]}(x^t)\|_{J_{t,j}}^2}{p_{J_t}} \right] = \sum_{t=1}^T \mathbb{E} \left[ \frac{\|G_{[j]}(x^t)\|_{J_{t,j}}^2}{p_{J_t}^2} \right],
\]

where the equality follows from the tower property. Since \( x^t \) depends on the \( p^j \), we view this as an online convex optimization problem and choose \( p^1, \ldots, p^T \) to minimize the regret

\[
\max_{p \in \Delta_b} \sum_{t=1}^T \mathbb{E} \left[ \sum_{j=1}^b \frac{\|G_{[j]}(x^t)\|_{J_{t,j}}^2}{p_{J_t}} \left( \frac{1}{p_{J_t}^2} - \frac{1}{p_{J_t}} \right) \right].
\]

Note that due to the block coordinate nature of Algorithm 1, we only compute \( \|G_{[j]}(x^t)\|_{J_{t,j}}^2 \) for the sampled \( j = J_t \) at each iteration. Hence, we treat this as a multi-armed bandit problem where the arms are the blocks \( j = 1, \ldots, b \) and we only observe the loss \( \|G_{[j]}(x^t)\|_{J_{t,j}}^2 / (p_{J_t}^2) \) associated with the arm \( J_t \) pulled at time \( t \).
Algorithm 3 Coordinate Descent with Adaptive Sampling

Input: Stepsize $\alpha_p > 0$, Threshold $p_{\text{min}} > 0$ with $P = \{ p \in R^b : p^T 1 = 1, p \geq p_{\text{min}} \}$

Initialize: $x^1 = x, p^1 = p$

for $t \leftarrow 1, \ldots, T$

Sample $J_t \sim p^t$

Choose $\alpha_{x,k}$ according to Algorithm 2

Update $x^t$:

$$x^{t+1}_{[j_t]} \leftarrow \min_{x \in X_{[j_t]}} \left\{ \left( \frac{G_{[j_t]}(x^t)}{p^{t}_{j_t}} \right)^2 + \frac{1}{\lambda_{x,k}} B \left( x, x_{[j_t]}^t \right) \right\}$$

Update $p^t$ for $\hat{\ell}_{t,j}(x)$ defined in (5),

$$w^{t+1} \leftarrow p^t \exp\left( - (\alpha_p \hat{\ell}_{t,j}(x^t)/p_{j_t}^t) e_{j_t} \right),$$

$$p^{t+1} \leftarrow \min_{q \in P} D_{\text{kl}} (q\|w^{t+1})$$

return $x_T \leftarrow \frac{1}{T} \sum_{t=1}^T x^t$

By using a bandit algorithm—another coordinate descent method—to update $p$, we show that our updates achieve performance comparable to the best stationary probability in $\Delta_b$ in hindsight. To this end, we first bound the regret (4) by the regret of a linear bandit problem. By convexity of $x \mapsto 1/x$ and $\frac{d}{dx} 1/x = -x^{-2}$, we have

$$\sum_{t=1}^T \mathbb{E} \left[ \sum_{j=1}^b \|G_{[j]}(x^t)\|^2_{j,*} \left( \frac{1}{p_j^t} - \frac{1}{p_j^t} \right) \right] \leq \sum_{t=1}^T \mathbb{E} \left[ \left\{ -\left\{ \|G_{[j]}(x^t)\|^2_{j,*}/(p_j^t)^2 \right\}_{j=1}^b, p^t - p \right\} \right].$$

Now, let us view the vector $(\ast)$ as the loss vector for a constrained linear bandit problem with feasibility region $\Delta_b$. We wish to apply EXP3 (due to Auer et al. (2002)) or equivalently, a 1-sparse mirror descent to $p$ with $\psi_p(p) = p \log p$ (see, for example, Section 5.3 of Bubeck and Cesa-Bianchi (2012) for the connections). However, EXP3 requires the loss values to be positive in order to be in the region where $\psi_p$ is strongly convex, so we scale our problem using the fact that $p$ and $p^t$'s are probability vectors. Namely,

$$\sum_{t=1}^T \mathbb{E} \left[ \left\{ -\left\{ \|G_{[j]}(x^t)\|^2_{j,*}/(p_j^t)^2 \right\}_{j=1}^b, p^t - p \right\} \right] \leq \sum_{t=1}^T \mathbb{E} \left[ \hat{\ell}_{t,j}(x^t), p^t - p \right],$$

where

$$\hat{\ell}_{t,j}(x) := - \|G_{[j]}(x)\|^2_{j,*} + \frac{L^2}{b p_{\text{min}}}.$$ (5)

Using scaled loss values, we perform EXP3 (Algorithm 3). Intuitively, we penalize the probability of the sampled block by the strength of the signal on the block. The scaling (5) ensures that we penalize blocks with low signal (as opposed to rewarding those with high signal) which enforces diversity in the sampled coordinates as well. In Section A.3, we will see how this scaling plays a key role in proving optimality of Algorithm 3. Here, the signal is measured by the relative size of the gradient in the block against the probability of sampling the block. This means that blocks with large “surprises”—those with higher gradient norms relative to their sampling probability—will get sampled more frequently in the subsequent iterations. Algorithm 3 guarantees low regret for the online convex optimization problem (4) which in turn yields the following guarantee for Algorithm 3.

**Theorem 3.** Under Assumption 1, the adaptive updates in Algorithm 3 with $\alpha_p = \frac{\sqrt{2b \log b}}{T}$ achieve

$$S(f, x_T) \leq \frac{6R}{T} \sum_{t=1}^T \mathbb{E} \left\{ \frac{\|G_{[j]}(x^t)\|^2_{j,*}}{p_j} \right\}$$

(a): best in hindsight

$$+ \frac{8LR}{TP_{\text{min}}} \left( \frac{T \log b}{b} \right)^{\frac{1}{2}} + \frac{2RL}{\sqrt{b} P_{\text{min}}} \log \left( \frac{4bTL^2}{p_{\text{min}}} \right).$$

(b): regret for bandit problem

where $S(f, x_T) = E[f(x_T)] - \inf_{x \in X} f(x)$.

See Section A.3 for the proof. Note that there is a trade-off in the regret bound (6) in terms of $p_{\text{min}}$: for small $p_{\text{min}}$, the first term is small, as the set $\Delta_b$ is large, but second (regret) term is large, and vice versa. To interpret the bound (6), take $p_{\text{min}} = \delta/b$ for some $\delta \in (0, 1)$. The first term dominates the remainder as long as $T \approx O(b \log b)$; we require $T \approx (bR^2L^2/e^2)$ to guarantee convergence of coordinate descent in Proposition 1, so that we roughly expect the first term in the bound (6) to dominate. Thus, Algorithm 3 attains the best convergence guarantee for the optimal stationary sampling distribution in hindsight.

2.4. Efficient updates for $p$

The updates for $p$ in Algorithm 3 can be done in $O(\log b)$ time by using a balanced binary tree. Let $D_{\text{kl}}(p\|q) := \sum_{i=1}^d p_i \log\frac{p_i}{q_i}$ denote the Kullback-Leibler divergence between $p$ and $q$. Ignoring the subscript on $t$ so that $w = w^{t+1}, p = p^t$ and $J_t$, the new probability vector $q$ is given by the minimizer of

$$D_{\text{kl}}(q\|w) \quad \text{s.t. } q^T 1 = 1, q \geq p_{\text{min}},$$ (7)

where $w$ is the previous probability vector $p$ modified only at the index $J$. We store $w$ in a binary tree, keeping values up to their normalization factor. At each node, we also store the sum of elements in the left/right subtree for
Algorithm 4 KL Projection

1: Input: $J$, $p_j$, $w_J$, mass $= \sum_i w_i$
2: $w_{\text{cand}} \leftarrow p_j \cdot \text{mass}$.
3: if $w_{\text{cand}}/(\text{mass} - w_J + w_{\text{cand}}) \leq p_{\text{min}}$ then
4: \hspace{1em} $w_{\text{cand}} \leftarrow \frac{p_{\text{min}}}{1-p_{\text{min}}} (\text{mass} - w_J)$
5: Update($w_{\text{cand}}$, $J$)

efficient sampling (for completeness, the pseudo-code for sampling from the binary tree in $O(\log b)$ time is given in Section B.3). The total mass of the tree can be assessed by inspecting the root of the tree alone.

The following proposition shows that it suffices to touch at most one element in the tree to do the update. See Section B for the proof.

**Proposition 4.** The solution to (7) is given by

$$q_{j \neq J} = \begin{cases} \frac{1}{1-p_J}\frac{w_J}{w_J} & \text{if } w_J \geq \frac{p_{\text{min}}(1-p_J)}{1-p_{\text{min}}} \\ \frac{1-p_J}{1-p_J} & \text{otherwise} \end{cases},$$

$$q_J = \begin{cases} \frac{1}{1-p_J}\frac{w_J}{w_J} & \text{if } w_J \geq \frac{p_{\text{min}}(1-p_J)}{1-p_{\text{min}}} \\ \frac{1-p_J}{1-p_J} & \text{otherwise}. \end{cases}$$

As seen in Algorithm 4, we need to modify at most one element in the binary tree. Here, the update function modifies the value at index $J$ and propagates the values up the tree so that the sum of left/right subtrees are appropriately updated. We provide the full pseudocode in Section B.2.

### 2.5. Example

The optimality guarantee given in Theorem 3 is not directly interpretable since the term $a$ in the upper bound (6) is only optimal given the iterates $x^1, \ldots, x^T$ despite the fact that $x^t$’s themselves depend on the sampling probabilities. Hence, we now study a setting where we can further bound (6) to get a explicit regret bound for Algorithm 3 that is provably better than non-adaptive counterparts. Indeed, under certain structural assumptions on the problem similar to those of McMahan and Streeter (2010) and Duchi et al. (2011), our adaptive sampling algorithm provably achieves regret polynomially better in the dimension than either using a uniform sampling distribution or gradient descent.

Consider the SVM objective

$$f(x) = \frac{1}{n} \sum_{i=1}^n (1 - y_i z_i^T x),$$

where $n$ is small and $d$ is large. Here, $\partial f(x) = \frac{1}{n} \sum_{i=1}^n (1 - y_i z_i^T x) \geq 0$ $z_i$. Assume that for some fixed $\alpha \in (1, \infty)$ and $L_j := \beta^j - \alpha$, we have $|\partial_j f(x)|^2 \leq \frac{1}{n} \sum_{i=1}^n |z_i|^2 \leq L_j$. In particular, this is the case if we have sparse features $z_U \in \{-1, 0, +1\}^d$ with power law tails $P(|z_U| = 1) = \beta j^{-\alpha}$ where $U$ is a uniform random variable over $\{1, \ldots, n\}$.

Take $C_j = \{j\}$ for $j = 1, \ldots, d$ (and $b = d$). First, we show that although for the uniform distribution $p = 1/d$

$$\sum_{j=1}^d \frac{\mathbb{E}[|G_j(x^t)|^2]}{1/d} \leq d \sum_{j=1}^d L_j^2 = O(d \log d),$$

the term $(a)$ in (6) can be orders of magnitude smaller.

**Proposition 5.** Let $b = d$, $p_{\text{min}} = \delta/d$ for some $\delta \in (0, 1)$, and $C_j = \{j\}$. If $\|G_j(x)\|^2 \leq L_j^2 := \beta^j - \alpha$ for some $\alpha \in (1, \infty)$, then

$$\min_{p \in \mathbb{A}, p \geq p_{\text{min}}} \sum_{j=1}^d \frac{\mathbb{E}[|G_j(x^t)|^2]}{p_j} = \begin{cases} O(\log d), & \text{if } \alpha \in [2, \infty) \\ O(d^{2-2\alpha}), & \text{if } \alpha \in (1, 2). \end{cases}$$

We defer the proof of the proposition to Section A.5. Using this bound, we can show explicit regret bounds for Algorithm 3. From Theorem 3 and Proposition 5, we have that Algorithm 3 attains

$$S(f, \bar{x}_T) \leq \begin{cases} O(\frac{R \log d}{\sqrt{T}}), & \text{if } \alpha \in [2, \infty) \\ R \sqrt{T} O(d^{1/2}), & \text{if } \alpha \in (1, 2) \\ + O \left(Rd^{3/4}T^{-3/4} \log^{5/4} d \right). \end{cases}$$

Setting above to be less than $\epsilon$ and inverting with respect to $T$, we obtain the iteration complexity in Table 1.

To see the runtime bounds for uniformly sampled coordinate descent and gradient descent, recall the regret bound (3) given in Proposition 1. Plugging $p_j = 1/d$ in the bound, we obtain

$$S(f, \bar{x}_T) \leq O(R \sqrt{\log d \sqrt{2d}T}).$$

for $\alpha_x = \sqrt{2R^2/(L^2Td)}$ where $L^2 := \sum_{j=1}^d L_j^2$. Similarly, gradient descent with $\alpha_x = \sqrt{2R^2/(L^2T)}$ attains

$$S(f, \bar{x}_T) \leq O(R \sqrt{\log d \sqrt{2T}}).$$

Since each gradient descent update takes $O(d)$, we obtain the same runtime bound.
While non-adaptive algorithms such as uniformly-sampled coordinate descent or gradient descent have the same runtime for all \(\alpha\), our adaptive sampling method automatically tunes to the value of \(\alpha\). Note that for \(\alpha \in (1, \infty)\), the first term in the runtime bound for our adaptive method given in Table 1 is strictly better than that of uniform coordinate descent or gradient descent. In particular, for \(\alpha \in [2, \infty)\) the best stationary sampling distribution in hindsight yields an improvement that is at most \(O(d)\) better in the dimension. However, due to the remainder terms for the bandit problem, this improvement only matters (i.e. first term is larger than second) when

\[
\epsilon = \begin{cases} 
O \left( \frac{Rd^2}{\alpha} \sqrt{\log d} \right) & \text{if } \alpha \in [2, \infty) \\
O \left( \frac{Rd^3}{\alpha} (1-\alpha) \log^{-5/2} d \right) & \text{if } \alpha \in (1, 2).
\end{cases}
\]

In Section 4, we show that these remainder terms can be made smaller than what their upper bounds indicate. Empirically, our adaptive method outperforms the uniformly-sampled counterpart for larger values of \(\epsilon\) than above.

### 3. Adaptive probabilities for stochastic gradient descent

Consider the empirical risk minimization problem

\[
\text{minimize } \frac{1}{n} \sum_{i=1}^{n} f_i(x) =: f(x)
\]

where \(X \subset \mathbb{R}^d\) is a closed convex set and \(f_i(\cdot)\) are convex functions. Let \(C_1, \ldots, C_k\) be a partition of the \(n\) samples so that each example belongs to some \(C_j\), a set of size \(n_j := |C_j|\) (note that the index \(j\) now refers to blocks of examples instead of coordinates). These block structures naturally arise, for example, when \(C_j\)'s are the examples with the same label in a multi-class classification problem. In this stochastic optimization setting, we now sample a block \(J_t \sim p^t\) at each iteration \(t\), and perform gradient updates using a gradient estimate on the block \(C_{J_t}\). We show how a similar adaptive updating scheme for \(p^t\)'s again achieves the optimality guarantees given in Section 2.

#### 3.1. Mirror descent with non-stationary probabilities

Following the approach of (Nemirovski et al., 2009), we run mirror descent for the updates on \(x\). At iteration \(t\), a block \(J_t\) is drawn from a \(b\)-dimensional probability vector \(p^t\). We assume that we have access to unbiased stochastic gradients \(G_j(x)\) for each block. That is, \(\mathbb{E}[G_j(x)] = \frac{1}{n_j} \sum_{i \in C_j} \partial f_i(x)\). In particular, the estimate \(G_{J_t}(x^t) := \partial f_{J_t}(x)\) where \(J_t\) is drawn uniformly in \(C_{J_t}\) gives the usual unbiased stochastic gradient of minibatch size 1. The other extreme is obtained by using a minibatch size of \(n_j\) where \(G_{J_t}(x^t) := \frac{1}{n_j} \sum_{i \in C_{J_t}} \partial f_i(x)\). Then, the importance sampling estimator \(\frac{n_{J_t}}{n_{p^t_{J_t}}} G_{J_t}(x^t)\) is an unbiased estimate for the subgradient of the objective.

Let \(\psi\) be a differentiable 1-strongly convex function on \(X\) with respect to the norm \(\|\cdot\|\) as before and denote by \(\|\cdot\|\) the dual norm of \(\|\cdot\|\). Let \(B(x, y) = \psi(x) - \psi(y) - \nabla \psi(y)^\top (x-y)\) be the Bregman divergence associated with \(\psi\). In this section, we assume the below (standard) bound.

**Assumption 2.** For all \(x, y \in X\), we have \(B(x, y) \leq R^2\) and \(\|G_j(x)\|^2 \leq L_j\) for \(j = 1, \ldots, b\).

We use these stochastic gradients to perform mirror updates, replacing the update in Algorithm 1 with the update

\[
x_{t+1}^j = \argmin_{x \in X} \left\{ \frac{n_{J_t}}{n_{p^t_{J_t}}} \langle G_{J_t}(x^t), x \rangle + \frac{1}{\alpha_j} B(x, x^t) \right\}.
\]

From a standard argument (e.g., (Nemirovski et al., 2009)), we obtain the following convergence guarantee. The proof follows an argument similar to that of Proposition 1.

**Proposition 6.** Under Assumption 2, the updates (8) attain

\[
S(f, \bar{x}_T) \leq \frac{R^2}{\alpha_j T} + \frac{\alpha_j}{2T} \sum_{t=1}^{T} \mathbb{E} \left[ \sum_{j=1}^{b} \frac{n_j^2}{n_{p^t_{J_t}}} \|G_j(x^t)\|^2 \right].
\]

where \(S(f, \bar{x}_T) = \mathbb{E}[f(\bar{x}_T)] - \inf_{x \in X} f(x)\).

Again, we wish to choose the optimal step size \(\alpha_j\) that minimizes the regret bound (9). To this end, modify the doubling trick given in Algorithm 2 as follows: use \(\sum_{t=1}^{T} \frac{n_j^2}{n_{p^t_{J_t}}} \|G_j(x^t)\|^2\) for the second while condition, and stepsizes \(\alpha_{x,k} = \sqrt{2R} \left( 4^k + L^2 \max_j n_j^2 \right)^{-\frac{1}{2}}\). Then, similar to Proposition 2, we have

\[
S(f, \bar{x}_T) \leq 6 \frac{R}{T} \left( \sum_{t=1}^{T} \mathbb{E} \left[ \sum_{j=1}^{b} \frac{n_j^2}{n_{p^t_{J_t}}} \|G_j(x^t)\|^2 \right] \right)^{\frac{1}{2}} + \frac{\sqrt{2RL}}{p_{\min}^T \log 4} \max_j \frac{n_j}{n} \log \left( \frac{4TL^2}{p_{\min}} \sum_{j=1}^{b} \frac{n_j^3}{n^2} \right).
\]

#### 3.2. Adaptive probabilities

Now, we consider an adaptive updating scheme for \(p^t\)'s similar to Section 2.3. Using the scaled gradient estimate

\[
\hat{\partial}_{t,J}(x) := -\left( \frac{n_{J_t}}{n_{p^t_{J_t}}} \|G_j(x)\|_2 \right)^2 + \frac{L^2 \max_j n_j^2}{n_{p^t_{J_t}}} \frac{1}{n_{p^t_{J_t}}} \|G_j(x)\|_2^2
\]

to run EXP3, we obtain Algorithm 5. Again, the additive scaling \(L^2 \max_j n_j^2/n_{p_{\min}}^2\) is to ensure that \(\hat{\partial} \geq 0\). As in Section 2.4, the updates for \(p^t\) in Algorithm 5 can be done in \(O(\log b)\) time. We can also show similar optimality guarantees for Algorithm 5 as before. The proof is essentially the same to that given in Section A.3.
Adaptive Sampling Probabilities for Non-Smooth Optimization

Algorithm 5 Mirror Descent with Adaptive Sampling

Input: Stepsize $\alpha_p > 0$
Initialization: $x^1 = x, p^1 = p$

for $t \leftarrow 1, \ldots, T$

Sample $J_t \sim p_t$

Choose $\alpha_{x,t} \in \mathcal{X}$ according to (modified) Algorithm 2.

Update $x$:

$$x_{t+1} = \arg \min_{x \in \mathcal{X}} \left\{ \frac{1}{\alpha_{x,t}} \langle G_{J_t}(x_t), x \rangle + \frac{1}{\alpha_{x,t}} B(x, x_{J_t}) \right\}$$

Update $p_t$:

$$p_{t+1} \leftarrow p_t \exp\left(-(\alpha_{p,t} \hat{G}_{J_t}(x^t)/p_{t,J_t})c_{J_t}\right)$$

return $\hat{x}_T \leftarrow \frac{1}{T} \sum_{t=1}^{T} x^t$

Theorem 7. Let $W := \frac{L \max \{n_j \}}{\min \{n_j \}}$. Under Assumption 2, Algorithm 5 with $\alpha_p = \frac{1}{W^2 \sqrt{2 \log b}}$ achieves

$$S(f, \bar{x}_T) \leq \frac{6R}{T} \min_{p \in \Delta_b} \sum_{t=1}^{T} \mathbb{E} \left[ \sum_{j=1}^{b} \frac{n_j^2}{n^2 p_j^2} \left\| G_{J_t}(x^t) \right\|_2^2 \right] + W(2T \log b)^{\frac{1}{2}} + \sqrt{2RW \log 4} \log \left( \frac{4TL^2}{\min_{p \in \Delta_b} \sum_{j=1}^{b} \frac{n_j^2}{n^2}} \right)$$

where $S(f, \bar{x}_T) = \mathbb{E}[f(\bar{x}_T)] - \inf_{x \in \mathcal{X}} f(x)$.

With equal block sizes $n_j = n/b$ and $p_{\min} = \delta/b$ for some $\delta \in (0, 1)$, the first term in the bound of Theorem 7 is $O(TL^2)$ which dominates the second term if $T = \Omega(b \log b)$. Since we usually have $T = \Theta(n)$ for SGD, as long as $n = \Omega(b \log b)$ we have

$$S(f, \bar{x}_T) \leq O \left( \frac{R}{T} \min_{p \in \Delta_b} \sum_{t=1}^{T} \mathbb{E} \left[ \sum_{j=1}^{b} \frac{\left\| G_{J_t}(x^t) \right\|_2^2}{p_j} \right] \right)$$

That is, Algorithm 5 attains the best regret bound achieved by the optimal stationary distribution in hindsight had the $x^t$’s had remained the same. Further, under similar structural assumptions $\left\| G_J(x) \right\|_2^2 \propto j^{-\alpha}$ as in Section 2.5, we can prove that the regret bound for our algorithm is better than that of the uniform distribution.

4. Experiments

We compare performance of our adaptive approach with stationary sampling distributions on real and synthetic datasets. To minimize parameter tuning, we fix $\alpha_p$ at the value suggested by theory in Theorems 3 and 7. However, we make a heuristic modification to our adaptive algorithm since rescaling the bandit gradient (5) and (10) dwarfs the signals in gradient values if $L$ is too large. We present performance of our algorithm with respect to multiple estimates of the Lipschitz constant $L = L/c$ for $c > 1$, where $L$ is the actual upper bound. $^1$ We tune the stepsize $\alpha_x$ for both methods, using the form $\beta/\sqrt{t}$ and tuning $\beta$.

For all our experiments, we compare our method against the uniform distribution and blockwise Lipschitz sampling distribution $p_j \propto L_j$ where $L_j$ is the Lipschitz constant of the $j$-th block (Zhao and Zhang, 2015). We observe that the latter method often performs very well with respect to iteration count. However, since computing the blockwise Lipschitz sampling distribution takes $O(nd)$, the method is not competitive in large-scale settings. Our algorithm, on the other hand, adaptively learns the latent structure and often outperforms stationary counterparts with respect to runtime. While all of our plots are for a single run with a random seed, we can reject the null hypothesis $f(x_{T}^{\text{uniform}}) < f(x_{T}^{\text{adaptive}})$ at $99\%$ confidence for all instances where our theory guarantees it. We take $\| \cdot \| = \| \cdot \|_2$ throughout this section.

4.1. Adaptive sampling for coordinate descent

Synthetic Data We begin with coordinate descent, first verifying the intuition of Section 2.5 on a synthetic dataset. We consider the problem minimize$_{\|x\|_2 \leq 1} \frac{1}{2} \| Ax - b \|_1$, and we endow $A \in \mathbb{R}^{n \times d}$ with the following block structure: the columns are drawn as $a_j \sim \mathcal{N}(0, I)$, thus, the gradients of the columns decay in a heavy-tailed manner as in Section 2.5 so that $L_j = j^{-\alpha}$. We set $n = d = b = 256; \text{the effects of changing } n/d \text{ and } b/d \text{ manifest themselves via relative norms of the gradients in the columns, which we control via } \alpha \text{ instead. We run all experiments with } p_{\min} = 0.1/b \text{ and multiple values of } c.$

Results are shown in Figure 1, where we show the optimality gap vs. runtime in (a) and the learned sampling distribution in (b). Increasing $\alpha$ (stronger block structure) improves our relative performance with respect to uniform sampling and our ability to accurately learn the underlying block structure. Experiments over more $\alpha$ and $c$ in Section C further elucidate the phase transition from uniform-like behavior to regimes learning/exploiting structure.

We also compare our method with (non-preconditioned) SGD using leverage scores $p_j \propto \| a_j \|_1$ given by (Yang et al., 2016). The leverage scores (i.e., sampling distribution proportional to blockwise Lipschitz constants) roughly correspond to using $p_j \propto j^{-\alpha/2}$, which is the stationary distribution that minimizes the bound (3); in this synthetic setting, this sampling probability coincides with the actual block structure. Although this is expensive to compute, taking $O(nd)$ time, it exploits the latent block structure very well as expected. Our method quickly learns the structure and performs comparably with this “optimal” distribution.

$^1$ We guarantee a positive loss by taking $\max(\tilde{G}_{t,j}(x), 0)$. 
Model selection  

Our algorithm’s ability to learn underlying block structure can be useful in its own right as an online feature selection mechanism. We present one example of this task, studying an aptamer selection problem (Cho et al., 2013), which consists of \( n = 2900 \) nucleotide sequences (aptamers) that are one-hot-encoded with all \( k \)-grams of the sequence, where \( 1 \leq k \leq 5 \) so that \( d = 105,476 \). We train an \( l_1 \)-regularized SVM on the binary labels, which denote (thresholded) binding affinity of the aptamer. We set the blocksize as 50 features \((b = 2110)\) and \( p_{\text{min}} = 0.01/b \). Results are shown in Figure 2, where we see that adaptive feature selection certainly improves training time in (a). The learned sampling distribution depicted in (b) for the best case \((c = 10^7)\) places larger weight on features known as \( G \)-complexes; these features are well-known to affect binding affinities (Cho et al., 2013).

4.2. Adaptive sampling for SGD

Synthetic data  

We use the same setup as in Section 4.1 but now endow block structure on the rows of \( A \) rather than the columns. In Figure 3, we see that when there is little block structure \((\alpha = 0.4)\) all sampling schemes perform similarly. When the block structure is apparent \((\alpha = 6)\), our adaptive method again learns the underlying structure and outperforms uniform sampling. We provide more experiments in Section C to illustrate behaviors over more \( c \) and \( \alpha \). We note that our method is able to handle online data streams unlike stationary methods such as leverage scores.

CUB-200-2011/ALOI  

We apply our method to two multi-class object detection datasets: Caltech-UCSD Birds-200-2011 (Wah et al., 2011) and ALOI (Geusebroek et al., 2005). Labels are used to form blocks so that \( b = 200 \) for CUB and \( b = 1000 \) for ALOI. We use softmax loss for CUB-200-2011 and a binary SVM loss for ALOI, where in the latter we do binary classification between shells and non-shell objects. We set \( p_{\text{min}} = 0.5/b \) to enforce enough exploration. For the features, outputs of the last fully-connected layer of ResNet-50 (He et al., 2016) are used for CUB so that we have 2049-dimensional features. Since our classifier \( x \) is \((b \cdot d)\)-dimensional, this is a fairly large scale problem. For ALOI, we use default histogram features \( (d = 128) \). In each case, we have \( n = 5994 \) and \( n = 108,000 \) respectively. We set \( \mathcal{X} := \{x \in \mathbb{R}^n : \|x\|_2 \leq r\} \) where \( r = 100 \) for CUB and \( r = 10 \) for ALOI. We observe in Figure 4 that our adaptive sampling method outperforms stationary counterparts.
Acknowledgements

HN was supported by the Samsung Scholarship. AS and SY were supported by Stanford Graduate Fellowships and AS was also supported by a Fannie & John Hertz Foundation Fellowship. JCD was supported by NSF-CAREER-1553086.

References


