Breaking locality accelerates block Gauss-Seidel

Stephen Tu Shivaram Venkataraman Ashia C. Wilson
Alex Gittens Michael I. Jordan Benjamin Recht

University of California, Berkeley

Abstract

Recent work by Nesterov and Stich [16] showed that momentum can be used to accelerate the rate of convergence for block Gauss-Seidel in the setting where a fixed partitioning of the coordinates is chosen ahead of time. We show that this setting is too restrictive, constructing instances where non-accelerated Gauss-Seidel which randomly samples a block of coordinates at every iteration substantially outperforms accelerated Gauss-Seidel over any fixed partitioning. Motivated by this finding, we extend the accelerated block Gauss-Seidel algorithm to the random coordinate sampling setting. Our analysis captures the benefit of acceleration with a new data-dependent parameter which is well behaved when the matrix sub-blocks are well-conditioned. Empirically, we show that accelerated Gauss-Seidel with random coordinate sampling provides speedups for large scale machine learning tasks when compared to non-accelerated Gauss-Seidel and the classical conjugate-gradient algorithm.

1 Introduction

The randomized Gauss-Seidel method is a commonly used iterative algorithm to compute the solution of an $n \times n$ linear system $Ax = b$ by updating a single coordinate at a time in a randomized order. While this approach is known to converge linearly to the true solution under the assumption that A is positive definite (see e.g. [9]), in practice it is often more efficient to update a small block of coordinates at a time due to the effects of cache locality.

In extending randomized Gauss-Seidel to the block setting, a natural question arises as to how to sample the next block. At one extreme, a *fixed partition* of the coordinates is chosen ahead of time, and the algorithm is restricted to selecting random blocks from this fixed partitioning. In this scheme, coordinates in different partitions are never updated simultaneously. At the other extreme, a new set of *random coordinates* are sampled without replacement to form a block at every iteration.

Theoretically, the fixed partition case is well understood both for Gauss-Seidel [19, 5] and its Nesterov accelerated variant [16]. More specifically, at most $O(\mu_{\text{part}}^{-1} \log(1/\varepsilon))$ iterations of Gauss-Seidel are sufficient to reach a solution with at most ε error measured in ℓ_2 norm, where μ_{part} is a quantity which measures how well the A matrix is preconditioned by the block diagonal matrix containing the sub-blocks corresponding to the fixed partitioning. When Nesterov-style acceleration is used, Nesterov and Stich [16] show that the rate improves to $O(\sqrt{\frac{n}{p}\mu_{\text{part}}^{-1}} \log(1/\varepsilon))$, where p denotes the size of each partition.

For the random coordinate selection model, the existing literature is less complete. While it is known [19, 5] that the iteration complexity with random coordinate section is $O(\mu_{\text{rand}}^{-1} \log(1/\varepsilon))$ for an ε error solution, μ_{rand} is another instance dependent quantity which is not directly comparable to μ_{part}. Hence it is not obvious how much better, if at all, one expects random coordinate selection to perform compared to fixed partitioning.

Our first contribution in this paper is to show that, when compared to the random coordinate selection model, the fixed partition model can perform very poorly in terms of iteration complexity to reach a pre-specified error. Specifically, we present a family of instances (similar to the matrices recently studied by Lee

and Wright [7]) where *non*-accelerated Gauss-Seidel with uniformly random coordinate selection performs *arbitrarily* faster than both non-accelerated and even accelerated Gauss-Seidel, using *any* fixed partition.

This finding motivates us to further study the benefits of acceleration under the random coordinate selection model. Interestingly, the benefits are more nuanced under this model. We show that acceleration improves the rate from $O(\mu_{\text{rand}}^{-1} \log(1/\varepsilon))$ to $O(\sqrt{\nu \mu_{\text{rand}}^{-1}} \log(1/\varepsilon))$, where ν is a new instance dependent quantity that satisfies $\nu \leq \mu_{\text{rand}}^{-1}$. We derive a bound on ν which suggests that if the sub-blocks of A are all well conditioned, then acceleration can provide substantial speedups. We note that this is merely a sufficient condition, and our experiments suggest that our bound is conservative.

In the process of deriving our results, we also develop a general proof framework for randomized accelerated methods based on Wilson et al. [29] which avoids the use of estimate sequences in favor of an explicit Lyapunov function. Using our proof framework we are able to recover recent results [16, 1] on accelerated coordinate descent. Furthermore, our proof framework allows us to immediately transfer our results on Gauss-Seidel over to the randomized accelerated Kaczmarz algorithm, extending a recent result by Liu and Wright [11] on updating a single constraint at a time to the block case.

Finally, we empirically demonstrate that despite its theoretical nuances, accelerated Gauss-Seidel using random coordinate selection can provide significant speedups in practical applications over Gauss-Seidel with fixed partition sampling, as well as the classical conjugate-gradient (CG) algorithm. As an example, for a kernel ridge regression (KRR) task in machine learning on the CIFAR-10 dataset, acceleration with random coordinate sampling performs up to $2\times$ faster than acceleration with a fixed partitioning to reach an error tolerance of 10^{-2}, with the gap substantially widening for smaller error tolerances. Furthermore, it performs over $5\times$ faster than conjugate-gradient on the same task.

2 Background

In this section, we develop the relevant context to explain our results in more detail. For the remainder of this paper, we assume that we are given an $n \times n$ matrix A which is positive definite, and an n dimensional response vector b. We also fix an integer p which denotes a block size. To avoid special edge cases we will assume that $1 < p < n$. We will also assume for simplicity that n is an integer multiple of p.

Under the assumption of A being positive definite, the function $f : \mathbb{R}^n \longrightarrow \mathbb{R}$ defined as

$$f(x) = \frac{1}{2} x^\mathsf{T} A x - x^\mathsf{T} b \,, \tag{2.1}$$

is a strongly convex and smooth function on $\mathbb{R}^n$. Recent analysis of Gauss-Seidel [5] proceeds by noting the connection between Gauss-Seidel and (block) coordinate descent on f. This is the point of view we will take in this paper.

Notation. We summarize here the standard notation used throughout this paper. $[n] = \{1, 2, ..., n\}$ refers to the set of integers from 1 to n, and $2^{[n]}$ refers to the set of all subsets of $[n]$. We let $\mathbf{1}_n \in \mathbb{R}^n$ denote the vector of all ones. Given a square matrix M with real eigenvalues, we let $\lambda_{\max}(M)$ (resp. $\lambda_{\min}(M)$) denote the maximum (resp. minimum) eigenvalue of M. For two symmetric matrices M, N, the notation $M \succcurlyeq N$ (resp. $M \succ N$) means that the matrix $M - N$ is positive semidefinite (resp. positive definite). Every such $M \succ 0$ defines a real inner product space via the inner product $\langle x, y \rangle_M = x^\mathsf{T} M y$. We refer to its induced norm as $\|x\|_M = \sqrt{\langle x, x \rangle_M}$. The standard Euclidean inner product and norm will be denoted as $\langle \cdot, \cdot \rangle$ and $\|\cdot\|_2$, respectively. For an arbitrary matrix M, we let $M^\dagger$ denote its Moore-Penrose pseudo-inverse and P_M the orthogonal projector onto the range of M, which we denote as $\mathcal{R}(M)$. When $M \succcurlyeq 0$, we let $M^{1/2}$ denote its unique Hermitian square root. Finally, for a square $n \times n$ matrix M, $\text{diag}(M)$ is the $n \times n$ diagonal matrix which contains the diagonal elements of M.

Partitions on $[n]$. In what follows, whenever we discuss a partition of $[n]$, we will assume, without loss of generality, that the partition is given by $\bigcup_{i=1}^{n/p} J_i$, where

$$J_1 = \{1, 2, ..., p\}\ ,\ \ J_2 = \{p+1, p+2, ..., 2p\},\ \ ...\ .$$

This is without loss of generality because for any arbitrary equal sized partition of $[n]$, there exists a permutation matrix Π such that all our results apply by the change of variables $A \leftarrow \Pi^\mathsf{T} A \Pi$ and $b \leftarrow \Pi^\mathsf{T} b$.

2.1 Existing rates for randomized block Gauss-Seidel

We first describe the sketching framework of [19, 5] and show how it yields rates on Gauss-Seidel when blocks are chosen via a fixed partition or randomly at every iteration. While the sketching framework is very general, in this paper we will only focus on the special case when the sketch matrix represents column sampling. We introduce the sketching framework solely because it allows us to provide a unified analysis of both cases; many of our structural results will be stated in terms of the sketching framework.

To be more precise, let $\mathcal{D}$ be a distribution over $\mathbb{R}^{n\times p}$, and let $S_k \sim \mathcal{D}$ be drawn iid from $\mathcal{D}$. If we perform block coordinate descent by minimizing f along the range of S_k, then the randomized block Gauss-Seidel update is given by

$$x_{k+1} = x_k - S_k(S_k^\mathsf{T} A S_k)^\dagger S_k^\mathsf{T}(Ax_k - b)\ ,\ \ k = 0, 1, ...\ . \tag{2.2}$$

Column sampling. We now describe the specialization of (2.2) to column sampling. Every index set $J \subseteq 2^{[n]}$ with $|J| = p$ induces a sketching matrix $S(J) = (e_{J(1)}, ..., e_{J(p)})$ where e_i denotes the i-th standard basis vector in $\mathbb{R}^n$, and $J(1), ..., J(p)$ is any ordering of the elements of J [1].

By equipping different probability measures on $2^{[n]}$ one can easily describe fixed partition sampling as well as random coordinate sampling (and many other sampling schemes). The former puts uniform mass on the index sets $J_1, ..., J_{n/p}$, whereas the latter puts uniform mass on all $\binom{n}{p}$ index sets of size p. Furthermore, in the sketching framework there is no limitation to use a uniform distribution, nor is there any limitation to use a fixed p for every iteration. For this paper, however, we will restrict our attention to these cases.

Existing rates. Under the assumptions stated above, [19, 5] show that that for every $k \geq 0$, the iteration (2.2) satisfies,

$$\mathbb{E}[\|x_k - x_*\|_A] \leq (1-\mu)^{k/2}\|x_0 - x_*\|_A,\ \ \mu = \lambda_{\min}(\mathbb{E}[P_{A^{1/2}S}])\ . \tag{2.3}$$

The expectation on the left hand side is taken with respect to the randomness of $S_0, S_1, ...$, and the expectation in the definition of μ is taken with respect to $S \sim \mathcal{D}$. Under both fixed partitioning and random coordinate selection, $\mu > 0$ is guaranteed (see e.g. [5], Lemma 4.3). Thus, (2.2) achieves a linear rate of convergence to the true solution, with the rate governed by the μ quantity shown above.

We now specialize (2.3) to fixed partitioning and random coordinate sampling, and provide some intutition for why we expect the latter to outperform the former in terms of iteration complexity. We first consider the case when the sampling distribution corresponds to fixed partitioning. Here, $\mu = \mu_{\text{part}}$ corresponds to measuring how effectively the block diagonal of A preconditions A, that is,

$$\mu_{\text{part}} = \frac{p}{n}\lambda_{\min}\left(A\begin{bmatrix} A_{J_1}^{-1} & 0 & ... & 0 \\ 0 & A_{J_2}^{-1} & ... & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & ... & A_{J_{n/p}}^{-1}\end{bmatrix}\right)\ . \tag{2.4}$$

[1] Note that the ordering is irrelevant because the quantity $S(J)(S(J)^\mathsf{T} A S(J))^\dagger S(J)^\mathsf{T}$ is invariant under any ordering.

Above, A_{J_i} denotes the $p \times p$ matrix corresponding to the sub-matrix of A indexed by the i-th partition. A loose lower bound on μ_{part} is

$$\mu_{\text{part}} \geq \frac{p}{n} \frac{\lambda_{\min}(A)}{\max_{1 \leq i \leq n/p} \lambda_{\max}(A_{J_i})} . \tag{2.5}$$

A preconditioning interpretation for $\mu = \mu_{\text{rand}}$ is less clear in the random coordinate case. However, [19] derives a lower bound on $\mu = \mu_{\text{rand}}$ as

$$\mu_{\text{rand}} \geq \frac{p}{n} \left(\frac{p-1}{n-1} + \left(1 - \frac{p-1}{n-1} \right) \frac{\max_{1 \leq i \leq n} A_{ii}}{\lambda_{\min}(A)} \right)^{-1} . \tag{2.6}$$

Using the upper bounds from (2.5) to (2.6) we can upper bound the iteration complexity of fixed partition Gauss-Seidel and random coordinate Gauss-Seidel as

$$T_{\text{part}} \leq O\left(\frac{n}{p} \frac{\max_{1 \leq i \leq n/p} \lambda_{\max}(A_{J_i})}{\lambda_{\min}(A)} \log(1/\varepsilon) \right) , \tag{2.7}$$

$$T_{\text{rand}} \leq O\left(\frac{n}{p} \frac{\max_{1 \leq i \leq n} A_{ii}}{\lambda_{\min}(A)} \log(1/\varepsilon) \right) . \tag{2.8}$$

Comparing the bounds on (2.7) to (2.8), it is not unreasonable to expect random coordinate sampling has better iteration complexity than fixed partition sampling in certain cases. In Section 4, we will construct such an example and we also experimentally verify this in Section 5.

2.2 Accelerated rates for fixed partition Gauss-Seidel

We now discuss previous work on accelerating Gauss-Seidel. Recently, there is a growing body of literature on accelerated coordinate descent methods (we review this literature more thoroughly in Section 3). Based on the interpretation of Gauss-Seidel as block coordinate descent on the function f, we can use an existing result and recover an accelerated algorithm for Gauss-Seidel when using fixed partitions. Specifically, we instantiate Theorem 1 of Nesterov and Stich [16] to recover a procedure and a rate for accelerating (2.2). The procedure is described in Algorithm 1, and the specific details to recover Algorithm 1 are described in Section B.2.3.

Algorithm 1 Accelerated randomized block Gauss-Seidel for fixed partitions [16].

Require: $A \in \mathbb{R}^{n \times n}$, $A \succ 0$, $b \in \mathbb{R}^n$, $x_0 \in \mathbb{R}^n$, block size p, μ_{part} defined in (2.4).

Set $A_0 = 0, B_0 = 1$.
Set $\sigma = \frac{n}{p} \mu_{\text{part}}$.
Set $y_0 = z_0 = x_0$.
for $k = 0, ..., T-1$ **do**
  $i_k \leftarrow$ uniform from $\{1, 2, ..., n/p\}$.
  $S_k \leftarrow$ column selector associated with partition J_{i_k}.
  $a_{k+1} \leftarrow$ positive solution to $a_{k+1}^2 (n/p)^2 = (A_k + a_{k+1})(B_k + \sigma a_{k+1})$.
  $A_{k+1} = A_k + a_{k+1}, B_{k+1} = B_k + \sigma a_{k+1}$.
  $\alpha_k = \frac{a_{k+1}}{A_{k+1}}, \beta_k = \sigma \frac{a_{k+1}}{B_{k+1}}$.
  $y_k = \frac{(1-\alpha_k) x_k + \alpha_k (1-\beta_k) z_k}{1 - \alpha_k \beta_k}$.
  $x_{k+1} = y_k - S_k (S_k^\mathsf{T} A S_k)^{-1} S_k^\mathsf{T} (A y_k - b)$.
  $z_{k+1} = (1 - \beta_k) z_k + \beta_k y_k - \frac{n a_{k+1}}{p B_{k+1}} S_k (S_k^\mathsf{T} A S_k)^{-1} S_k^\mathsf{T} (A y_k - b)$.
end for
return x_T.

Furthermore, Theorem 1 of [16] states that for $k \geq 0$, the iterates produced by Algorithm 1 satisfy

$$\mathbb{E}[\|x_k - x_*\|_A] \leq O\left(\left(1 - \sqrt{\frac{p}{n}\mu_{\text{part}}}\right)^{k/2} \|x_0 - x_*\|_A\right). \tag{2.9}$$

Above, μ_{part} is the same quantity defined in (2.4). Comparing (2.9) to the non-accelerated Gauss-Seidel rate given in (2.3), we see that acceleration improves the iteration complexity to reach a solution with ε error from $O(\mu_{\text{part}}^{-1} \log(1/\varepsilon))$ to $O(\sqrt{\frac{n}{p}\mu_{\text{part}}^{-1}} \log(1/\varepsilon))$, as discussed in Section 1.

3 Related work

We split the related work into two broad categories of interest: (a) work related to coordinate descent (CD) methods on convex functions and (b) randomized solvers designed for solving consistent linear systems. See [6] for classical background on numerical methods for solving linear systems. For a more recent survey on coordinate descent methods in optimization, see [30].

As mentioned in Section 2, whenever A is positive definite, Gauss-Seidel can be interpreted as an instance of coordinate descent on a strongly convex quadratic function. We therefore review related work on both non-accelerated and accelerated coordinate descent, focusing on the randomized setting instead of the more classical cyclic order or Gauss-Southwell rule for selecting the next coordinate. See [27] for a discussion on non-random selection rules.

Nesterov's original paper [15] first considered randomized CD on convex functions, assuming a partitioning of coordinates fixed ahead of time. The analysis included both non-accelerated and accelerated variants for convex functions. This work sparked a resurgence of interest in CD methods for large scale data analysis. Most relevant to our paper are extensions to the block setting [23], to handling arbitrary sampling distributions [17, 18, 4], and second order updates for quadratic functions [20].

For accelerated CD, Lee and Sidford [8] generalize the analysis of Nesterov [15]. While the analysis of [8] was limited to selecting a single coordinate at a time, several follow on works [17, 10, 12, 3] generalize to block and non-smooth settings. More recently, both Allen-Zhu et al. [1] and Nesterov and Stich [16] independently improve the results of [8] by using a somewhat surprising non-uniform sampling distribution. One of the most notable aspects of the analysis in [1] is a departure from the (probabilistic) estimate sequence framework of Nesterov (see [14] for background on estimate sequences). Instead, the authors of [1] construct a valid Lyapunov function for coordinate descent, although they do not explicitly mention this. In our work, we make this Lyapunov point of view explicit. The constants we choose in our acceleration updates arise from a particular discretization and Lyapunov function outlined from Wilson et al. [29]. Using this framework makes our proof particularly transparent, and allows us to recover results for strongly convex functions from [1] and [16] as a special case.

From the numerical analysis side, we focus on the literature for solving consistent systems. Both the Gauss-Seidel and Kaczmarz algorithm are considered classical. Strohmer and Vershynin [26] were the first to prove a linear rate of convergence for randomized Kaczmarz, and Leventhal and Lewis [9] provide a similar kind of analysis for randomized Gauss-Seidel. Both [26] and [9] were in the single constraint/coordinate setting. The block setting was later analyzed by Needell and Tropp [13]. More recently, Gower and Richtárik [5] provide a unified analysis for both randomized block Gauss-Seidel and Kaczmarz in the sketching framework, which we adopt in this paper. Finally, Liu and Wright [11] provide an accelerated analysis of randomized Kaczmarz once again in the single constraint setting. In the appendix, we describe how our results also apply for randomized Kaczmarz, and extend the results to the block setting as well.

4 Results

In this section, we describe our main results. We first present in Section 4.1 a family of instances for which Gauss-Seidel with random coordinate sampling outperforms any fixed partitioning. Next, we propose in

Section 4.2 an accelerated Gauss-Seidel algorithm (Algorithm 2) in the sketching framework described in Section 2 and state its convergence guarantee in Theorem 4.3. In Section 4.3, we describe algorithmic lower bounds for both Gauss-Seidel and its accelerated variant. We conclude in Section 4.4 by specializing Theorem 4.3 to the case of random coordinate sampling. All proofs are deferred to the appendix.

4.1 Separation between fixed partition and random coordinate sampling

We first present our results on the separation between Gauss-Seidel on fixed partition versus random coordinate sampling. We do this by constructing a family of instances for which a separation exists. The particular family of $n \times n$ positive definite matrices we focus on is $\mathscr{A} = \{A_{\alpha,\beta} : \alpha > 0, \alpha + \beta > 0\}$ with $A_{\alpha,\beta}$ defined as

$$A_{\alpha,\beta} = \alpha I + \frac{\beta}{n} \mathbf{1}_n \mathbf{1}_n^\mathsf{T} . \tag{4.1}$$

The family $\mathscr{A}$ exhibits a crucial property that $\Pi^\mathsf{T} A_{\alpha,\beta} \Pi = A_{\alpha,\beta}$ for every $n \times n$ permutation matrix Π. By exploiting this fact, [7] recently used matrices of the form (4.1) to illustrate the behavior of cyclic versus randomized permutations in coordinate descent. In this section, we will describe similar kinds of calculations.

We are mainly interested in the behavior of Gauss-Seidel as the matrices $A_{\alpha,\beta}$ become ill-conditioned. To do this, we explore a particular parameterization which holds the minimum eigenvalue equal to one and sends the maximum eigenvalue to infinity. We make this explicit via the sub-family $\{A_{1,\beta}\}_{\beta>0}$. Our first proposition characterizes the behavior of Gauss-Seidel with fixed partitions on instances of $A_{1,\beta}$ matrices.

Proposition 4.1. *Fix any $\beta > 0$ and integers n, p such that $1 \le p \le n$ and $n/p = k$ for some integer $k \ge 1$. Let $Q_1, ...Q_k$ be any partition of $\{1, ..., n\}$ with $|Q_i| = p$ for all $1 \le i \le k$. Define the matrices $S_i \in \mathbb{R}^{n \times p}$ to be the column selector matrix corresponding to partition Q_i. Suppose the random matrix $S \in \mathbb{R}^{n \times p}$ takes on value S_i with probability $1/k$. For every $A_{1,\beta} \in \mathscr{A}$ we have that*

$$\mu_{\text{part}} = \lambda_{\min}(\mathbb{E}[P_{A_{1,\beta}^{1/2} S}]) = \frac{p}{n + \beta p} . \tag{4.2}$$

Next, we perform a similar calculation under the random column sampling model.

Proposition 4.2. *Fix any $\beta > 0$ and integers n, p such that $1 < p < n$. Suppose the random matrix $S \in \mathbb{R}^{n \times p}$ is defined such that each column of S is chosen uniformly at random from $\{e_1, ..., e_n\}$ without replacement. For every $A_{1,\beta} \in \mathscr{A}$ we have that*

$$\mu_{\text{rand}} = \lambda_{\min}(\mathbb{E}[P_{A_{1,\beta}^{1/2} S}]) = \frac{p}{n + \beta p} + \frac{(p-1)\beta p}{(n-1)(n + \beta p)} . \tag{4.3}$$

The differences between (4.2) and (4.3) are striking. Let us assume that $\beta \gg n$. Then, we have that $\mu_{\text{part}} \approx 1/\beta$ for (4.2), whereas $\mu_{\text{rand}} \approx \frac{p-1}{n-1}$ for (4.3). That is, $\mu_{\text{part}} \ll \mu_{\text{rand}}$, and we can make Gauss-Seidel *arbitrarily* slower on the same input by using any fixed partition instead of randomly chosen coordinates. More specifically, by applying (2.2) and (2.9), we see that Gauss-Seidel computes a solution x_k satisfying $\mathbb{E}[\|x_k - x_*\|_{A_{1,\beta}}] \le \varepsilon$ in $k = O(\frac{n}{p} \log(1/\varepsilon))$ iterations using random coordinates versus $k = O(\beta \log(1/\varepsilon))$ iterations using any fixed partition, and $k = O(\sqrt{\frac{n}{p}\beta} \log(1/\varepsilon))$ iterations using any fixed partition with acceleration (Algorithm 1). In Section 4.3, we show that these upper bounds are essentially tight, and we also verify the findings numerically in Section 5.1.

4.2 A convergence result for accelerated Gauss-Seidel

Motivated by our findings in the last section, we investigate the behavior of accelerated Gauss-Seidel beyond fixed partition sampling. Algorithm 2 describes an accelerated Gauss-Seidel algorithm which generalizes Algorithm 1 to the sketching framework of [19, 5] discussed in Section 2. We note that the difference in the structure of the updates between Algorithm 1 and Algorithm 2 is superficial. In the appendix, we show that a

minor tweak to Algorithm 1 is sufficient to make it identical to Algorithm 2 invoked with sketching matrices corresponding to a fixed partition, and hence can be viewed as a special case. Our main convergence result concerning Algorithm 2 is presented in Theorem 4.3.

Algorithm 2 Accelerated randomized block Gauss-Seidel.

Require: $A \in \mathbb{R}^{n\times n}$, $A \succ 0$, $b \in \mathbb{R}^n$, sketching matrices $\{S_k\}_{k=0}^{T-1} \subseteq \mathbb{R}^{n\times p}$, $x_0 \in \mathbb{R}^n$, $\mu \in (0,1)$, $\nu \geq 1$.

Set $\tau = \sqrt{\mu/\nu}$.
Set $y_0 = z_0 = x_0$.
for $k = 0, ..., T-1$ **do**
    $x_{k+1} = \frac{1}{1+\tau} y_k + \frac{\tau}{1+\tau} z_k$.
    $y_{k+1} = x_{k+1} - S_k (S_k^\mathsf{T} A S_k)^\dagger S_k^\mathsf{T} (A x_{k+1} - b)$.
    $z_{k+1} = z_k + \tau(x_{k+1} - z_k) - \frac{\tau}{\mu} S_k (S_k^\mathsf{T} A S_k)^\dagger S_k^\mathsf{T} (A x_{k+1} - b)$.
end for
return y_T.

Theorem 4.3. *Let A be an $n \times n$ positive definite matrix and $b \in \mathbb{R}^n$. Let $x_* \in \mathbb{R}^n$ denote the unique vector satisfying $Ax_* = b$. Suppose each S_k, $k = 0, 1, 2, ...$ is an independent copy of a random matrix $S \in \mathbb{R}^{n\times p}$. Let $\mu = \lambda_{\min}(\mathbb{E}[P_{A^{1/2}S}])$. Suppose the distribution of S satisfies $\mu > 0$. Invoke Algorithm 2 with μ and ν, where ν is defined as*

$$\nu = \lambda_{\max}\left(\mathbb{E}\left[(G^{-1/2} H G^{-1/2})^2\right]\right), \quad G = \mathbb{E}[H], \quad H = S(S^\mathsf{T} A S)^\dagger S^\mathsf{T}. \tag{4.4}$$

Then we have that for all $k \geq 0$,

$$\mathbb{E}[\|y_k - x_*\|_A] \leq \sqrt{2}\left(1 - \sqrt{\frac{\mu}{\nu}}\right)^{k/2} \|x_0 - x_*\|_A. \tag{4.5}$$

We note that in the setting of Theorem 4.3, by the definitions it is always the case that $\nu \leq 1/\mu$, and hence the iteration complexity of acceleration is at least as good as the iteration complexity without acceleration.

The proof of Theorem 4.3 follows the framework of Lyapunov analysis described in Wilson et al. [29], and is itself a special case of a general proof framework for randomized accelerated methods which we develop in the appendix. At a high level, the proof works by showing that the following function V_k defined as

$$V_k = f(y_k) - f(x_*) + \frac{\mu}{2}\|z_k - x_*\|^2_{G^{-1}},$$

satisfies, for all $k \geq 0$, the inequality

$$\mathbb{E}[V_{k+1}] \leq \left(1 - \sqrt{\frac{\mu}{\nu}}\right)\mathbb{E}[V_k],$$

where $f(x) = \frac{1}{2}x^\mathsf{T} A x - x^\mathsf{T} b$. Unrolling this recursion and noting that $f(y_k) - f(x_*) = \frac{1}{2}\|y_k - x_*\|^2_A$ yields the claimed result.

4.3 Algorithmic lower bounds for Gauss-Seidel

In this section, we discuss algorithmic lower bounds for both Gauss-Seidel and its accelerated variant. Our lower bounds are of the following form: we fix the algorithm at hand, and show that for every instance presented to the algorithm, there exists an adversarial starting point such that the upper bounds on iteration complexity are order-wise sharp. These kinds of lower bounds are weaker than the standard types of oracle lower bounds common in optimization, but they are strong enough to allow us to compare the iteration complexity of non-accelerated and accelerated Gauss-Seidel meaningfully.

We first describe an algorithmic lower bound for the Gauss-Seidel iteration (2.2).

Proposition 4.4. *Let A be an $n \times n$ symmetric positive definite matrix, and let S be a random matrix satisfying $\mu = \lambda_{\min}(\mathbb{E}[P_{A^{1/2}S}]) > 0$. Let x_* denote the solution to $Ax = b$. There exists a starting point $x_0 \in \mathbb{R}^n$ such that (2.2) satisfies for all $k \geq 0$,*

$$\mathbb{E}[\|x_k - x_*\|_A] \geq (1-\mu)^k \|x_0 - x_*\|_A \,. \tag{4.6}$$

Next, we describe a lower bound on the sequence of updates defined by Algorithm 2.

Proposition 4.5. *Under the setting of Theorem 4.3, there exists starting positions $y_0, z_0 \in \mathbb{R}^n$ such that the iterates $\{(y_k, z_k)\}_{k\geq 0}$ produced by Algorithm 2 satisfy*

$$\mathbb{E}[\|y_k - x_*\|_A + \|z_k - x_*\|_A] \geq \left(1 - \sqrt{\frac{\mu}{\nu}}\right)^k \|y_0 - x_*\|_A \,. \tag{4.7}$$

As mentioned previously, since Algorithm 1 can be viewed as a special case of Algorithm 2, Proposition 4.5 applies as well to Algorithm 1.

4.4 Specializing Gauss-Seidel to random coordinate sampling

We now instantiate Theorem 4.3 to random coordinate sampling. The μ quantity which appears in Theorem 4.3 is identical to the quantity appearing in the rate of non-accelerated Gauss-Seidel (2.3). Hence, the convergence rate is $O(\sqrt{\nu\mu_{\text{rand}}^{-1}}\log(1/\varepsilon))$ and the only new term here is ν. In order to provide a more intuitive interpretation of the ν quantity, we present an upper bound on ν in terms of an effective block condition number which we define below. We also demonstrate a lower bound on ν which characterizes the smallest ν possible. Given an index set $J \subseteq 2^{[n]}$, we define the effective block condition number of a matrix A as

$$\kappa_{\text{eff},J}(A) = \frac{\max_{i\in J} A_{ii}}{\lambda_{\min}(A_J)} \,. \tag{4.8}$$

Note that $\kappa_{\text{eff},J}(A) \leq \kappa(A_J)$ always. With this notation in place, we state the following lemma.

Lemma 4.6. *Let A be an $n \times n$ positive definite matrix and let p satisfy $1 < p < n$. We have that*

$$\frac{n}{p} \leq \nu \leq \frac{n}{p}\left(\frac{p-1}{n-1} + \left(1 - \frac{p-1}{n-1}\right)\kappa_{\text{eff},p}(A)\right) \,, \quad \kappa_{\text{eff},p}(A) = \max_{J\subseteq 2^{[n]}:|J|=p} \kappa_{\text{eff},J}(A) \,.$$

where ν is defined in (4.4) and the distribution of S corresponds to randomly selecting p coordinates without replacement.

Lemma 4.6 states that if the $p \times p$ sub-blocks of A are well-conditioned as defined by the effective block condition number $\kappa_{\text{eff},J}(A)$, then the speed-up of accelerated Gauss-Seidel with random coordinate selection over its non-accelerate counterpart parallels the case of fixed partitioning sampling (i.e. the rate described in (2.9) versus the rate in (2.3)). This is a reasonable condition, since very ill-conditioned sub-blocks will lead to numerical instabilities in solving the sub-problems when implementing Gauss-Seidel. On the other hand, we note that Lemma 4.6 provides merely a sufficient condition for speed-ups from acceleration. In Section 5, we show that on many instances we encountered, the ν parameter behaves much closer to the lower bound n/p than Lemma 4.6 suggests. We also compare numerically the bound from Lemma 4.6 to the actual ν value on various instances in Section 5.4. We leave a more thorough theoretical analysis of this parameter to future work.

With Lemma 4.6 in place, we can combine Theorem 4.3 with (2.6) to derive the following bound on the iteration complexity of accelerated Gauss-Seidel with random coordinates as (recalling the definition of $\kappa_{\text{eff}}(A)$ from (2.8))

$$T_{\text{rand,acc}} \leq O\left(\frac{n}{p}\sqrt{\kappa_{\text{eff}}(A)\kappa_{\text{eff},p}(A)}\log(1/\varepsilon)\right) \,.$$

We conclude this section by constructing an instance for which this effective block condition number $\kappa_{\text{eff},p}$ is well behaved. Consider the sub-family $\{A_\delta\}_{\delta>0} \subseteq \mathscr{A}$, with

$$A_\delta = A_{n+\delta,-n}\,, \quad \delta > 0\,. \tag{4.9}$$

We have that

$$(A_\delta)_{ii} = n - 1 + \delta\,, \;\; \lambda_{\min}(A_J) = n - p + \delta\,, \;\; \kappa_{\text{eff},p}(A_\delta) = \frac{n-1+\delta}{n-p+\delta}\,.$$

Plugging these calculations into the bound from Lemma 4.6, we conclude that for any $\delta > 0$,

$$\frac{n}{p} \le \nu(A_\delta) \le \frac{n}{p}\left(1 + \frac{p-1}{n-1}\right)\,. \tag{4.10}$$

Therefore this bound is sharp up to a constant factor of two. Let us finally combine our bound on ν with Theorem 4.3 and compare the iteration complexity of Gauss-Seidel with and without acceleration under the random coordinate sampling model. For ease of comparison, let us assume $p = o(n)$ and $\delta \in (0, 1)$. By a nearly identical calculation to Proposition 4.2, it is straightforward to show (see Proposition A.1 in the appendix) that

$$\mu_{\text{rand}} = \lambda_{\min}(\mathbb{E}[P_{A_\delta^{1/2}}]) = \frac{p\delta}{n(n-p+\delta)}\,.$$

With this calculation, Theorem 4.3 states that at most $O(\frac{n^{3/2}}{p\sqrt{\delta}}\log(1/\varepsilon))$ iterations are sufficient for an ε-accurate solution. On the other hand, when we do not use acceleration, (2.3) states that $O(\frac{n^2}{p\delta}\log(1/\varepsilon))$ iterations are sufficient and Proposition 4.4 shows there exists a starting position for which it is necessary. Hence, as either n grows large or δ tends to zero, the benefits of acceleration on this instance become more pronounced.

5 Experiments

In this section we experimentally validate our theoretical results on how our accelerated algorithms can improve convergence rates. Our experiments use a combination of synthetic matrices and matrices from large scale machine learning tasks.

Setup. We run all our experiments on a 4 socket Intel Xeon CPU E7-8870 machine with 18 cores per socket and 1TB of DRAM. We implement all our algorithms in Python using `numpy`, and use the Intel MKL library which ships with the Anaconda distribution for numerical operations. To study how the execution time relates to the runtime complexity proposed in Section 4, we configure MKL to only use a single thread for execution. We discuss opportunities for parallelizing Gauss-Seidel in Section 6. We run each randomized algorithm for 5 different random seeds, and plot the average value across trials, presenting the minimum and maximum values as shaded regions. We report errors as relative errors, i.e. $\|x_k - x_*\|_A^2/\|x_*\|_A^2$. Finally, we use the best values of μ and ν which we found when tuning each experiment.

5.1 Fixed partitioning versus random coordinate sampling

Our first set of experiments numerically verify the separation between fixed partitioning sampling versus random coordinate sampling.

5.1.1 Example from Section 4.1

Figure 1 shows the progress per iteration on solving $A_{1,\beta}x = b$, with the $A_{1,\beta}$ defined in Section 4.1. Here we set $n = 5000$, $p = 500$, $\beta = 1000$, and $b \sim N(0, I)$. Figure 1 verifies our analytical findings in Section 4.1, that

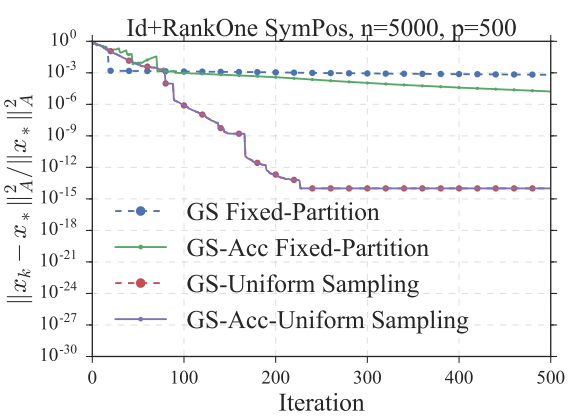

Figure 1: Experiments comparing fixed partitions versus random coordinate sampling for the example from Section 4.1 with $n = 5000$ coordinates, block size $p = 500$.

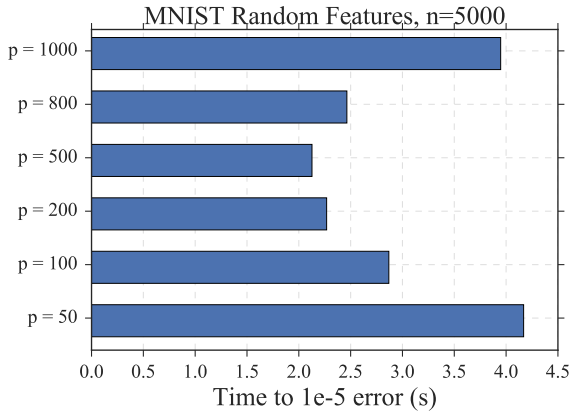

Figure 2: The effect of block size on the accelerated Gauss-Seidel method. For the MNIST dataset (pre-processed using random features [21]) we see that block size of $p = 500$ works best.

the fixed partition scheme is substantially worse than uniform sampling on this instance. It also shows that in this case, acceleration provides little benefit in the case of random coordinate sampling. This is because both μ and $1/\nu$ are order-wise p/n, and hence the rate for accelerated and non-accelerated coordinate descent coincide. However we note that this only applies for matrices where μ is as large as it can be (i.e. p/n), that is instances for which Gauss-Seidel is converging already at the optimal rate.

5.1.2 Kernel ridge regression

We next evaluate how fixed partitioning and random coordinate sampling affects the performance of Gauss-Seidel on large scale machine learning tasks. To do this we use two popular image classification datasets, MNIST[2] and CIFAR-10[3]. The task we evaluate is kernel ridge regression (KRR) with a Gaussian kernel. Specifically, given a labelled dataset $\{(x_i, y_i)\}_{i=1}^n$, we solve the linear system

$$(K + \lambda I)\alpha = Y \ , \quad K_{ij} = \exp(-\gamma \|x_i - x_j\|_2^2) \ , \quad Y = (y_1, ..., y_n)^\mathsf{T} \ , \tag{5.1}$$

where $\lambda, \gamma > 0$ are tunable parameters. (See e.g. [24] for background on KRR). The key property of kernel ridge regression is that the kernel matrix K is always positive semi-definite, and hence Algorithm 2 applies to solving (5.1).

For the MNIST dataset, we apply the Gaussian kernel on the raw pixels to generate the kernel matrix K with $n = 60000$ rows and columns. For CIFAR-10, we first apply standard pre-processing steps used in image classification [2, 25] and then apply the Gaussian kernel on top of our pre-processed images; the resulting kernel matrix has $n = 50000$ rows and columns.

Results from running 500 iterations of random coordinate sampling and fixed partitioning algorithms are shown in Figure 4. We plot the convergence rates both across time and across iterations. Comparing convergence across iterations we see that for both datasets random coordinate sampling is essential to achieve errors of 10^{-4} or lower.

In terms of running time, fixed partitioning is desirable in scenarios where the inputs are spread across machines or stored on disk drives, as random access is expensive in these cases. Further the partitions can be computed once and cached across iterations. However, we see that this speedup in implementation comes at

[2] http://yann.lecun.com/exdb/mnist/
[3] https://www.cs.toronto.edu/~kriz/cifar.html

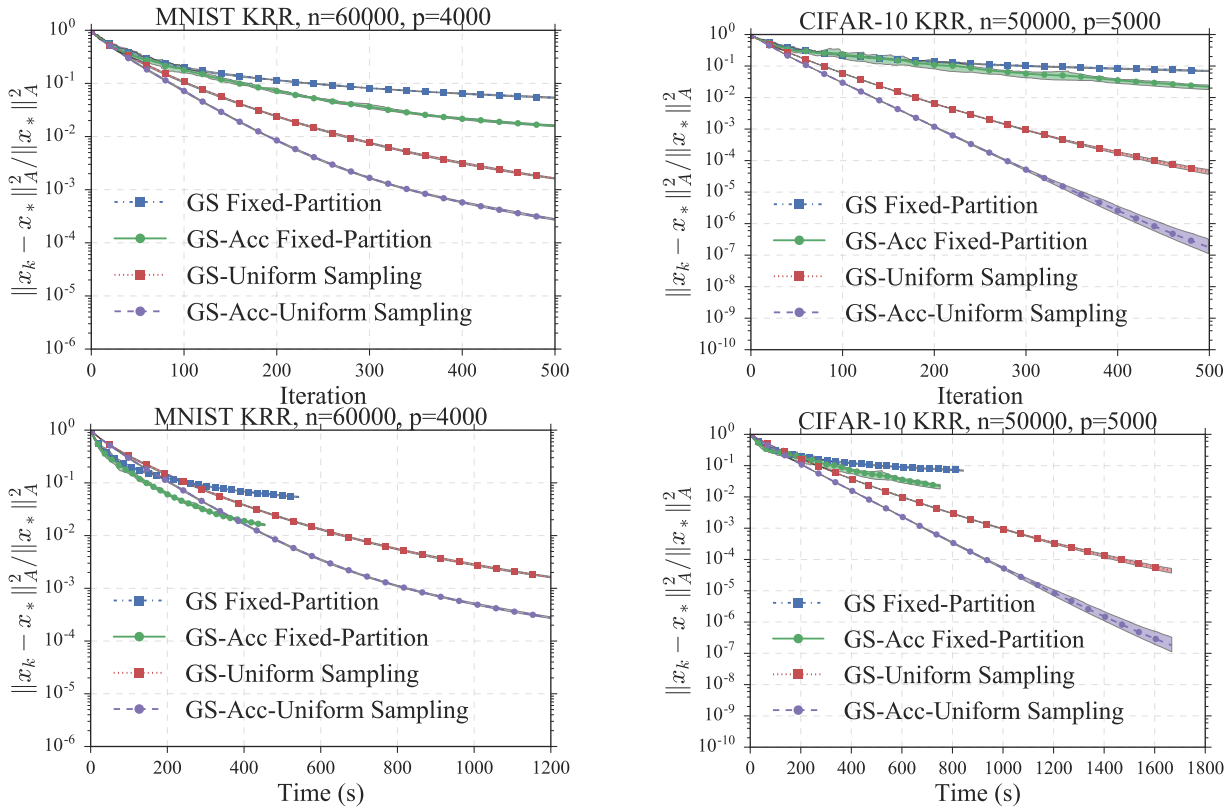

Figure 3: Experiments comparing fixed partitions versus uniform random sampling for MNIST and CIFAR-10 while running kernel ridge regression. MNIST has $n = 60000$ coordinates while CIFAR-10 has $n = 50000$ coordinates. We set block size to $p = 4000$ and $p = 5000$, respectively.

a substantial cost in terms of convergence rate. For example in the case of CIFAR-10, using fixed partitions leads to an error of 2.2×10^{-2} after around 750 seconds. In comparison we see that random coordinate sampling achieves a similar error in around 370 seconds and is thus $2\times$ faster. We also note that this speedup increases for lower error tolerances.

5.2 Comparing Gauss-Seidel to Conjugate Gradient

We next compare Gauss-Seidel with random coordinate sampling to the classical conjugate-gradient (CG) algorithm. CG is an important baseline to compare with, as it is the de-facto standard iterative algorithm for solving linear systems in the numerical analysis community. Our goal is to see if the wins from random coordinate sampling make Gauss-Seidel competitive with CG. For this experiment, we use the same MNIST and CIFAR-10 matrices described in Section 5.1.2.

The results of our experiment are shown in Figure 4. We note that Gauss-Seidel both with and without acceleration outperform CG. As an example, we note that to reach error 10^{-1} on CIFAR-10, CG takes roughly 2,500 seconds, compared to less than 500 seconds for accelerated Gauss-Seidel, which is a $5\times$ improvement. We also see that the accelerated Gauss-Seidel method is around $1.5\times$ faster than the non-accelerated Gauss-Seidel method to achieve an error of e.g. 10^{-4}. This speedup increases for lower error thresholds.

We now discuss the performance of CG relative to Gauss-Seidel. Recall that our matrices A are fully dense, and hence each iteration of CG takes $O(n^2)$ in general, since there is no sparsity to exploit. On the other hand, each iteration of both non-accelerated and accelerated Gauss-Seidel takes $O(np^2 + p^3)$. Hence, as long as $p = O(n^{2/3})$, the time per iteration of Gauss-Seidel is order-wise no worse than CG. On the

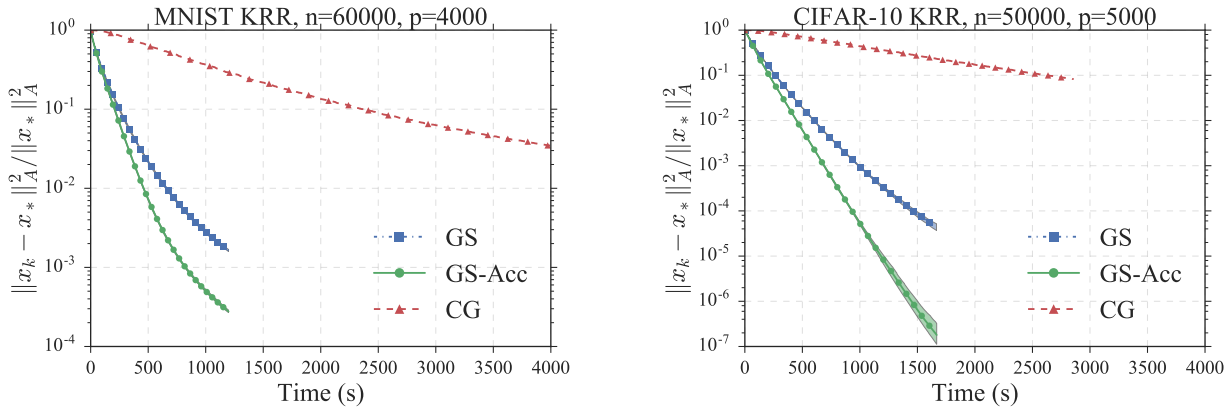

Figure 4: Experiments comparing accelerated Gauss-Seidel to non-accelerated Gauss-Seidel as well as conjugate gradient for kernel ridge regression run on MNIST and CIFAR-10 datasets. MNIST has $n = 60000$ coordinates while CIFAR-10 has $n = 50000$ coordinates. We set block size to $p = 4000$ and $p = 5000$, respectively.

other hand, let us consider the iteration complexity. For simplicity, let us compare to the non-accelerated Gauss-Seidel upper bound discussed in (2.8). Standard results state that CG takes at most $O(\sqrt{\kappa}\log(1/\varepsilon))$ iterations to reach an ε error solution, where κ denotes the condition number of A. On the other hand, (2.8) states that Gauss-Seidel takes at most $O(\frac{n}{p}\kappa_{\text{eff}}\log(1/\varepsilon))$, where $\kappa_{\text{eff}} = \frac{\max_{1\le i\le n} A_{ii}}{\lambda_{\min}(A)}$. In the case of any (normalized) kernel matrix associated with a translation-invariant kernel such as the Gaussian kernel, we have $\max_{1\le i\le n} A_{ii} = 1$, and hence generally speaking κ_{eff} is much lower than κ. As an example, for the CIFAR-10 kernel matrix, $\kappa \approx 3.9 \times 10^7$ and $\sqrt{\kappa} \approx 6200$, whereas $\kappa_{\text{eff}} \approx 1600$.

5.3 Effect of block size

We next analyze the importance of the block size p for the accelerated Gauss-Seidel method. As the values of μ and ν change for each setting of p, we use a smaller MNIST matrix for this experiment. We apply a random feature transformation [21] to generate an $n \times d$ matrix F with $d = 5000$ features. We then use the input $A = F^\mathsf{T}F$ and $b = F^\mathsf{T}Y$ for the accelerated Gauss-Seidel algorithm. Figure 2 shows the wall clock time to converge to tolerance 10^{-5} as we vary the block size from $p = 50$ to $p = 1000$.

Increasing the block-size improves the amount of progress that is made per iteration but the time taken per iteration increases as $O(p^3)$ (Line 5, Algorithm 2). However, using efficient BLAS-3 primitives usually affords a speedup from systems techniques like cache blocking. We see the effects of this in Figure 2 where using $p = 500$ performs better than using $p = 50$. We also see that these benefits reduce for much larger block sizes and thus $p = 1000$ is slower.

5.4 Computing the μ and ν constants

In our last experiment, we explicitly compute the μ and ν constants from Theorem 4.3 for a few 16×16 positive definite matrices constructed as follows.

Linspaced eigenvalues. We first draw Q uniformly at random from $n \times n$ orthogonal matrices. We then construct $A_i = Q\Sigma_i Q^\mathsf{T}$ for $i = 1, 2, 3$, where Σ_1 is `diag(linspace(1, 10, 16))`, Σ_2 is `diag(linspace(1, 100, 16))`, and Σ_3 is `diag(linspace(1, 1000, 16))`.
Random Wishart. We first draw B_i with iid $N(0,1)$ entries, where $B_i \in \mathbb{R}^{m_i \times 16}$ with $m_1 = 18$, $m_2 = 20$, and $m_3 = 22$. We then set $A_i = B_i^\mathsf{T} B_i$.
Sobolev kernel. We form the matrix $A_{ij} = \min(i, j)$ with $1 \le i, j \le n$. This corresponds to the gram matrix for the set of points $x_1, ..., x_n \in \mathbb{R}$ with $x_i = i$ under the Sobolev kernel $\min(x, y)$.

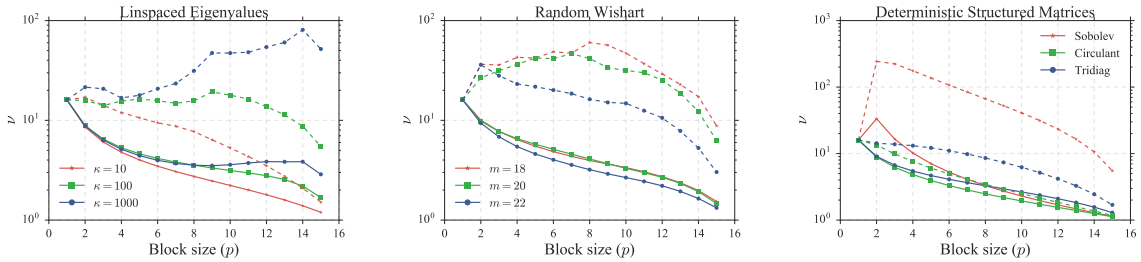

Figure 5: Comparison of the computed ν constant (solid lines) and ν bound from Theorem 4.3 (dotted lines) on random matrices with linspaced eigenvalues and random Wishart matrices.

Circulant matrix. We let A be a 16×16 instance of the family of circulant matrices $A_n = F_n \operatorname{diag}(c_n) F_n^*$ where F_n is the $n \times n$ unitary DFT matrix and $c_n = (1, 1/2, ..., 1/(n/2+1), ..., 1/2, 1)$. By construction this yields a real valued circulant matrix which is positive definite.
Tridiagonal matrix. We let A be a tridiagonal matrix with the diagonal value equal to one, and the off diagonal value equal to $(\delta - a)/(2\cos(\pi n/(n+1)))$ for $\delta = 1/10$. The matrix has a minimum eigenvalue of δ.

Figure 5 shows the results of our computation for the linspace eigenvalue ensemble, the random Wishart ensemble and the other deterministic structured matrices. Alongside with the actual ν values, we plot the bound given for each instance by Lemma 4.6. From the figures we see that our bound is quite close to the computed value of ν for circulant matrices and for random matrices with linspaced eigenvalues with small κ. We plan to extend our analysis to derive a tighter bound in the future.

6 Conclusion

In this paper, we extended accelerated Gauss-Seidel beyond fixed partition sampling. We introduced a new data-dependent parameter ν which governs the speed-up of acceleration under more general sampling schemes. Specializing our theory to random coordinate sampling, we derived an upper bound on ν which shows that well conditioned blocks are a sufficient condition to ensure speedup. Experimentally, we showed that in practice Gauss-Seidel with random coordinate sampling is readily accelerated beyond instances which our bound suggests.

The most obvious question remains to derive a sharper bound on the ν constant from Theorem 4.3. We believe that the experiments from Section 5 suggest that a sharper bound is possible, even if the matrix sub-blocks are ill-conditioned. Another interesting question is whether or not the iteration complexity of random coordinate sampling is always bounded above by the iteration complexity with fixed coordinate sampling. This question applies to both the non-accelerated and accelerated setting.

Moving beyond sharper analysis, another potential direction is to extend our analysis to more general non-linear problems which arise commonly in practice. We believe the theory of self-concordant functions provides a natural setting to extend our results.

Another interesting direction of future research is a practical implementation of accelerated Gauss-Seidel in a parallel or distributed setting, in the style of Tu et al. [28]. The main challenge in implementing an efficient parallel algorithm is determining how to sample coordinates. While this paper suggests that random coordinate sampling is preferable, random coordinate sampling cannot be easily parallelized in a multi-core setting and incurs significant communication overheads. We plan to explore if other sampling schemes such as shuffling the coordinates at the end of every epoch [22] (or few epochs) can be used to balance time spent in communication versus computation.

In a distributed setting, a successful implementation would also have to resolve the question of figuring out how to only update a block of variables at a time. Fercoq and Richtárik [3] propose one method of doing this, and we leave it to future work to incorporate their technique into accelerated Gauss-Seidel.

Acknowledgements

We thank Ross Boczar for assisting us with Mathematica support for non-commutative algebras, and Orianna DeMasi for providing useful feedback on earlier drafts of this manuscript. ACW is supported by an NSF Graduate Research Fellowship. BR is generously supported by ONR awards N00014-11-1-0723 and N00014-13-1-0129, NSF award CCF-1359814, the DARPA Fundamental Limits of Learning (Fun LoL) Program, a Sloan Research Fellowship, and a Google Research Award. This research is supported in part by DHS Award HSHQDC-16-3-00083, NSF CISE Expeditions Award CCF-1139158, DOE Award SN10040 DE-SC0012463, and DARPA XData Award FA8750-12-2-0331, and gifts from Amazon Web Services, Google, IBM, SAP, The Thomas and Stacey Siebel Foundation, Apple Inc., Arimo, Blue Goji, Bosch, Cisco, Cray, Cloudera, Ericsson, Facebook, Fujitsu, HP, Huawei, Intel, Microsoft, Mitre, Pivotal, Samsung, Schlumberger, Splunk, State Farm and VMware.

References

[1] Z. Allen-Zhu, P. Richtárik, Z. Qu, and Y. Yuan. Even Faster Accelerated Coordinate Descent Using Non-Uniform Sampling. In *ICML*, 2016.

[2] A. Coates and A. Y. Ng. Learning Feature Representations with K-Means. In *Neural Networks: Tricks of the Trade*. Springer, 2012.

[3] O. Fercoq and P. Richtárik. Accelerated, parallel, and proximal coordinate descent. *SIAM J. Optim.*, 25(4), 2015.

[4] K. Fountoulakis and R. Tappenden. A Flexible Coordinate Descent Method. *arXiv*, 1507.03713, 2016.

[5] R. M. Gower and P. Richtárik. Randomized Iterative Methods for Linear Systems. *SIAM Journal on Matrix Analysis and Applications*, 36, 2015.

[6] C. Kelley. *Iterative Methods for Linear and Nonlinear Equations*. Society for Industrial and Applied Mathematics, 1995.

[7] C.-P. Lee and S. J. Wright. Random Permutations Fix a Worst Case for Cyclic Coordinate Descent. *arXiv*, 1607.08320, 2016.

[8] Y. T. Lee and A. Sidford. Efficient Accelerated Coordinate Descent Methods and Faster Algorithms for Solving Linear Systems. In *FOCS*, 2013.

[9] D. Leventhal and A. S. Lewis. Randomized methods for linear constraints: Convergence rates and conditioning. *Mathematics of Operations Research*, 35(3), 2010.

[10] Q. Lin, Z. Lu, and L. Xiao. An Accelerated Proximal Coordinate Gradient Method. In *NIPS*, 2014.

[11] J. Liu and S. J. Wright. An accelerated randomized Kaczmarz algorithm. *Mathematics of Computation*, 85(297), 2016.

[12] Z. Lu and L. Xiao. On the complexity analysis of randomized block-coordinate descent methods. *Mathematical Programming*, 152(1–2), 2015.

[13] D. Needell and J. A. Tropp. Paved with good intentions: Analysis of a randomized block kaczmarz method. *Linear Algebra and its Applications*, 441, 2014.

[14] Y. Nesterov. *Introductory Lectures on Convex Programming. Basic Course*, volume 87 of *Applied Optimization*. Springer, 2004.

[15] Y. Nesterov. Efficiency of Coordinate Descent Methods on Huge-Scale Optimization Problems. *SIAM J. Optim.*, 22(2), 2012.

[16] Y. Nesterov and S. Stich. Efficiency of accelerated coordinate descent method on structured optimization problems. Technical report, Université catholique de Louvain, CORE Discussion Papers, 2016.

[17] Z. Qu and P. Richtárik. Coordinate descent with arbitrary sampling I: Algorithms and complexity. *arXiv*, 1412.8060, 2014.

[18] Z. Qu and P. Richtárik. Coordinate descent with arbitrary sampling II: Expected separable overapproximation. *arXiv*, 1412.8063, 2014.

[19] Z. Qu, P. Richtárik, et al. Randomized dual coordinate ascent with arbitrary sampling. In *NIPS*, 2015.

[20] Z. Qu, P. Richtárik, et al. SDNA: Stochastic Dual Newton Ascent for Empirical Risk Minimization. In *ICML*, 2016.

[21] A. Rahimi and B. Recht. Random features for large-scale kernel machines. In *NIPS*, 2007.

[22] B. Recht and C. Ré. Parallel stochastic gradient algorithms for large-scale matrix completion. *Mathematical Programming Computation*, 5(2):201–226, 2013.

[23] P. Richtárik and M. Takáč. Iteration complexity of randomized block-coordinate descent methods for minimizing a composite function. *Mathematical Programming*, 114, 2014.

[24] B. Schölkopf and A. J. Smola. *Learning with kernels*. MIT Press, 2001.

[25] E. R. Sparks, S. Venkataraman, T. Kaftan, M. Franklin, and B. Recht. KeystoneML: Optimizing Pipelines for Large-Scale Advanced Analytics. *arXiv*, 1610.09451, 2016.

[26] T. Strohmer and R. Vershynin. A randomized Kaczmarz algorithm with exponential convergence. *Journal of Fourier Analysis and Applications*, 15(1), 2009.

[27] P. Tseng and S. Yun. A coordinate gradient descent method for nonsmooth separable minimization. *Mathematical Programming*, 117(1), 2009.

[28] S. Tu, R. Roelofs, S. Venkataraman, and B. Recht. Large Scale Kernel Learning using Block Coordinate Descent. *arXiv*, 1602.05310, 2016.

[29] A. C. Wilson, B. Recht, and M. I. Jordan. A Lyapunov Analysis of Momentum Methods in Optimization. *arXiv*, 1611.02635, 2016.

[30] S. J. Wright. Coordinate descent algorithms. *Mathematical Programming*, 151(1), 2015.

[31] F. Zhang. *The Schur Complement and its Applications*, volume 4 of *Numerical Methods and Algorithms*. Springer, 2005.

A Proofs for separation results (Section 4.1)

Recall the family of $n \times n$ positive definite matrices $\mathscr{A}$ defined in (4.9) as

$$A_{\alpha,\beta} = \alpha I + \frac{\beta}{n}\mathbf{1}_n\mathbf{1}_n^\mathsf{T}\,, \quad \alpha > 0, \alpha + \beta > 0\,. \tag{A.1}$$

We first gather some elementary formulas. By the matrix inversion lemma,

$$A_{\alpha,\beta}^{-1} = \left(\alpha I + \frac{\beta}{n}\mathbf{1}_n\mathbf{1}_n^\mathsf{T}\right)^{-1} = \alpha^{-1}I - \frac{\beta/n}{\alpha(\alpha+\beta)}\mathbf{1}_n\mathbf{1}_n^\mathsf{T}\,. \tag{A.2}$$

Furthermore, let $S \in \mathbb{R}^{n\times p}$ be any column selector matrix with no duplicate columns. We have again by the matrix inversion lemma

$$(S^\mathsf{T}A_{\alpha,\beta}S)^{-1} = \left(\alpha I + \frac{\beta}{n}\mathbf{1}_p\mathbf{1}_p^\mathsf{T}\right)^{-1} = \alpha^{-1}I - \frac{\beta/n}{\alpha(\alpha+\beta p/n)}\mathbf{1}_p\mathbf{1}_p^\mathsf{T}\,. \tag{A.3}$$

The fact that the right hand side is independent of S is the key property which makes our calculations possible. Indeed, we have that

$$S(S^\mathsf{T}A_{\alpha,\beta}S)^{-1}S^\mathsf{T} = \alpha^{-1}SS^\mathsf{T} - \frac{\beta/n}{\alpha(\alpha+\beta p/n)}S\mathbf{1}_p\mathbf{1}_p^\mathsf{T}S^\mathsf{T}\,. \tag{A.4}$$

With these formulas in hand, our next proposition gathers calculations for the case when S represents uniformly choosing p columns without replacement.

Proposition A.1. *Consider the family of $n \times n$ positive definite matrices $\{A_{\alpha,\beta}\}$ from (A.1). Fix any integer p such that $1 < p < n$. Let $S \in \mathbb{R}^{n\times p}$ denote a random column selector matrix where each column of S is chosen uniformly at random without replacement from $\{e_1, ..., e_n\}$. For any $A_{\alpha,\beta}$,*

$$\mathbb{E}[S(S^\mathsf{T} A_{\alpha,\beta} S)^{-1} S^\mathsf{T} A_{\alpha,\beta}] = p\frac{(n-1)\alpha + (p-1)\beta}{(n-1)(n\alpha + p\beta)} I + \frac{(n-p)p\beta}{n(n-1)(n\alpha+p\beta)} \mathbf{1}_n \mathbf{1}_n^\mathsf{T}, \tag{A.5}$$

$$\begin{aligned}\mathbb{E}[S(S^\mathsf{T} A_{\alpha,\beta} S)^{-1} S^\mathsf{T} G_{\alpha,\beta}^{-1} S (S^\mathsf{T} A_{\alpha,\beta} S)^{-1} S^\mathsf{T}] &= \left(\frac{1}{\alpha} - \frac{(n-p)^2\beta}{(n-1)((n-1)\alpha + (p-1)\beta)(n\alpha+p\beta)}\right) I \\ &\quad + \frac{(p-1)\beta(n\alpha(1-2n) + np(\alpha-\beta) + p\beta)}{(n-1)n\alpha((n-1)\alpha + (p-1)\beta)(n\alpha+p\beta)} \mathbf{1}_n \mathbf{1}_n^\mathsf{T}. \end{aligned} \tag{A.6}$$

Above, $G_{\alpha,\beta} = \mathbb{E}[S(S^\mathsf{T} A_{\alpha,\beta} S)^{-1} S^\mathsf{T}]$.

Proof. First, we have the following elementary expectation calculations,

$$\mathbb{E}[SS^\mathsf{T}] = \frac{p}{n} I, \tag{A.7}$$

$$\mathbb{E}[S\mathbf{1}_p\mathbf{1}_p^\mathsf{T} S^\mathsf{T}] = \frac{p}{n}\left(1 - \frac{p-1}{n-1}\right) I + \frac{p}{n}\left(\frac{p-1}{n-1}\right) \mathbf{1}_n \mathbf{1}_n^\mathsf{T}, \tag{A.8}$$

$$\mathbb{E}[SS^\mathsf{T}\mathbf{1}_n\mathbf{1}_p^\mathsf{T} S^\mathsf{T}] = \mathbb{E}[S\mathbf{1}_p\mathbf{1}_n^\mathsf{T} SS^\mathsf{T}] = \mathbb{E}[SS^\mathsf{T}\mathbf{1}_n\mathbf{1}_n^\mathsf{T} SS^\mathsf{T}] = \mathbb{E}[S\mathbf{1}_p\mathbf{1}_p^\mathsf{T} S^\mathsf{T}], \tag{A.9}$$

$$\mathbb{E}[S\mathbf{1}_p\mathbf{1}_p^\mathsf{T} S^\mathsf{T}\mathbf{1}_n\mathbf{1}_n^\mathsf{T} S\mathbf{1}_p\mathbf{1}_p^\mathsf{T} S^\mathsf{T}] = \frac{p^3}{n}\left(1 - \frac{p-1}{n-1}\right) I + \frac{p^3}{n}\left(\frac{p-1}{n-1}\right) \mathbf{1}_n\mathbf{1}_n^\mathsf{T}. \tag{A.10}$$

To compute $G_{\alpha,\beta}$, we simply plug (A.7) and (A.8) into (A.4). After simplification,

$$G_{\alpha,\beta} = \mathbb{E}[S(S^\mathsf{T} A_{\alpha,\beta} S)^{-1} S^\mathsf{T}] = \frac{p}{\alpha n}\left(1 - \frac{\beta/n}{\alpha + \beta p/n}\left(1 - \frac{p-1}{n-1}\right)\right) I - \frac{p}{n}\frac{p-1}{n-1}\frac{\beta/n}{\alpha(\alpha+\beta p/n)} \mathbf{1}_n\mathbf{1}_n^\mathsf{T}.$$

From this formula for $G_{\alpha,\beta}$, (A.5) follows immediately.

Our next goal is to compute $\mathbb{E}[S(S^\mathsf{T} A_{\alpha,\beta} S)^{-1} S^\mathsf{T} G_{\alpha,\beta}^{-1} S (S^\mathsf{T} A_{\alpha,\beta} S)^{-1} S^\mathsf{T}]$. To do this, we first invert $G_{\alpha,\beta}$. Applying the matrix inversion lemma, we can write down a formula for the inverse of $G_{\alpha,\beta}$,

$$G_{\alpha,\beta}^{-1} = \underbrace{\frac{(n-1)\alpha(n\alpha+p\beta)}{(n-1)p\alpha + (p-1)p\beta}}_{\gamma} I + \underbrace{\frac{(p-1)\beta(n\alpha+p\beta)}{np((n-1)\alpha + (p-1)\beta)}}_{\eta} \mathbf{1}_n\mathbf{1}_n^\mathsf{T}. \tag{A.11}$$

Next, we note for any r, q, using the properties that $S^\mathsf{T} S = I$, $\mathbf{1}_n^\mathsf{T} S \mathbf{1}_p = p$, and $\mathbf{1}_p^\mathsf{T}\mathbf{1}_p = p$, we have that

$$\begin{aligned}&(rSS^\mathsf{T} + qS\mathbf{1}_p\mathbf{1}_p^\mathsf{T} S^\mathsf{T})(\gamma I + \eta \mathbf{1}_n\mathbf{1}_n^\mathsf{T})(rSS^\mathsf{T} + qS\mathbf{1}_p\mathbf{1}_p^\mathsf{T} S^\mathsf{T}) \\ &\quad = \gamma r^2 SS^\mathsf{T} + 2r\gamma q S\mathbf{1}_p\mathbf{1}_p^\mathsf{T} S^\mathsf{T} + \eta r^2 SS^\mathsf{T}\mathbf{1}_n\mathbf{1}_n^\mathsf{T} SS^\mathsf{T} \\ &\qquad + pr\eta q(SS^\mathsf{T}\mathbf{1}_n\mathbf{1}_p^\mathsf{T} S^\mathsf{T} + S\mathbf{1}_p\mathbf{1}_n^\mathsf{T} SS^\mathsf{T}) + pq^2\gamma S\mathbf{1}_p\mathbf{1}_p^\mathsf{T} S^\mathsf{T} \\ &\qquad + \eta q^2 S\mathbf{1}_p\mathbf{1}_p^\mathsf{T} S^\mathsf{T}\mathbf{1}_n\mathbf{1}_n^\mathsf{T} S\mathbf{1}_p\mathbf{1}_p^\mathsf{T} S^\mathsf{T}.\end{aligned}$$

Taking expectations of both sides of the above equation and using the formulas in (A.7), (A.8), (A.9), and (A.10),

$$\begin{aligned}&\mathbb{E}[(rSS^\mathsf{T} + qS\mathbf{1}_p\mathbf{1}_p^\mathsf{T} S^\mathsf{T})(\gamma I + \eta\mathbf{1}_n\mathbf{1}_n^\mathsf{T})(rSS^\mathsf{T} + qS\mathbf{1}_p\mathbf{1}_p^\mathsf{T} S^\mathsf{T})] \\ &\quad = \frac{p(p(n-p)q^2 + 2(n-p)qr + (n-1)r^2)\gamma + p(n-p)(pq+r)^2\eta}{n(n-1)} I \\ &\qquad + \frac{p(p-1)(q(pq+2r)\gamma + (pq+r)^2\eta)}{n(n-1)} \mathbf{1}_n\mathbf{1}_n^\mathsf{T}.\end{aligned}$$

We now set $r = \alpha^{-1}$, $q = -\frac{\beta/n}{\alpha(\alpha+\beta p/n)}$, and γ, η from (A.11) to reach the desired formula for (A.6). □

Proposition 4.2 follows immediately from Proposition A.1 by plugging in $\alpha = 1$ into (A.5). We next consider how (A.4) behaves under a fixed partition of $\{1, ..., n\}$. Recall our assumption on partitions: $n = pk$ for some integer $k \geq 1$, and we sequentially partition $\{1, ..., n\}$ into k partitions of size p, i.e. $J_1 = \{1, ..., p\}$, $J_2 = \{p+1, ..., 2p\}$, and so on. Define $S_1, ..., S_k \in \mathbb{R}^{n \times p}$ such that S_i is the column selector matrix for the partition J_i, and S uniformly chooses S_i with probability $1/k$.

Proposition A.2. *Consider the family of $n \times n$ positive definite matrices $\{A_{\alpha,\beta}\}$ from (A.1), and let n, p, and S be described as in the preceding paragraph. We have that*

$$\mathbb{E}[S(S^\mathsf{T} A_{\alpha,\beta} S)^{-1} S^\mathsf{T} A_{\alpha,\beta}] = \frac{p}{n} I + \frac{p\beta}{n^2\alpha + np\beta} \mathbf{1}_n \mathbf{1}_n^\mathsf{T} - \frac{p\beta}{n^2\alpha + np\beta} \mathrm{blkdiag}(\underbrace{\mathbf{1}_p \mathbf{1}_p^\mathsf{T}, ..., \mathbf{1}_p \mathbf{1}_p^\mathsf{T}}_{k \text{ times}}) \,. \tag{A.12}$$

Proof. Once again, the expectation calculations are

$$\mathbb{E}[SS^\mathsf{T}] = \frac{p}{n} I, \;\; \mathbb{E}[S\mathbf{1}_p \mathbf{1}_p^\mathsf{T} S^\mathsf{T}] = \frac{p}{n} \mathrm{blkdiag}(\underbrace{\mathbf{1}_p \mathbf{1}_p^\mathsf{T}, ..., \mathbf{1}_p \mathbf{1}_p^\mathsf{T}}_{k \text{ times}}) \,.$$

Therefore,

$$\mathbb{E}[S(S^\mathsf{T} A_{\alpha,\beta} S)^{-1} S^\mathsf{T}] = \frac{p}{\alpha n} I - \frac{p}{n} \frac{\beta/n}{\alpha(\alpha + \beta p/n)} \mathrm{blkdiag}(\mathbf{1}_p \mathbf{1}_p^\mathsf{T}, ..., \mathbf{1}_p \mathbf{1}_p^\mathsf{T}) \,.$$

Furthermore,

$$\mathrm{blkdiag}(\mathbf{1}_p \mathbf{1}_p^\mathsf{T}, ..., \mathbf{1}_p \mathbf{1}_p^\mathsf{T}) \mathbf{1}_n \mathbf{1}_n^\mathsf{T} = \mathbf{1}_n \mathbf{1}_n^\mathsf{T} \mathrm{blkdiag}(\mathbf{1}_p \mathbf{1}_p^\mathsf{T}, ..., \mathbf{1}_p \mathbf{1}_p^\mathsf{T}) = p \mathbf{1}_n \mathbf{1}_n^\mathsf{T} \,,$$

Hence, the formula for $\mathbb{E}[S(S^\mathsf{T} A_{\alpha,\beta} S)^{-1} S^\mathsf{T} A_{\alpha,\beta}]$ follows. □

We now make the following observation. Let $Q_1, ..., Q_k$ be any partition of $\{1, ..., n\}$ into k partitions of size p. Let $\mathbb{E}_{S \sim Q_i}$ denote expectation with respect to S uniformly chosen as column selectors among $Q_1, ..., Q_k$, and let $\mathbb{E}_{S \sim J_i}$ denote expectation with respect to the S in the setting of Proposition A.2. It is not hard to see there exists a permutation matrix Π such that

$$\Pi^\mathsf{T} \mathbb{E}_{S \sim Q_i}[S(S^\mathsf{T} A_{\alpha,\beta} S)^{-1} S^\mathsf{T}] \Pi = \mathbb{E}_{S \sim J_i}[S(S^\mathsf{T} A_{\alpha,\beta} S)^{-1} S^\mathsf{T}] \,.$$

Using this permutation matrix Π,

$$\begin{aligned}
\lambda_{\min}(\mathbb{E}_{S \sim Q_i}[P_{A_{\alpha,\beta}^{1/2} S}]) &= \lambda_{\min}(\mathbb{E}_{S \sim Q_i}[S(S^\mathsf{T} A_{\alpha,\beta} S)^{-1} S^\mathsf{T}] A_{\alpha,\beta}) \\
&= \lambda_{\min}(\mathbb{E}_{S \sim Q_i}[S(S^\mathsf{T} A_{\alpha,\beta} S)^{-1} S^\mathsf{T}] \Pi A_{\alpha,\beta} \Pi^\mathsf{T}) \\
&= \lambda_{\min}(\Pi^\mathsf{T} \mathbb{E}_{S \sim Q_i}[S(S^\mathsf{T} A_{\alpha,\beta} S)^{-1} S^\mathsf{T}] \Pi A_{\alpha,\beta}) \\
&= \lambda_{\min}(\mathbb{E}_{S \sim J_i}[S(S^\mathsf{T} A_{\alpha,\beta} S)^{-1} S^\mathsf{T}] A_{\alpha,\beta}) \\
&= \lambda_{\min}(\mathbb{E}_{S \sim J_i}[P_{A_{\alpha,\beta}^{1/2} S}]) \,.
\end{aligned}$$

Above, the second equality holds because $A_{\alpha,\beta}$ is invariant under a similarity transform by any permutation matrix. Therefore, Proposition A.2 yields the μ_{part} value for every partition $Q_1, ..., Q_k$. The claim of Proposition 4.1 now follows by substituting $\alpha = 1$ into (A.12).

B Proofs for convergence results (Section 4.2)

We now state our main structural result for accelerated coordinate descent. Let $\mathbb{P}$ be a probability measure on $\Omega = \mathcal{S}^{n \times n} \times \mathbb{R}_+ \times \mathbb{R}_+$, with $\mathcal{S}^{n \times n}$ denoting $n \times n$ positive semi-definite matrices and $\mathbb{R}_+$ denoting positive

reals. Write $\omega \in \Omega$ as the tuple $\omega = (H, \Gamma, \gamma)$, and let $\mathbb{E}$ denote expectation with respect to $\mathbb{P}$. Suppose that $G = \mathbb{E}[\frac{1}{\gamma} H]$ exists and is positive definite.

Now suppose that $f : \mathbb{R}^n \longrightarrow \mathbb{R}$ is a differentiable and strongly convex function, and put $f_* = \min_x f(x)$, with x_* attaining the minium value. Suppose that f is both μ-strongly convex and has L-Lipschitz gradients with respect to the G^{-1} norm. This means that for all $x, y \in \mathbb{R}^n$, we have

$$f(y) \geq f(x) + \langle \nabla f(x), y - x \rangle + \frac{\mu}{2} \|y - x\|_{G^{-1}}^2 \,, \tag{B.1a}$$

$$f(y) \leq f(x) + \langle \nabla f(x), y - x \rangle + \frac{L}{2} \|y - x\|_{G^{-1}}^2 \,. \tag{B.1b}$$

We now define a random sequence as follows. Let $\omega_0 = (H_0, \Gamma_0, \gamma_0), \omega_1 = (H_1, \Gamma_1, \gamma_1), ...$ be independent realizations from $\mathbb{P}$. Starting from $y_0 = z_0 = x_0$ with x_0 fixed, consider the sequence $\{(x_k, y_k, z_k)\}_{k \geq 0}$ defined by the recurrence

$$\tau(x_{k+1} - z_k) = y_k - x_{k+1} \,, \tag{B.2a}$$

$$y_{k+1} = x_{k+1} - \frac{1}{\Gamma_k} H_k \nabla f(x_{k+1}) \,, \tag{B.2b}$$

$$z_{k+1} - z_k = \tau \left(x_{k+1} - z_k - \frac{1}{\mu \gamma_k} H_k \nabla f(x_{k+1}) \right) \,. \tag{B.2c}$$

It is easily verified that $(x, y, z) = (x_*, x_*, x_*)$ is a fixed point of the aforementioned dynamical system. Our goal for now is to describe conditions on f, μ, and τ such that the sequence of updates (B.2a), (B.2b), and (B.2c) converges to this fixed point. As described in Wilson et al. [29], our main strategy for proving convergence will be to introduce the following Lyapunov function

$$V_k = f(y_k) - f_* + \frac{\mu}{2} \|z_k - x_*\|_{G^{-1}}^2 \,, \tag{B.3}$$

and show that V_k decreases along every trajectory. We let $\mathbb{E}_k$ denote the expectation conditioned on $\mathcal{F}_k = \sigma(\omega_0, \omega_1, ..., \omega_{k-1})$. Observe that x_{k+1} is $\mathcal{F}_k$-measurable, a fact we will use repeatedly throughout our calculations. With the preceding definitions in place, we state and prove our main structural theorem.

Theorem B.1. *Let f and G be as defined above, with f satisfying μ-strongly convexity and L-Lipschitz gradients with respect to the $\|\cdot\|_{G^{-1}}$ norm, as defined in (B.1a) and (B.1b). Suppose that for all fixed $x \in \mathbb{R}^n$, we have that the following holds for almost every $\omega \in \Omega$,*

$$f(\Phi(x; \omega)) \leq f(x) - \frac{1}{2\Gamma} \|\nabla f(x)\|_H^2 \,, \;\; \Phi(x; \omega) = x - \frac{1}{\Gamma} H \nabla f(x) \,. \tag{B.4}$$

Furthermore, suppose that $\nu > 0$ satisfies

$$\mathbb{E}\left[\frac{1}{\gamma^2} H G^{-1} H \right] \preccurlyeq \nu \mathbb{E}\left[\frac{1}{\gamma^2} H \right] \,. \tag{B.5}$$

Then as long as we set $\tau > 0$ such that τ satisfies for almost every $\omega \in \Omega$,

$$\tau \leq \frac{\gamma}{\sqrt{\Gamma}} \sqrt{\frac{\mu}{\nu}} \,, \;\; \tau \leq \sqrt{\frac{\mu}{L}} \,, \tag{B.6}$$

we have that V_k defined in (B.3) satisfies for all $k \geq 0$,

$$\mathbb{E}_k[V_{k+1}] \leq (1 - \tau) V_k \,. \tag{B.7}$$

Proof. First, recall the following two point equality valid for any vectors $a, b, c \in V$ in a real inner product space V,

$$\|a - b\|_V^2 - \|c - b\|_V^2 = \|a - c\|_V^2 + 2\langle a - c, c - b \rangle_V \,. \tag{B.8}$$

Now we can proceed with our analysis,

$$
\begin{aligned}
V_{k+1} - V_k &\overset{(B.8)}{=} f(y_{k+1}) - f(y_k) - \mu\langle z_{k+1} - z_k, x_* - z_k\rangle_{G^{-1}} + \frac{\mu}{2}\|z_{k+1} - z_k\|^2_{G^{-1}} \\
&= f(y_{k+1}) - f(x_{k+1}) + f(x_{k+1}) - f(y_k) - \mu\langle z_{k+1} - z_k, x_* - z_k\rangle_{G^{-1}} + \frac{\mu}{2}\|z_{k+1} - z_k\|^2_{G^{-1}} \\
&\overset{(B.1a)}{\leq} f(y_{k+1}) - f(x_{k+1}) + \langle \nabla f(x_{k+1}), x_{k+1} - y_k\rangle - \frac{\mu}{2}\|x_{k+1} - y_k\|^2_{G^{-1}} \\
&\quad - \mu\langle z_{k+1} - z_k, x_* - z_k\rangle_{G^{-1}} + \frac{\mu}{2}\|z_{k+1} - z_k\|^2_{G^{-1}} \qquad \text{(B.9a)} \\
&\overset{(B.2c)}{=} f(y_{k+1}) - f(x_{k+1}) + \langle \nabla f(x_{k+1}), x_{k+1} - y_k\rangle - \frac{\mu}{2}\|x_{k+1} - y_k\|^2_{G^{-1}} \\
&\quad + \tau\langle \frac{1}{\gamma_k} H_k \nabla f(x_{k+1}) - \mu(x_{k+1} - z_k), x_* - z_k\rangle_{G^{-1}} + \frac{\mu}{2}\|z_{k+1} - z_k\|^2_{G^{-1}} \qquad \text{(B.9b)} \\
&= f(y_{k+1}) - f(x_{k+1}) + \langle \nabla f(x_{k+1}), x_{k+1} - y_k\rangle - \frac{\mu}{2}\|x_{k+1} - y_k\|^2_{G^{-1}} \\
&\quad + \tau\langle \frac{1}{\gamma_k} H_k \nabla f(x_{k+1}), x_* - x_{k+1}\rangle_{G^{-1}} + \tau\langle \frac{1}{\gamma_k} H_k \nabla f(x_{k+1}), x_{k+1} - z_k\rangle_{G^{-1}} \\
&\quad - \tau\mu\langle x_{k+1} - z_k, x_* - z_k\rangle_{G^{-1}} + \frac{\mu}{2}\|z_{k+1} - z_k\|^2_{G^{-1}} \\
&\overset{(B.2c)}{=} f(y_{k+1}) - f(x_{k+1}) + \langle \nabla f(x_{k+1}), x_{k+1} - y_k\rangle - \frac{\mu}{2}\|x_{k+1} - y_k\|^2_{G^{-1}} \\
&\quad + \tau\langle \frac{1}{\gamma_k} H_k \nabla f(x_{k+1}), x_* - x_{k+1}\rangle_{G^{-1}} + \tau\langle \frac{1}{\gamma_k} H_k \nabla f(x_{k+1}), x_{k+1} - z_k\rangle_{G^{-1}} \\
&\quad - \tau\mu\langle x_{k+1} - z_k, x_* - z_k\rangle_{G^{-1}} + \frac{\mu}{2}\|\tau(x_{k+1} - z_k)\|^2_{G^{-1}} + \frac{\tau^2}{2\mu\gamma_k^2}\|H_k \nabla f(x_{k+1})\|^2_{G^{-1}} \\
&\quad - \tau\langle x_{k+1} - z_k, \tau\frac{1}{\gamma_k} H_k \nabla f(x_{k+1})\rangle_{G^{-1}} \qquad \text{(B.9c)} \\
&\overset{(B.4)}{\leq} -\frac{1}{2\Gamma_k}\|\nabla f(x_{k+1})\|^2_{H_k} + \langle \nabla f(x_{k+1}), x_{k+1} - y_k\rangle - \frac{\mu}{2}\|x_{k+1} - y_k\|^2_{G^{-1}} \\
&\quad + \tau\langle \frac{1}{\gamma_k} H_k \nabla f(x_{k+1}), x_* - x_{k+1}\rangle_{G^{-1}} + \tau\langle \frac{1}{\gamma_k} H_k \nabla f(x_{k+1}), x_{k+1} - z_k\rangle_{G^{-1}} \\
&\quad - \tau\mu\langle x_{k+1} - z_k, x_* - z_k\rangle_{G^{-1}} + \frac{\mu}{2}\|\tau(x_{k+1} - z_k)\|^2_{G^{-1}} + \frac{\tau^2}{2\mu\gamma_k^2}\|H_k \nabla f(x_{k+1})\|^2_{G^{-1}} \\
&\quad - \tau\langle x_{k+1} - z_k, \tau\frac{1}{\gamma_k} H_k \nabla f(x_{k+1})\rangle_{G^{-1}} \,. \qquad \text{(B.9d)}
\end{aligned}
$$

Above, (B.9a) follows from μ-strong convexity, (B.9b) and (B.9c) both use the definition of the update sequence given in (B.2), and (B.9d) follows using the gradient inequality assumption (B.4). Now letting $x \in \mathbb{R}^n$ be fixed, we observe that

$$
\begin{aligned}
\mathbb{E}\left[\frac{\tau^2}{2\mu\gamma^2}\nabla f(x)^\mathsf{T} H G^{-1} H \nabla f(x) - \frac{1}{2\Gamma}\|\nabla f(x)\|^2_H\right] &\overset{(B.5)}{\leq} \mathbb{E}\left[\left(\frac{\tau^2\nu}{2\mu\gamma^2} - \frac{1}{2\Gamma}\right)\|\nabla f(x)\|^2_H\right] \\
&\overset{(B.6)}{\leq} 0\,. \qquad \text{(B.10)}
\end{aligned}
$$

The first inequality uses the assumption on ν, and the second inequality uses the requirement that $\tau \leq \frac{\gamma}{\sqrt{\Gamma}}\sqrt{\frac{\mu}{\nu}}$.

Now taking expectations with respect to $\mathbb{E}_k$,

$$\begin{aligned}
\mathbb{E}_k[V_{k+1}] - V_k \leq\ & \mathbb{E}_k\left[\frac{\tau^2}{2\mu\gamma_k^2}\nabla f(x_{k+1})^\top H_k G^{-1} H_k \nabla f(x_{k+1}) - \frac{1}{2\Gamma_k}\|\nabla f(x_{k+1})\|_{H_k}^2\right] \\
& + \langle \nabla f(x_{k+1}), x_{k+1} - y_k\rangle - \frac{\mu}{2}\|x_{k+1} - y_k\|_{G^{-1}}^2 \\
& + \tau\langle \nabla f(x_{k+1}), x_* - x_{k+1}\rangle + \tau\langle \nabla f(x_{k+1}), x_{k+1} - z_k\rangle - \tau\mu\langle x_{k+1} - z_k, x_* - z_k\rangle_{G^{-1}} \\
& + \frac{\mu}{2}\|\tau(x_{k+1} - z_k)\|_{G^{-1}}^2 - \tau\langle x_{k+1} - z_k, \tau\nabla f(x_{k+1})\rangle \\
\overset{(B.10)}{\leq}\ & \langle \nabla f(x_{k+1}), x_{k+1} - y_k\rangle - \frac{\mu}{2}\|x_{k+1} - y_k\|_{G^{-1}}^2 + \tau\langle \nabla f(x_{k+1}), x_* - x_{k+1}\rangle \\
& + \tau\langle \nabla f(x_{k+1}), x_{k+1} - z_k\rangle - \tau\mu\langle x_{k+1} - z_k, x_* - z_k\rangle_{G^{-1}} \\
& + \frac{\mu}{2}\|\tau(x_{k+1} - z_k)\|_{G^{-1}}^2 - \tau\langle x_{k+1} - z_k, \tau\nabla f(x_{k+1})\rangle \\
\overset{(B.1a)}{\leq}\ & -\tau\left(f(x_{k+1}) - f_* + \frac{\mu}{2}\|x_{k+1} - x_*\|_{G^{-1}}^2\right) + \langle \nabla f(x_{k+1}), x_{k+1} - y_k\rangle - \frac{\mu}{2}\|x_{k+1} - y_k\|_{G^{-1}}^2 \\
& + \tau\langle \nabla f(x_{k+1}), x_{k+1} - z_k\rangle - \tau\mu\langle x_{k+1} - z_k, x_* - z_k\rangle_{G^{-1}} \\
& + \frac{\mu}{2}\|\tau(x_{k+1} - z_k)\|_{G^{-1}}^2 - \tau\langle x_{k+1} - z_k, \tau\nabla f(x_{k+1})\rangle \qquad \text{(B.11a)} \\
\overset{(B.2a)}{=}\ & -\tau\left(f(x_{k+1}) - f_* + \frac{\mu}{2}\|x_{k+1} - x_*\|_{G^{-1}}^2\right) - \frac{\mu}{2}\|x_{k+1} - y_k\|_{G^{-1}}^2 - \tau\mu\langle x_{k+1} - z_k, x_* - z_k\rangle_{G^{-1}} \\
& + \frac{\mu}{2}\|\tau(x_{k+1} - z_k)\|_{G^{-1}}^2 - \tau\langle y_k - x_{k+1}, \nabla f(x_{k+1})\rangle \qquad \text{(B.11b)} \\
\overset{(B.1b)}{\leq}\ & -\tau\left(f(x_{k+1}) - f_* + \frac{\mu}{2}\|x_{k+1} - x_*\|_{G^{-1}}^2\right) - \frac{\mu}{2}\|x_{k+1} - y_k\|_{G^{-1}}^2 - \tau\mu\langle x_{k+1} - z_k, x_* - z_k\rangle_{G^{-1}} \\
& + \frac{\mu}{2}\|\tau(x_{k+1} - z_k)\|_{G^{-1}}^2 + \tau(f(x_{k+1}) - f(y_k)) + \frac{\tau L}{2}\|y_k - x_{k+1}\|_{G^{-1}}^2 \qquad \text{(B.11c)} \\
\overset{(B.8)}{=}\ & -\tau\left(f(x_{k+1}) - f_* + \frac{\mu}{2}\|x_{k+1} - z_k\|_{G^{-1}}^2 + \frac{\mu}{2}\|z_k - x_*\|_{G^{-1}}^2 + \mu\langle x_{k+1} - z_k, z_k - x_*\rangle_{G^{-1}}\right) \\
& - \frac{\mu}{2}\|x_{k+1} - y_k\|_{G^{-1}}^2 - \tau\mu\langle x_{k+1} - z_k, x_* - z_k\rangle_{G^{-1}} \\
& + \frac{\mu}{2}\|\tau(x_{k+1} - z_k)\|_{G^{-1}}^2 + \tau(f(x_{k+1}) - f(y_k)) + \frac{\tau L}{2}\|y_k - x_{k+1}\|_{G^{-1}}^2 \qquad \text{(B.11d)} \\
\overset{(B.3)}{=}\ & -\tau V_k - \frac{\mu}{2}\|x_{k+1} - y_k\|_{G^{-1}}^2 - \frac{\tau\mu}{2}\|x_{k+1} - z_k\|_{G^{-1}}^2 + \frac{\mu}{2}\|\tau(x_{k+1} - z_k)\|_{G^{-1}}^2 + \frac{\tau L}{2}\|y_k - x_{k+1}\|_{G^{-1}}^2 \\
\overset{(B.2a)}{=}\ & -\tau V_k + \left(\frac{\tau L}{2} - \frac{\mu}{2\tau}\right)\|y_k - x_{k+1}\|_{G^{-1}}^2 \qquad \text{(B.11e)} \\
\overset{(B.6)}{\leq}\ & -\tau V_k\,.
\end{aligned}$$

Above, (B.11a) follows from μ-strong convexity, (B.11b) and (B.11e) both use the definition of the sequence (B.2), (B.11c) follows from L-Lipschitz gradients, (B.11d) uses the two-point inequality (B.8), and the last inequality follows from the assumption of $\tau \leq \sqrt{\frac{\mu}{L}}$. The claim (B.7) now follows by re-arrangement. □

B.1 Proof of Theorem 4.3

Next, we describe how to recover Theorem 4.3 from Theorem B.1. We do this by applying Theorem B.1 to the function $f(x) = \frac{1}{2}x^\top Ax - x^\top b$.

The first step in applying Theorem B.1 is to construct a probability measure on $\mathcal{S}^{n\times n} \times \mathbb{R}_+ \times \mathbb{R}_+$ for which the randomness of the updates is drawn from. We already have a distribution on $\mathcal{S}^{n\times n}$ from setting of Theorem 4.3 via the random matrix H. We trivially augment this distribution by considering the

random variable $(H, 1, 1) \in \Omega$. By setting $\Gamma = \gamma = 1$, the sequence (B.2a), (B.2b), (B.2c) reduces to that of Algorithm 2. Furthermore, the requirement on the ν parameter from (B.5) simplifies to the requirement listed in (4.4). This holds by the following equivalences which are valid since conjugation by G (which is assumed to be positive definite) preserves the semi-definite ordering,

$$\begin{aligned}\lambda_{\max}\left(\mathbb{E}\left[(G^{-1/2}HG^{-1/2})^2\right]\right) \leq \nu &\iff \mathbb{E}\left[(G^{-1/2}HG^{-1/2})^2\right] \preccurlyeq \nu I \\ &\iff \mathbb{E}\left[G^{-1/2}HG^{-1}HG^{-1/2}\right] \preccurlyeq \nu I \\ &\iff \mathbb{E}\left[HG^{-1}H\right] \preccurlyeq \nu G\,. \end{aligned} \tag{B.12}$$

It remains to check the gradient inequality (B.4) and compute the strong convexity and Lipschitz parameters. These computations fall directly from the calculations made in Theorem 1 of [19], but we replicate them here for completeness.

To check the gradient inequality (B.4), because f is a quadratic function, its second order Taylor expansion is exact. Hence for almost every $\omega \in \Omega$,

$$\begin{aligned} f(\Phi(x;\omega)) &= f(x) - \langle \nabla f(x), H\nabla f(x)\rangle + \frac{1}{2}\nabla f(x)^\mathsf{T} HAH\nabla f(x) \\ &= f(x) - \langle \nabla f(x), H\nabla f(x)\rangle + \frac{1}{2}\nabla f(x)^\mathsf{T} S(S^\mathsf{T}AS)^\dagger S^\mathsf{T}AS(S^\mathsf{T}AS)^\dagger S^\mathsf{T}\nabla f(x) \\ &= f(x) - \langle \nabla f(x), H\nabla f(x)\rangle + \frac{1}{2}\nabla f(x)^\mathsf{T} S(S^\mathsf{T}AS)^\dagger S^\mathsf{T}\nabla f(x) \\ &= f(x) - \frac{1}{2}\nabla f(x)^\mathsf{T} H\nabla f(x)\,. \end{aligned}$$

Hence the inequality (B.4) holds with equality.

We next compute the strong convexity and Lipschitz gradient parameters. We first show that f is $\lambda_{\min}(\mathbb{E}[P_{A^{1/2}S}])$-strongly convex with respect to the $\|\cdot\|_{G^{-1}}$ norm. This follows since for any $x, y \in \mathbb{R}^n$, using the assumption that G is positive definite,

$$\begin{aligned} f(y) &= f(x) + \langle \nabla f(x), y - x\rangle + \frac{1}{2}(y-x)^\mathsf{T} A(y-x) \\ &= f(x) + \langle \nabla f(x), y - x\rangle + \frac{1}{2}(y-x)^\mathsf{T} G^{-1/2}G^{1/2}AG^{1/2}G^{-1/2}(y-x) \\ &\geq f(x) + \langle \nabla f(x), y - x\rangle + \frac{\lambda_{\min}(A^{1/2}GA^{1/2})}{2}\|y-x\|_{G^{-1}}^2\,. \end{aligned}$$

The strong convexity bound now follows since

$$A^{1/2}GA^{1/2} = A^{1/2}\mathbb{E}[H]A^{1/2} = \mathbb{E}[A^{1/2}S(S^\mathsf{T}AS)^\dagger S^\mathsf{T}A^{1/2}] = \mathbb{E}[P_{A^{1/2}S}]\,.$$

An nearly identical argument shows that f is $\lambda_{\max}(\mathbb{E}[P_{A^{1/2}S}])$-strongly convex with respect to the $\|\cdot\|_{G^{-1}}$ norm. Since the eigenvalues of projector matrices are bounded by 1, we have that f is 1-Lipschitz with respect to the $\|\cdot\|_{G^{-1}}$ norm. This calculation shows that the requirement on τ from (B.6) simplifies to $\tau \leq \sqrt{\frac{\mu}{\nu}}$, since $L = 1$ and $\nu \geq 1$ by Proposition D.1 which we state and prove later.

At this point, Theorem B.1 yields that $\mathbb{E}[V_k] \leq (1-\tau)^k V_0$. To recover the final claim (4.5), recall that $f(y_k) - f_* = \frac{1}{2}\|y_k - x_*\|_A^2$. Furthermore, $\mu G^{-1} \preccurlyeq A$, since

$$\begin{aligned} \mu \leq \lambda_{\min}(A^{1/2}GA^{1/2}) &\iff \mu \leq \lambda_{\min}(G^{1/2}AG^{1/2}) \\ &\iff \mu I \preccurlyeq G^{1/2}AG^{1/2} \\ &\iff \mu G^{-1} \preccurlyeq A\,. \end{aligned}$$

Hence, we can upper bound V_0 as follows

$$V_0 = f(y_0) - f_* + \frac{\mu}{2}\|z_0 - x_*\|^2_{G^{-1}} = \frac{1}{2}\|y_0 - x_*\|^2_A + \frac{\mu}{2}\|z_0 - x_*\|^2_{G^{-1}}$$
$$\leq \frac{1}{2}\|y_0 - x_*\|^2_A + \frac{1}{2}\|z_0 - x_*\|^2_A = \|x_0 - x_*\|^2_A \,.$$

On the other hand, we have that $\frac{1}{2}\|y_k - x_*\|^2_A \leq V_k$. Putting the inequalities together,

$$\frac{1}{\sqrt{2}}\mathbb{E}[\|y_k - x_*\|_A] \leq \sqrt{\mathbb{E}[\frac{1}{2}\|y_k - x_*\|^2_A]} \leq \sqrt{\mathbb{E}[V_k]} \leq \sqrt{(1-\tau)^k V_0} \leq (1-\tau)^{k/2}\|x_0 - x_*\|^2_A \,,$$

where the first inequality holds by Jensen's inequality. The claimed inequality (4.5) now follows.

B.2 Recovering ACDM result from Nesterov and Stich [16] using Theorem B.1

We next show how to recover Theorem 1 of Nesterov and Stich [16] using Theorem B.1, in the case of $\alpha = 1$. A nearly identical argument can also be used to recover the result of Allen-Zhu et al. [1] under the strongly convex setting in the case of $\beta = 0$. Our argument proceeds in two steps. First, we prove a convergence result for a simplified accelerated coordinate descent method which we introduce in Algorithm 3. Then, we describe how a minor tweak to ACDM shows the equivalence between ACDM and Algorithm 3.

Before we proceed, we first describe the setting of Theorem 1. Let $f : \mathbb{R}^n \longrightarrow \mathbb{R}$ be a twice differentiable strongly convex function with Lipschitz gradients. Let $J_1, ..., J_m$ denote a partition of $\{1, ..., n\}$ into m partitions. Without loss of generality, we can assume that the partitions are in order, i.e. $J_1 = \{1, ..., n_1\}$, $J_2 = \{n_1 + 1, ..., n_2\}$, and so on. This is without loss of generality since we can always consider the function $g(x) = f(\Pi x)$ for a suitable permutation matrix Π. Let $B_1, ..., B_m$ be fixed positive definite matrices such that $B_i \in \mathbb{R}^{|J_i| \times |J_i|}$. Set $H_i = S_i B_i^{-1} S_i^\mathsf{T}$, where $S_i \in \mathbb{R}^{n \times |J_i|}$ is the column selector matrix associated to partition J_i, and define $L_i = \sup_{x \in \mathbb{R}^n} \lambda_{\max}(B_i^{-1/2} S_i^\mathsf{T} \nabla^2 f(x) S_i B_i^{-1/2})$ for $i = 1, ..., m$. Furthermore, define $p_i = \frac{\sqrt{L_i}}{\sum_{j=1}^m \sqrt{L_j}}$.

B.2.1 Proof of convergence of a simplified accelerated coordinate descent method

Now consider the following accelerated randomized coordinate descent algorithm in Algorithm 3.

Algorithm 3 Accelerated randomized coordinate descent.

Require: $\mu > 0$, partition $\{J_i\}_{i=1}^m$, positive definite $\{B_i\}_{i=1}^m$, Lipschitz constants $\{L_i\}_{i=1}^m$, $x_0 \in \mathbb{R}^n$.

Set $\tau = \frac{\sqrt{\mu}}{\sum_{i=1}^m \sqrt{L_i}}$.
Set $H_i = S_i B_i^{-1} S_i^\mathsf{T}$ for $i = 1, ..., m$. `/* S_i denotes the column selector for partition J_i. */`
Set $p_i = \frac{\sqrt{L_i}}{\sum_{j=1}^m \sqrt{L_j}}$ for $i = 1, ..., m$.
Set $y_0 = z_0 = x_0$.
for $k = 0, ..., T-1$ **do**
    $i_k \leftarrow$ random sample from $\{1, ..., m\}$ with $\mathbb{P}(i_k = i) = p_i$.
    $x_{k+1} = \frac{1}{1+\tau} y_k + \frac{\tau}{1+\tau} z_k$.
    $y_{k+1} = x_{k+1} - \frac{1}{L_{i_k}} H_{i_k} \nabla f(x_{k+1})$.
    $z_{k+1} = z_k + \tau(x_{k+1} - z_k) - \frac{\tau}{\mu p_{i_k}} H_{i_k} \nabla f(x_{k+1})$.
end for
return y_T.

Theorem B.1 is readily applied to Algorithm 3 to give a convergence guarantee which matches the bound of Theorem 1 of Nesterov and Stich. We sketch the argument below.

Algorithm 3 instantiates (B.2) with the definitions above and particular choices $\Gamma_k = L_{i_k}$ and $\gamma_k = p_{i_k}$. We will specify the choice of μ at a later point. To see that this setting is valid, we construct a discrete probability measure on $\mathcal{S}^{n\times n} \times \mathbb{R}_+ \times \mathbb{R}_+$ by setting $\omega_i = (H_i, L_i, p_i)$ and $\mathbb{P}(\omega = \omega_i) = p_i$ for $i = 1, ..., m$. Hence, in the context of Theorem B.1, $G = \mathbb{E}[\frac{1}{\gamma} H] = \sum_{i=1}^m H_i = \text{blkdiag}(B_1^{-1}, B_2^{-1}, ..., B_m^{-1})$. We first verify the gradient inequality (B.4). For every fixed $x \in \mathbb{R}^n$, for every $i = 1, ..., m$ there exists a $c_i \in \mathbb{R}^n$ such that

$$\begin{aligned}
f(\Phi(x;\omega_i)) &= f(x) - \frac{1}{L_i}\langle \nabla f(x), H_i \nabla f(x)\rangle + \frac{1}{2L_i^2} \nabla f(x)^\mathsf{T} H_i \nabla^2 f(c_i) H_i \nabla f(x) \\
&= f(x) - \frac{1}{L_i}\langle \nabla f(x), H_i \nabla f(x)\rangle + \frac{1}{2L_i^2} \nabla f(x)^\mathsf{T} S_i B_i^{-1/2} B_i^{-1/2} S_i^\mathsf{T} \nabla^2 f(c_i) S_i B_i^{-1/2} B_i^{-1/2} S_i^\mathsf{T} \nabla f(x) \\
&\leq f(x) - \frac{1}{L_i}\langle \nabla f(x), H_i \nabla f(x)\rangle + \frac{1}{2L_i} \nabla f(x)^\mathsf{T} S_i B_i^{-1} S_i^\mathsf{T} \nabla f(x) \\
&= f(x) - \frac{1}{2L_i} \|\nabla f(x)\|_{H_i}^2 .
\end{aligned}$$

We next compute the ν constant defined in (B.5). We do this by checking the sufficient condition that $H_i G^{-1} H_i \preccurlyeq \nu H_i$ for $i = 1, ..., m$. Doing so yields that $\nu = 1$, since

$$H_i G^{-1} H_i = S_i B_i^{-1} S_i^\mathsf{T} \text{blkdiag}(B_1, B_2, ..., B_m) S_i B_i^{-1} S_i^\mathsf{T} = S_i B_i^{-1} B_i B_i^{-1} S_i^\mathsf{T} = S_i B_i^{-1} S_i^\mathsf{T} = H_i .$$

To complete the argument, we set μ as the strong convexity constant and L as the Lipschitz gradient constant of f with respect to the $\|\cdot\|_{G^{-1}}$ norm. It is straightforward to check that

$$\mu = \inf_{x\in\mathbb{R}^n} \lambda_{\max}(G^{1/2} \nabla^2 f(x) G^{1/2}) , \quad L = \sup_{x\in\mathbb{R}^n} \lambda_{\max}(G^{1/2} \nabla^2 f(x) G^{1/2}) .$$

We now argue that $\sqrt{L} \leq \sum_{i=1}^m \sqrt{L_i}$. Let $x \in \mathbb{R}^n$ achieve the supremum in the definition of L (if no such x exists, then let x be arbitrarily close and take limits). Then,

$$\begin{aligned}
L &= \lambda_{\max}(G^{1/2} \nabla^2 f(x) G^{1/2}) = \lambda_{\max}((\nabla^2 f(x))^{1/2} G (\nabla^2 f(x))^{1/2}) \\
&= \lambda_{\max}\left((\nabla^2 f(x))^{1/2} \left(\sum_{i=1}^m S_i B_i^{-1} S_i^\mathsf{T} \right) (\nabla^2 f(x))^{1/2} \right) \\
&\overset{(a)}{\leq} \sum_{i=1}^m \lambda_{\max}((\nabla^2 f(x))^{1/2} S_i B_i^{-1} S_i^\mathsf{T} (\nabla^2 f(x))^{1/2}) \\
&\overset{(b)}{=} \sum_{i=1}^m \lambda_{\max}(S_i S_i^\mathsf{T} \nabla^2 f(x) S_i S_i^\mathsf{T} S_i B_i^{-1} S_i^\mathsf{T}) \\
&\overset{(c)}{=} \sum_{i=1}^m \lambda_{\max}(B_i^{-1/2} S_i^\mathsf{T} \nabla^2 f(x) S_i B_i^{-1/2}) \leq \sum_{i=1}^m L_i .
\end{aligned}$$

Above, (a) follows by the triangle inequality, (b) follows by the fact that $S_i^\mathsf{T} S_i = I$, and (c) follows since $\lambda_{\max}(S_i M S_i^\mathsf{T}) = \lambda_{\max}(M)$ for any $p \times p$ symmetric matrix M. Using the fact that $\sqrt{a+b} \leq \sqrt{a} + \sqrt{b}$ for any non-negative a, b, the inequality $\sqrt{L} \leq \sum_{i=1}^m \sqrt{L_i}$ immediately follows. To conclude the proof, it remains to calculate the requirement on τ via (B.6). Since $\frac{\gamma_i}{\sqrt{\Gamma_i}} = \frac{p_i}{\sqrt{L_i}} = \frac{1}{\sum_{i=1}^m \sqrt{L_i}}$, we have that $\frac{\gamma_i}{\sqrt{\Gamma_i}} \leq \frac{1}{\sqrt{L}}$, and hence the requirement is that $\tau \leq \frac{\sqrt{\mu}}{\sum_{i=1}^m \sqrt{L_i}}$.

B.2.2 Relating Algorithm 3 to ACDM

For completeness, we replicate the description of the ACDM algorithm from Nesterov and Stich in Algorithm 4. We make one minor tweak in the initialization of the A_k, B_k sequence which greatly simplifies the exposition of what follows.

Algorithm 4 ACDM from Nesterov and Stich [16], $\alpha = 1, \beta = 1/2$ case.

Require: $\mu > 0$, partition $\{J_i\}_{i=1}^m$, positive definite $\{B_i\}_{i=1}^m$, Lipschitz constants $\{L_i\}_{i=1}^m$, $x_0 \in \mathbb{R}^n$.

Set $H_i = S_i B_i^{-1} S_i^\mathsf{T}$ for $i = 1, ..., m$. /* S_i denotes the column selector for partition J_i. */

Set $p_i = \frac{\sqrt{L_i}}{\sum_{j=1}^m \sqrt{L_j}}$ for $i = 1, ..., m$.

Set $A_0 = 1, B_0 = \mu$. /* Modified from $A_0 = 0, B_0 = 1$. */

Set $S_{1/2} = \sum_{i=1}^m \sqrt{L_i}$.

Set $y_0 = z_0 = x_0$.

for $k = 0, ..., T-1$ **do**

  $i_k \leftarrow$ random sample from $\{1, ..., m\}$ with $\mathbb{P}(i_k = i) = p_i$.

  $a_{k+1} \leftarrow$ positive solution to $a_{k+1}^2 S_{1/2}^2 = (A_k + a_{k+1})(B_k + \mu a_{k+1})$.

  $A_{k+1} = A_k + a_{k+1}, B_{k+1} = B_k + \mu a_{k+1}$.

  $\alpha_k = \frac{a_{k+1}}{A_{k+1}}, \beta_k = \mu \frac{a_{k+1}}{B_{k+1}}$.

  $y_k = \frac{(1-\alpha_k)x_k + \alpha_k(1-\beta_k)z_k}{1-\alpha_k\beta_k}$.

  $x_{k+1} = y_k - \frac{1}{L_{i_k}} H_{i_k} \nabla f(y_k)$.

  $z_{k+1} = (1-\beta_k)z_k + \beta_k y_k - \frac{a_{k+1}}{B_{k+1} p_{i_k}} H_{i_k} \nabla f(y_k)$.

end for

return x_T.

We first write the sequence produced by Algorithm 4 as

$$y_k = \frac{(1-\alpha_k)x_k + \alpha_k(1-\beta_k)z_k}{1-\alpha_k\beta_k}\,, \tag{B.13a}$$

$$x_{k+1} = y_k - \frac{1}{L_{i_k}} H_{i_k} \nabla f(y_k)\,, \tag{B.13b}$$

$$z_{k+1} - z_k = \beta_k \left(y_k - z_k - \frac{a_{k+1}}{B_{k+1} p_{i_k} \beta_k} H_{i_k} \nabla f(y_k) \right)\,. \tag{B.13c}$$

Since $\beta_k B_{k+1} = \mu a_{k+1}$, the z_{k+1} update simplifies to

$$z_{k+1} - z_k = \beta_k \left(y_k - z_k - \frac{1}{\mu p_{i_k}} H_{i_k} \nabla f(y_k) \right)\,.$$

A simple calculation shows that

$$(1-\alpha_k\beta_k)y_k = (1-\alpha_k)x_k + \alpha_k(1-\beta_k)z_k\,,$$

from which we conclude that

$$\frac{\alpha_k(1-\beta_k)}{1-\alpha_k}(y_k - z_k) = x_k - y_k\,. \tag{B.14}$$

Observe that

$$A_{k+1} = \sum_{i=1}^{k+1} a_i + A_0\,,\quad B_{k+1} = \mu \sum_{i=1}^{k+1} a_i + B_0\,.$$

Hence as long as $\mu A_0 = B_0$ (which is satisfied by our modification), we have that $\mu A_{k+1} = B_{k+1}$ for all $k \geq 0$. With this identity, we have that $\alpha_k = \beta_k$ for all $k \geq 0$. Therefore, (B.14) simplifies to

$$\beta_k(y_k - z_k) = x_k - y_k\,.$$

We now calculate the value of β_k. At every iteration, we have that

$$a_{k+1}^2 S_{1/2}^2 = A_{k+1} B_{k+1} = \frac{1}{\mu} B_{k+1}^2 \implies \frac{a_{k+1}}{B_{k+1}} = \frac{1}{\sqrt{\mu} S_{1/2}} .$$

By the definition of β_k,

$$\beta_k = \mu \frac{a_{k+1}}{B_{k+1}} = \frac{\sqrt{\mu}}{S_{1/2}} = \frac{\sqrt{\mu}}{\sum_{i=1}^m \sqrt{L_i}} = \tau .$$

Combining these identities, we have shown that (B.13a), (B.13b), and (B.13c) simplifies to

$$y_k = \frac{1}{1+\tau} x_k + \frac{\tau}{1+\tau} z_k , \tag{B.15a}$$

$$x_{k+1} = y_k - \frac{1}{L_{i_k}} H_{i_k} \nabla f(y_k) , \tag{B.15b}$$

$$z_{k+1} - z_k = \tau \left(y_k - z_k - \frac{1}{\mu p_{i_k}} H_{i_k} \nabla f(y_k) \right) . \tag{B.15c}$$

This sequence directly coincides with the sequence generated by Algorithm 3 after a simple relabeling.

B.2.3 Recovering Algorithm 1 from ACDM

We describe the instantiation of Algorithm 1 using the notation described in the previous section. As mentioned previously, we set the function $f(x) = \frac{1}{2} x^\mathsf{T} A x - x^\mathsf{T} b$. Given a partition $\{J_i\}_{i=1}^{n/p}$, we let $B_i = S_i^\mathsf{T} A S_i$, where $S_i \in \mathbb{R}^{n \times p}$ is the column selector matrix associated to the partition J_i. With this setting, we have that $L_1 = L_2 = ... = L_{n/p} = 1$, and hence we have $p_i = p/n$ for all i (i.e. the sampling distribution is uniform over all partitions). We now need to compute the strong convexity constant μ. With the simplifying assumption that the partitions are ordered, μ is simply the strong convexity constant with respect to the norm induced by the matrix $\mathrm{blkdiag}(B_1, B_2, ..., B_{n/p})$. Hence, using the definition of μ_{part} from (2.4), we have that $\mu = \frac{n}{p} \mu_{\text{part}}$. Algorithm 1 now follows from plugging our particular choices of f and the constants into Algorithm 4.

B.3 A result for randomized block Kaczmarz using Theorem B.1

We now use Theorem B.1 to derive a result similar to Theorem 4.3 for the randomized accelerated Kaczmarz algorithm. In this setting, we let $A \in \mathbb{R}^{m \times n}$, $m \geq n$ be a matrix with full column rank, and $b \in \mathbb{R}^m$ such that $b \in \mathcal{R}(A)$. That is, there exists a unique $x_* \in \mathbb{R}^n$ such that $Ax_* = b$. We note that this section generalizes the result of [11] to the block case (although the proof strategy is quite different).

We first describe the randomized accelerated block Kaczmarz algorithm in Algorithm 5. Our main convergence result concerning Algorithm 5 is presented in Theorem B.2.

Theorem B.2. *Let A be an $m \times n$ matrix with full column rank, and $b \in \mathcal{R}(A)$. Let $x_* \in \mathbb{R}^n$ denote the unique vector satisfying $Ax_* = b$. Suppose each S_k, $k = 0, 1, 2, ...$ is an independent copy of a random sketching matrix $S \in \mathbb{R}^{m \times p}$. Let $\mu = \lambda_{\min}(\mathbb{E}[P_{A^\mathsf{T} S}])$. Suppose the distribution of S satisfies $\mu > 0$. Invoke Algorithm 5 with μ and ν, where ν is defined as*

$$\nu = \lambda_{\max} \left(\mathbb{E} \left[(G^{-1/2} H G^{-1/2})^2 \right] \right) , \quad G = \mathbb{E}[H] , \quad H = P_{A^\mathsf{T} S} . \tag{B.16}$$

Then for all $k \geq 0$ we have

$$\mathbb{E}[\|y_k - x_*\|_2] \leq \sqrt{2} \left(1 - \sqrt{\frac{\mu}{\nu}} \right)^{k/2} \|x_0 - x_*\|_2 . \tag{B.17}$$

Algorithm 5 Accelerated randomized block Kaczmarz.

Require: $A \in \mathbb{R}^{m\times n}$, A full column rank, $b \in \mathcal{R}(A)$, sketching matrices $\{S_k\}_{k=0}^{T-1} \subseteq \mathbb{R}^{m\times p}$, $x_0 \in \mathbb{R}^n$, $\mu \in (0,1)$, $\nu \geq 1$.

Set $\tau = \sqrt{\mu/\nu}$.
Set $y_0 = z_0 = x_0$.
for $k = 0, ..., T-1$ **do**
    $x_{k+1} = \frac{1}{1+\tau} y_k + \frac{\tau}{1+\tau} z_k$.
    $y_{k+1} = x_{k+1} - (S_k^\mathsf{T} A)^\dagger S_k^\mathsf{T}(Ax_{k+1} - b)$.
    $z_{k+1} = z_k + \tau(x_{k+1} - z_k) - \frac{\tau}{\mu}(S_k^\mathsf{T} A)^\dagger S_k^\mathsf{T}(Ax_{k+1} - b)$.
end for
return y_T.

Proof. The proof is very similar to that of Theorem 4.3, so we only sketch the main argument. The key idea is to use the correspondence between randomized Kaczmarz and coordinate descent (see e.g. Section 5.2 of [8]). To do this, we apply Theorem B.1 to $f(x) = \frac{1}{2}\|x - x_*\|_2^2$. As in the proof of Theorem 4.3, we construct a probability measure on $\mathcal{S}^{n\times n} \times \mathbb{R}_+ \times \mathbb{R}_+$ from the given random matrix H by considering the random variable $(H, 1, 1)$. To see that the sequence (B.2a), (B.2b), and (B.2c) induces the same update sequence as Algorithm 5, the crucial step is to notice that

$$\begin{aligned} H_k \nabla f(x_{k+1}) &= P_{A^\mathsf{T} S_k} \nabla f(x_{k+1}) = A^\mathsf{T} S_k (S_k^\mathsf{T} A A^\mathsf{T} S_k)^\dagger S_k^\mathsf{T} A(x_{k+1} - x_*) \\ &= A^\mathsf{T} S_k (S_k^\mathsf{T} A A^\mathsf{T} S_k)^\dagger S_k^\mathsf{T}(Ax_{k+1} - b) = (S_k^\mathsf{T} A)^\dagger S_k^\mathsf{T}(Ax_{k+1} - b)\,. \end{aligned}$$

Next, the fact that f is $\lambda_{\min}(\mathbb{E}[P_{A^\mathsf{T} S}])$-strongly convex and 1-Lipschitz with respect to the $\|\cdot\|_{G^{-1}}$ norm, where $G = \mathbb{E}[P_{A^\mathsf{T} S}]$, follows immediately by a nearly identical argument used in the proof of Theorem 4.3. It remains to check the gradient inequaity (B.4). Let $x \in \mathbb{R}^n$ be fixed. Then using the fact that f is quadratic, for almost every $\omega \in \Omega$,

$$\begin{aligned} f(\Phi(x;\omega)) &= f(x) - \langle \nabla f(x), H(x - x_*)\rangle + \frac{1}{2}\|H(x - x_*)\|_2^2 \\ &= f(x) - \langle x - x_*, P_{A^\mathsf{T} S}(x - x_*)\rangle + \frac{1}{2}\|P_{A^\mathsf{T} S}(x - x_*)\|_2^2 \\ &= f(x) - \frac{1}{2}\langle x - x_*, P_{A^\mathsf{T} S}(x - x_*)\rangle\,. \end{aligned}$$

Hence the gradient inequality (B.4) holds with equality. □

C Proofs for algorithmic lower bounds (Section 4.3)

C.1 Proof of Proposition 4.4

Define $e_k = x_k - x_*$, $H_k = S_k(S_k^\mathsf{T} A S_k)^\dagger S_k^\mathsf{T}$ and $G = \mathbb{E}[H_k]$. From the update rule (2.2),

$$e_{k+1} = (I - H_k A)e_k \implies A^{1/2} e_{k+1} = (I - A^{1/2} H_k A^{1/2}) A^{1/2} e_k\,.$$

Taking and iterating expectations,

$$\mathbb{E}[A^{1/2} e_{k+1}] = (I - A^{1/2} G A^{1/2})\mathbb{E}[A^{1/2} e_k]\,.$$

Unrollowing this recursion yields for all $k \geq 0$,

$$\mathbb{E}[A^{1/2} e_k] = (I - A^{1/2} G A^{1/2})^k A^{1/2} e_0\,.$$

Choose $A^{1/2}e_0 = v$, where v is an eigenvector of $I - A^{1/2}GA^{1/2}$ with eigenvalue $\lambda_{\max}(I - A^{1/2}GA^{1/2}) = 1 - \lambda_{\min}(GA) = 1 - \mu$. Now by Jensen's inequality,

$$\mathbb{E}[\|e_k\|_A] = \mathbb{E}[\|A^{1/2}e_k\|_2] \geq \|\mathbb{E}[A^{1/2}e_k]\|_2 = (1-\mu)^k \|e_0\|_A \,.$$

This establishes the claim.

C.2 Proof of Proposition 4.5

We first state and prove an elementary linear algebra fact which we will use below in our calculations.

Proposition C.1. *Let A, B, C, D be $n \times n$ diagonal matrices, and define $M = \begin{bmatrix} A & B \\ C & D \end{bmatrix}$. The eigenvalues of M are given by the union of the eigenvalues of the 2×2 matrices*

$$\begin{bmatrix} A_i & B_i \\ C_i & D_i \end{bmatrix}, \quad i = 1, ..., n \,,$$

where A_i, B_i, C_i, D_i denote the i-th digonal entry of A, B, C, D respectively.

Proof. For every $s \in \mathbb{C}$ we have that the matrices $-C$ and $sI - D$ are diagonal and hence commute. Applying the corresponding formula for a block matrix determinant under this assumption,

$$\begin{aligned} 0 &= \det \begin{bmatrix} sI - A & -B \\ -C & sI - D \end{bmatrix} = \det((sI - A)(sI - D) - BC) \\ &= \prod_{i=1}^{n} ((s - A_i)(s - D_i) - B_i C_i) = \prod_{i=1}^{n} \det \begin{bmatrix} s - A_i & -B_i \\ -C_i & s - D_i \end{bmatrix} . \end{aligned}$$

□

Now we proceed with the proof of Proposition 4.5. Define $e_k = \begin{bmatrix} y_k - x_* \\ z_k - x_* \end{bmatrix}$. It is easy to see from the definition of Algorithm 2 that $\{e_k\}$ satisfies the recurrence

$$e_{k+1} = \frac{1}{1+\tau} \begin{bmatrix} I - H_k A & \tau(I - H_k A) \\ \tau(I - \frac{1}{\mu} H_k A) & I - \frac{\tau^2}{\mu} H_k A \end{bmatrix} e_k \,.$$

Hence,

$$\begin{aligned} &\begin{bmatrix} A^{1/2} & 0 \\ 0 & \mu^{1/2} G^{-1/2} \end{bmatrix} e_{k+1} \\ &\quad = \frac{1}{1+\tau} \begin{bmatrix} A^{1/2} & 0 \\ 0 & \mu^{1/2} G^{-1/2} \end{bmatrix} \begin{bmatrix} I - H_k A & \tau(I - H_k A) \\ \tau(I - \frac{1}{\mu} H_k A) & I - \frac{\tau^2}{\mu} H_k A \end{bmatrix} e_k \\ &\quad = \frac{1}{1+\tau} \begin{bmatrix} A^{1/2} - A^{1/2} H_k A & \tau(A^{1/2} - A^{1/2} H_k A) \\ \mu^{1/2} \tau G^{-1/2}(I - \frac{1}{\mu} H_k A) & \mu^{1/2} G^{-1/2}(I - \frac{\tau^2}{\mu} H_k A) \end{bmatrix} e_k \\ &\quad = \frac{1}{1+\tau} \begin{bmatrix} I - A^{1/2} H_k A^{1/2} & \mu^{-1/2} \tau (A^{1/2} - A^{1/2} H_k A) G^{1/2} \\ \mu^{1/2} \tau G^{-1/2}(I - \frac{1}{\mu} H_k A) A^{-1/2} & G^{-1/2}(I - \frac{\tau^2}{\mu} H_k A) G^{1/2} \end{bmatrix} \begin{bmatrix} A^{1/2} & 0 \\ 0 & \mu^{1/2} G^{-1/2} \end{bmatrix} e_k \,. \end{aligned}$$

Define $P = \begin{bmatrix} A & 0 \\ 0 & \mu G^{-1} \end{bmatrix}$. By taking and iterating expectations,

$$\mathbb{E}[P^{1/2} e_{k+1}] = \frac{1}{1+\tau} \begin{bmatrix} I - A^{1/2} G A^{1/2} & \mu^{-1/2} \tau (A^{1/2} G^{1/2} - A^{1/2} G A G^{1/2}) \\ \mu^{1/2} \tau (G^{-1/2} A^{-1/2} - \frac{1}{\mu} G^{1/2} A^{1/2}) & I - \frac{\tau^2}{\mu} G^{1/2} A G^{1/2} \end{bmatrix} \mathbb{E}[P^{1/2} e_k] \,.$$

Denote the matrix $Q = A^{1/2}G^{1/2}$. Unrolling the recurrence above yields that

$$\mathbb{E}[P^{1/2}e_k] = R^k P^{1/2} e_0 \,, \;\; R = \frac{1}{1+\tau}\begin{bmatrix} I - QQ^\mathsf{T} & \mu^{-1/2}\tau(Q - QQ^\mathsf{T}Q) \\ \mu^{1/2}\tau(Q^{-1} - \frac{1}{\mu}Q^\mathsf{T}) & I - \frac{\tau^2}{\mu}Q^\mathsf{T}Q \end{bmatrix} .$$

Write the SVD of Q as $Q = U\Sigma V^\mathsf{T}$. Both U and V are $n \times n$ orthonormal matrices. It is easy to see that R^k is given by

$$R^k = \frac{1}{(1+\tau)^k}\begin{bmatrix} U & 0 \\ 0 & V \end{bmatrix}\begin{bmatrix} I - \Sigma^2 & \mu^{-1/2}\tau(\Sigma - \Sigma^3) \\ \mu^{1/2}\tau(\Sigma^{-1} - \frac{1}{\mu}\Sigma) & I - \frac{\tau^2}{\mu}\Sigma^2 \end{bmatrix}^k \begin{bmatrix} U^\mathsf{T} & 0 \\ 0 & V^\mathsf{T} \end{bmatrix} . \tag{C.1}$$

Suppose we choose $P^{1/2}e_0$ to be a right singular vector of R^k corresponding to the maximum singular value $\sigma_{\max}(R^k)$. Then we have that

$$\mathbb{E}[\|P^{1/2}e_k\|_2] \geq \|\mathbb{E}[P^{1/2}e_k]\|_2 = \|R^k P^{1/2} e_0\|_2 = \sigma_{\max}(R^k)\|P^{1/2}e_0\|_2 \geq \rho(R^k)\|P^{1/2}e_0\|_2 \,,$$

where $\rho(\cdot)$ denotes the spectral radius. The first inequality is Jensen's inequality, and the second inequality uses the fact that the spectral radius is bounded above by any matrix norm. The eigenvalues of R^k are the k-th power of the eigenvalues of R which, using the similarity transform (C.1) along with Proposition C.1, are given by the eigenvalues of the 2×2 matrices R_i defined as

$$R_i = \frac{1}{1+\tau}\begin{bmatrix} 1 - \sigma_i^2 & \mu^{-1/2}\tau(\sigma_i - \sigma_i^3) \\ \mu^{1/2}\tau(\sigma_i^{-1} - \frac{1}{\mu}\sigma_i) & 1 - \frac{\tau^2}{\mu}\sigma_i^2 \end{bmatrix} \,, \;\; \sigma_i = \Sigma_{ii} \,, \;\; i = 1, ..., n \,.$$

On the other hand, since the entries in Σ are given by the eigenvalues of $A^{1/2}G^{1/2}G^{1/2}A^{1/2} = \mathbb{E}[P_{A^{1/2}S}]$, there exists an i such that $\sigma_i = \sqrt{\mu}$. This R_i is upper triangular, and hence its eigenvalues can be read off the diagonal. This shows that $\frac{1-\tau^2}{1+\tau} = 1 - \tau$ is an eigenvalue of R, and hence $(1-\tau)^k$ is an eigenvalue of R^k. But this means that $(1-\tau)^k \leq \rho(R^k)$. Hence, we have shown that

$$\mathbb{E}[\|P^{1/2}e_k\|_2] \geq (1-\tau)^k \|P^{1/2}e_0\|_2 \,.$$

The desired claim now follows from

$$\begin{aligned} \|P^{1/2}e_k\|_2 &= \sqrt{\|y_k - x_*\|_A^2 + \mu\|z_k - x_*\|_{G^{-1}}^2} \\ &\leq \sqrt{\|y_k - x_*\|_A^2 + \|z_k - x_*\|_A^2} \leq \|y_k - x_*\|_A + \|z_k - x_*\|_A \,, \end{aligned}$$

where the first inequality holds since $\mu G^{-1} \preccurlyeq A$ and the second inequality holds since $\sqrt{a+b} \leq \sqrt{a} + \sqrt{b}$ for non-negative a, b.

D Proofs for random coordinate sampling (Section 4.4)

Our primary goal in this section is to provide a proof of Lemma 4.6. Along the way, we prove a few other results which are of independent interest. We first provide a proof of the lower bound claim in Lemma 4.6.

Proposition D.1. *Let A be an $n \times n$ matrix and let $S \in \mathbb{R}^{n \times p}$ be a random matrix. Put $G = \mathbb{E}[P_{A^{1/2}S}]$ and suppose that G is positive definite. Let $\nu > 0$ be any positive number such that*

$$\mathbb{E}[P_{A^{1/2}S}G^{-1}P_{A^{1/2}S}] \preccurlyeq \nu G \,, \;\; G = \mathbb{E}[P_{A^{1/2}S}] \,. \tag{D.1}$$

Then $\nu \geq n/p$.

Proof. Since trace commutes with expectation and respects the positive semi-definite ordering, taking trace of both sides of (D.1) yields that

$$\begin{aligned} n &= \mathbf{Tr}(GG^{-1}) = \mathbf{Tr}(\mathbb{E}[P_{A^{1/2}S}G^{-1}]) = \mathbb{E}[\mathbf{Tr}(P_{A^{1/2}S}G^{-1})] = \mathbb{E}[\mathbf{Tr}(P_{A^{1/2}S}G^{-1}P_{A^{1/2}S})] \\ &= \mathbf{Tr}(\mathbb{E}[P_{A^{1/2}S}G^{-1}P_{A^{1/2}S}]) \overset{(D.1)}{\leq} \nu\mathbf{Tr}(\mathbb{E}[P_{A^{1/2}S}]) \\ &= \nu\mathbb{E}[\mathbf{Tr}(P_{A^{1/2}S})] = \nu\mathbb{E}[\mathrm{rank}(A^{1/2}S)] \leq \nu p\,. \end{aligned}$$

□

Next, the upper bound relies on the following lemma, which generalizes Lemma 2 of [20].

Lemma D.2. *Let M be a random matrix. We have that*

$$\mathbb{E}[P_M] \succcurlyeq \mathbb{E}[M](\mathbb{E}[M^\mathsf{T}M])^\dagger\mathbb{E}[M^\mathsf{T}]\,. \tag{D.2}$$

Proof. Our proof follows the strategy in the proof of Theorem 3.2 from [31]. First, write $P_B = B(B^\mathsf{T}B)^\dagger B^\mathsf{T}$. Since $\mathcal{R}(B^\mathsf{T}) = \mathcal{R}(B^\mathsf{T}B)$, we have by generalized Schur complements (see e.g. Theorem 1.20 from [31]) and the fact that expectation preserves the semidefinite order,

$$\begin{bmatrix} B^\mathsf{T}B & B^\mathsf{T} \\ B & P_B \end{bmatrix} \succcurlyeq 0 \Longrightarrow \begin{bmatrix} \mathbb{E}[B^\mathsf{T}B] & \mathbb{E}[B^\mathsf{T}] \\ \mathbb{E}[B] & \mathbb{E}[P_B] \end{bmatrix} \succcurlyeq 0\,.$$

To finish the proof, we need to argue that $\mathcal{R}(\mathbb{E}[B^\mathsf{T}]) \subseteq \mathcal{R}(\mathbb{E}[B^\mathsf{T}B])$, which would allow us to apply the generalized Schur complement again to the right hand side. Fix a $z \in \mathcal{R}(\mathbb{E}[B^\mathsf{T}])$; we can write $z = \mathbb{E}[B^\mathsf{T}]y$ for some y. Now let $q \in \mathrm{Kern}(\mathbb{E}[B^\mathsf{T}B])$. We have that $\mathbb{E}[B^\mathsf{T}B]q = 0$, which implies $0 = q^\mathsf{T}\mathbb{E}[B^\mathsf{T}B]q = \mathbb{E}[\|Bq\|_2^2]$. Therefore, $Bq = 0$ a.s. But this means that $z^\mathsf{T}q = \mathbb{E}[y^\mathsf{T}Bq] = 0$. Hence, $z \in \mathrm{Kern}(\mathbb{E}[B^\mathsf{T}B])^\perp = \mathcal{R}(\mathbb{E}[B^\mathsf{T}B])$. Now applying the generalized Schur complement one more time yields the claim. □

We are now in a position to prove the upper bound of Lemma 4.6. We apply Lemma D.2 to $M = A^{1/2}SS^\mathsf{T}A^{1/2}$ to conclude, using the fact that $\mathcal{R}(M) = \mathcal{R}(MM^\mathsf{T})$, that

$$\mathbb{E}[P_{A^{1/2}S}] = \mathbb{E}[P_{A^{1/2}SS^\mathsf{T}A^{1/2}}] \succcurlyeq \mathbb{E}[A^{1/2}SS^\mathsf{T}A^{1/2}](\mathbb{E}[A^{1/2}SS^\mathsf{T}ASS^\mathsf{T}A^{1/2}])^\dagger\mathbb{E}[A^{1/2}SS^\mathsf{T}A^{1/2}]\,. \tag{D.3}$$

Elementary calculations now yield that for any fixed symmetric matrix $A \in \mathbb{R}^{n\times n}$,

$$\mathbb{E}[SS^\mathsf{T}] = \frac{p}{n}I, \qquad \mathbb{E}[SS^\mathsf{T}ASS^\mathsf{T}] = \frac{p}{n}\left(\frac{p-1}{n-1}A + \left(1 - \frac{p-1}{n-1}\right)\mathrm{diag}(A)\right)\,. \tag{D.4}$$

Hence plugging (D.4) into (D.3),

$$\mathbb{E}[P_{A^{1/2}S}] \succcurlyeq \frac{p}{n}\left(\frac{p-1}{n-1}I + \left(1 - \frac{p-1}{n-1}\right)A^{-1/2}\,\mathrm{diag}(A)A^{-1/2}\right)^{-1}\,. \tag{D.5}$$

We note that the lower bound (2.6) for μ_{rand} presented in Section 2 follows immediately from (D.5).

We next manipulate (4.4) in order to use (D.5). Recall that $G = \mathbb{E}[H]$ and $H = S(S^\mathsf{T}AS)^\dagger S^\mathsf{T}$. From (B.12), we have

$$\lambda_{\max}\left(\mathbb{E}\left[(G^{-1/2}HG^{-1/2})^2\right]\right) \leq \nu \Longleftrightarrow \mathbb{E}\left[HG^{-1}H\right] \preccurlyeq \nu G\,.$$

Next, a simple computation yields

$$\mathbb{E}[HG^{-1}H] = \mathbb{E}[S(S^\mathsf{T}AS)^{-1}S^\mathsf{T}G^{-1}S(S^\mathsf{T}AS)^{-1}S^\mathsf{T}] = A^{-1/2}\mathbb{E}[P_{A^{1/2}S}(\mathbb{E}[P_{A^{1/2}S}])^{-1}P_{A^{1/2}S}]A^{-1/2}\,.$$

Again, since conjugation by $A^{1/2}$ preserves semi-definite ordering, we have that

$$\mathbb{E}[HG^{-1}H] \preccurlyeq \nu G \iff \mathbb{E}[P_{A^{1/2}S}(\mathbb{E}[P_{A^{1/2}S}])^{-1}P_{A^{1/2}S}] \preccurlyeq \nu \mathbb{E}[P_{A^{1/2}S}] \,.$$

Using the fact that for positive definite matrices X, Y we have $X \preccurlyeq Y$ iff $Y^{-1} \preccurlyeq X^{-1}$, (D.5) is equivalent to

$$(\mathbb{E}[P_{A^{1/2}S}])^{-1} \preceq \frac{n}{p}\left(\frac{p-1}{n-1}I + \left(1 - \frac{p-1}{n-1}\right) A^{-1/2}\operatorname{diag}(A)A^{-1/2}\right) \,.$$

Conjugating both sides by $P_{A^{1/2}S}$ and taking expectations,

$$\mathbb{E}[P_{A^{1/2}S}(\mathbb{E}[P_{A^{1/2}S}])^{-1}P_{A^{1/2}S}] \preceq \frac{n}{p}\left(\frac{p-1}{n-1}\mathbb{E}[P_{A^{1/2}S}] + \left(1 - \frac{p-1}{n-1}\right) \mathbb{E}[P_{A^{1/2}S}A^{-1/2}\operatorname{diag}(A)A^{-1/2}P_{A^{1/2}S}]\right) \,. \tag{D.6}$$

Next, letting $J \subseteq 2^{[n]}$ denote the index set associated to S, for every S we have

$$\begin{aligned}
&P_{A^{1/2}S}A^{-1/2}\operatorname{diag}(A)A^{-1/2}P_{A^{1/2}S} \\
&\quad = A^{1/2}S(S^\mathsf{T}AS)^{-1}S^\mathsf{T}A^{1/2}A^{-1/2}\operatorname{diag}(A)A^{-1/2}A^{1/2}S(S^\mathsf{T}AS)^{-1}S^\mathsf{T}A^{1/2} \\
&\quad = A^{1/2}S(S^\mathsf{T}AS)^{-1/2}(S^\mathsf{T}AS)^{-1/2}(S^\mathsf{T}\operatorname{diag}(A)S)(S^\mathsf{T}AS)^{-1/2}(S^\mathsf{T}AS)^{-1/2}S^\mathsf{T}A^{1/2} \\
&\quad \preccurlyeq \lambda_{\max}((S^\mathsf{T}\operatorname{diag}(A)S)(S^\mathsf{T}AS)^{-1})A^{1/2}S(S^\mathsf{T}AS)^{-1}S^\mathsf{T}A^{1/2} \\
&\quad \preccurlyeq \frac{\max_{i\in J} A_{ii}}{\lambda_{\min}(A_J)}P_{A^{1/2}S} \\
&\quad \preccurlyeq \max_{J\in 2^{[n]}:|J|=p} \kappa_{\mathrm{eff},J}(A)P_{A^{1/2}S} \,.
\end{aligned}$$

Plugging this calculation back into (D.6) yields the desired upper bound of Lemma 4.6.