A2BCD: Asynchronous Acceleration with Optimal Complexity

Anonymous authors
Paper under double-blind review

Abstract

In this paper, we propose the Asynchronous Accelerated Nonuniform Randomized Block Coordinate Descent algorithm (A2BCD). We prove A2BCD converges linearly to a solution of the convex minimization problem at the same rate as NU_ACDM, so long as the maximum delay is not too large. This is the first asynchronous Nesterov-accelerated algorithm that attains any provable speedup. Moreover, we then prove that these algorithms both have optimal complexity. Asynchronous algorithms complete much faster iterations, and A2BCD has optimal complexity. Hence we observe in experiments that A2BCD is the top-performing coordinate descent algorithm, converging up to $4 - 5 \times$ faster than NU_ACDM on some data sets in terms of wall-clock time. To motivate our theory and proof techniques, we also derive and analyze a continuous-time analogue of our algorithm and prove it converges at the same rate.

1 Introduction

In this paper, we propose and prove the convergence of the Asynchronous Accelerated Nonuniform Randomized Block Coordinate Descent algorithm (A2BCD), the first asynchronous Nesterov-accelerated algorithm that achieves optimal complexity. No previous attempts have been able to prove a speedup for asynchronous Nesterov acceleration. We aim to find the minimizer $x^*$ of the unconstrained minimization problem:

$$\min_{x \in \mathbb{R}^d} f(x) = f(x(1), \ldots, x(n)) \quad (1.1)$$

where $f$ is $\sigma$-strongly convex for $\sigma > 0$ with $L$-Lipschitz gradient $\nabla f = (\nabla_1 f, \ldots, \nabla_n f)$. $x \in \mathbb{R}^d$ is composed of coordinate blocks $x(1), \ldots, x(n)$. The coordinate blocks of the gradient $\nabla_i f$ are assumed $L_i$-Lipschitz with respect to the $i$th block. That is, $\forall x, h \in \mathbb{R}^d$:

$$\|\nabla_i f(x + P_i h) - \nabla_i f(x)\| \leq L_i \|h\| \quad (1.2)$$

where $P_i$ is the projection onto the $i$th block of $\mathbb{R}^d$. Let $\bar{L} \triangleq \frac{1}{n} \sum_{i=1}^n L_i$ be the average block Lipschitz constant. These conditions on $f$ are assumed throughout this whole paper. Our algorithm can also be applied to non-strongly convex objectives ($\sigma = 0$) or non-smooth objectives using the black box reduction techniques proposed in Allen-Zhu & Hazan (2016). Hence we consider only the coordinate smooth, strongly-convex case. Our algorithm can also be applied to the convex regularized ERM problem via the standard dual transformation (see for instance Lin et al. (2014)):

$$f(x) = \frac{1}{n} \sum_{i=1}^n f_i(\langle a_i, x \rangle) + \frac{\lambda}{2} \|x\|^2 \quad (1.3)$$
Hence \texttt{A2BCD} can be used as an asynchronous Nesterov-accelerated finite-sum algorithm.

Coordinate descent methods, in which a chosen coordinate block $i_k$ is updated at every iteration, are a popular way to solve equation 1.1. Randomized block coordinate descent (\texttt{RBCD}, Nesterov (2012)) updates a uniformly randomly chosen coordinate block $i_k$ with a gradient-descent-like step:

$$x_{k+1} = x_k - \frac{1}{L_i} \nabla i_k f(x_k).$$

The complexity $K(\epsilon)$ of an algorithm is defined as the number of iterations required to decrease the error $E(f(x_k) - f(x_0))$ to less than $\epsilon(f(x_0) - f(x_0))$. Randomized coordinate descent has a complexity of $K(\epsilon) = \mathcal{O}(n(\bar{L}/\sigma) \ln(1/\epsilon))$.

Using a series of averaging and extrapolation steps, \textit{accelerated} \texttt{RBCD} Nesterov (2012) improves \texttt{RBCD}'s iteration complexity $K(\epsilon)$ to $\mathcal{O}(n\sqrt{\bar{L}/\sigma} \ln(1/\epsilon))$, which leads to much faster convergence when $\frac{\bar{L}}{\sigma}$ is large. This rate is optimal when all $L_i$ are equal Lan & Zhou (2015). Finally, using a special probability distribution for the random block index $i_k$, the non-uniform accelerated coordinate descent method Allen-Zhu et al. (2015) (\texttt{NU_ACDM}) can further decrease the complexity to $\mathcal{O}(\sum_{i=1}^{n} \sqrt{L_i/\sigma} \ln(1/\epsilon))$, which can be up to $\sqrt{n}$ times faster than accelerated \texttt{RBCD}, since some $L_i$ can be significantly smaller than $\bar{L}$. \texttt{NU_ACDM} is the current state-of-the-art coordinate descent algorithm for solving equation 1.1.

Our \texttt{A2BCD} algorithm generalizes \texttt{NU_ACDM} to the asynchronous-parallel case. We solve equation 1.1 with a collection of $p$ computing nodes that continually read a shared-access solution vector $y$ into local memory then compute a block gradient $\nabla_i f$, which is used to update shared solution vectors $(x, y, v)$. Proving convergence in the asynchronous case requires extensive new technical machinery.

A traditional synchronous-parallel implementation is organized into rounds of computation: Every computing node must complete an update in order for the next iteration to begin. However, this synchronization process can be extremely costly, since the lateness of a single node can halt the entire system. This becomes increasingly problematic with scale, as differences in node computing speeds, load balancing, random network delays, and bandwidth constraints mean that a synchronous-parallel solver may spend more time waiting than computing a solution.

Computing nodes in an asynchronous solver do not wait for others to complete and share their updates before starting the next iteration. They simply continue to update the solution vectors with the most recent information available, without any central coordination. This eliminates costly idle time, meaning that asynchronous algorithms can be much faster than traditional ones, since they have much faster iterations. For instance, random network delays cause asynchronous algorithms to complete iterations $\Omega(\ln(p))$ time faster than synchronous algorithms at scale. This and other factors that influence the speed of iterations are discussed in Hannah & Yin (2017a). However, since many iterations may occur between the time that a node reads the solution vector, and the time that its computed update is applied, effectively the solution vector is being updated with outdated information. At iteration $k$, the block gradient $\nabla_{i_k} f$ is computed at a \textit{delayed iterate} $\hat{y}_k$ defined as:

$$\hat{y}_k = \left( (y_{k-j(k,1)})_1, \ldots, (y_{k-j(k,n)})_n \right)$$

for delay parameters $j(k, 1), \ldots, j(k, n) \in \mathbb{N}$. Here $j(k, i)$ denotes how many iterations out of date coordinate block $i$ is at iteration $k$. Different blocks may be out of date by different amounts, which is known as an \textit{inconsistent read}. We assume\footnote{\textit{Every coordinate can be outdated by a different amount without significantly changing the proofs.}} that $j(k, i) \leq \tau$ for some constant $\tau < \infty$.

Asynchronous algorithms were proposed in Chazan & Miranker (1969) to solve linear systems. General convergence results and theory were developed later in Bertsekas (1983); Bertsekas &
1.1 Summary of Contributions

Tsitsiklis (1997); Tseng et al. (1990); Luo & Tseng (1992; 1993); Tseng (1991) for partially and totally asynchronous systems, with essentially-cyclic block sequence $i_k$. More recently, there has been renewed interest in asynchronous algorithms with random block coordinate updates. Linear and sublinear convergence results were proven for asynchronous RBCD Liu & Wright (2015); Liu et al. (2014); Avron et al. (2014), and similar was proven for asynchronous SGD Recht et al. (2011), and variance reduction algorithms Reddi et al. (2015); Leblond et al. (2017); Mania et al. (2015); Huo & Huang (2016), and primal-dual algorithms Combettes & Eckstein (2018).

There is also a rich body of work on asynchronous SGD. In the distributed setting, Zhou et al. (2018) showed global convergence for stochastic variationally coherent problems even when the delays grow at a polynomial rate. In Lian et al. (2018), an asynchronous decentralized SGD was proposed with the same optimal sublinear convergence rate as SGD and linear speedup with respect to the number of workers. In Liu et al. (2018), authors obtained an asymptotic rate of convergence for asynchronous momentum SGD on streaming PCA, which provides insight into the tradeoff between asynchrony and momentum. In Dutta et al. (2018), authors prove convergence results for asynchronous SGD that highlight the tradeoff between faster iterations and iteration complexity. Further related work is discussed in Section 4.

1.1 Summary of Contributions

In this paper, we prove that $A2BCD$ attains $NU_{ACDM}$'s state-of-the-art iteration complexity to highest order for solving equation 1.1, so long as delays are not too large (see Section 2). The proof is very different from that of Allen-Zhu et al. (2015), and involves significant technical innovations and complexity related to the analysis of asynchronicity.

We also prove that $A2BCD$ (and hence $NU_{ACDM}$) has optimal complexity to within a constant factor over a fairly general class of randomized block coordinate descent algorithms (see Section 2.1). This extends results in Lan & Zhou (2015) to asynchronous algorithms with $L_i$ not all equal. Since asynchronous algorithms complete faster iterations, and $A2BCD$ has optimal complexity, we expect $A2BCD$ to be faster than all existing coordinate descent algorithms. We confirm with numerical experiments that $A2BCD$ is the current fastest coordinate descent algorithm (see Section 5).

We are only aware of one previous and one contemporaneous attempt at proving convergence results for asynchronous Nesterov-accelerated algorithms. However, the first is not accelerated and relies on extreme assumptions, and the second obtains no speedup. Therefore, we claim that our results are the first-ever analysis of asynchronous Nesterov-accelerated algorithms that attains a speedup. Moreover, our speedup is optimal for delays not too large$^4$.

The work of Meng et al. claims to obtain square-root speedup for an asynchronous accelerated SVRG. In the case where all component functions have the same Lipschitz constant $L$, the complexity they obtain reduces to $(n + \kappa) \ln(1/\epsilon) \text{ for } \kappa = \mathcal{O}(\tau n^2)$ (Corollary 4.4). Hence authors do not even obtain accelerated rates. Their convergence condition is $\tau < \frac{1}{\Delta^2 \sqrt{\pi}}$ for sparsity parameter $\Delta$. Since the dimension $d$ satisfies $d \geq \frac{1}{\Delta}$, they require $d \geq 2^{16} \tau^8$. So $\tau = 20$ requires dimension $d > 10^{15}$.

In a contemporaneous preprint, authors in Fang et al. (2018) skillfully devised accelerated schemes for asynchronous coordinate descent and SVRG using momentum compensation techniques. Although their complexity results have the improved $\sqrt{\kappa}$ dependence on the condition number, they do not prove any speedup. Their complexity is $\tau$ times larger than the serial complexity. Since $\tau$ is necessarily greater than $p$, their results imply that adding more computing nodes will increase running time. The authors claim that they can extend their results to linear speedup for asynchronous,

$^4$Speedup is defined precisely in Section 2
accelerated SVRG under sparsity assumptions. And while we think this is quite likely, they have not yet provided proof.

We also derive a second-order ordinary differential equation (ODE), which is the continuous-time limit of A2BCD (see Section 3). This extends the ODE found in Su et al. (2014) to an asynchronous accelerated algorithm minimizing a strongly convex function. We prove this ODE linearly converges to a solution with the same rate as A2BCD’s, without needing to resort to the restarting techniques. The ODE analysis motivates and clarifies the our proof strategy of the main result.

2 Main results

We should consider functions $f$ where it is efficient to calculate blocks of the gradient, so that coordinate-wise parallelization is efficient. That is, the function should be “coordinate friendly” Peng et al. (2016b). This is a very wide class that includes regularized linear regression, logistic regression, etc. The $L^2$-regularized empirical risk minimization problem is not coordinate friendly in general, however the equivalent dual problem is, and hence can be solved efficiently by A2BCD (see Lin et al. (2014), and Section 5).

To calculate the $k+1$th iteration of the algorithm from iteration $k$, we use only one block of the gradient $\nabla f_k$. We assume that the delays $j(k,i)$ are independent of the block sequence $i_k$, but otherwise arbitrary (This is a standard assumption found in the vast majority of papers, but can be relaxed Sun et al. (2017); Leblond et al. (2017); Cannelli et al. (2017)).

Definition 1. Asynchronous Accelerated Randomized Block Coordinate Descent (A2BCD). Let $f$ be $\sigma$-strongly convex, and let its gradient $\nabla f$ be $L$-Lipschitz with block coordinate Lipschitz parameters $L_i$ as in equation 1.2. We define the condition number $\kappa = L/\sigma$, and let $\bar{L} = \min_i L_i$. Using these parameters, we sample $i_k$ in an independent and identically distributed (IID) fashion according to

$$
\Pr[i_k = j] = \frac{L_j^{1/2}}{S}, \quad j \in \{1, \ldots, n\}, \quad \text{for } S = \sum_{i=1}^n L_i^{1/2}.
$$

(2.1)

Let $\tau$ be the maximum asynchronicity delay. We define the dimensionless \textbf{asynchronicity parameter} $\psi$, which is proportional to $\tau$, and quantifies how strongly asynchronicity will affect convergence:

$$
\psi = 9 \left( S^{-1/2} L_1^{-1/2} L_3^{3/4} \kappa^{1/4} \right) \times \tau
$$

(2.2)

We use the above system parameters and $\psi$ to define the coefficients $\alpha, \beta$, and $\gamma$ via eqs. (2.3) to (2.5). Hence A2BCD algorithm is defined via the iterations: eqs. (2.6) to (2.8).

$$
\alpha \triangleq \left( 1 + (1 + \psi)\sigma^{-1/2}S \right)^{-1}
$$

(2.3)

$$
y_k = \alpha x_k + (1 - \alpha)x_k,
$$

(2.6)

$$
\beta \triangleq 1 - (1 - \psi)\sigma^{-1/2}S^{-1}
$$

(2.4)

$$
x_{k+1} = y_k - hL_{i_k}^{-1} \nabla_{i_k} f(\hat{y}_k),
$$

(2.7)

$$
h \triangleq 1 - \frac{1}{2} \sigma^{1/2} L_1^{-1/2} \psi.
$$

(2.5)

$$
v_{k+1} = \beta v_k + (1 - \beta)y_k - \sigma^{-1/2}L_{i_k}^{-1/2} \nabla_{i_k} f(\hat{y}_k).
$$

(2.8)

See Section A for a discussion of why it is practical and natural to have the gradient $\nabla_{i_k} f(\hat{y}_k)$ to be outdated, while the actual variables $x_k, y_k, v_k$ can be efficiently kept up to date. Essentially it is because most of the computation lies in computing $\nabla_{i_k} f(\hat{y}_k)$. After this is computed, $x_k, y_k, v_k$ can be updated more-or-less atomically with minimal overhead, meaning that they will always be up to date. However our main results still hold for more general asynchronicity.

A natural quantity to consider in asynchronous convergence analysis is the \textbf{asynchronicity error}, a powerful tool for analyzing asynchronous algorithms used in several recent works Peng et al. (2016a); Hannah & Yin (2017b); Sun et al. (2017); Hannah & Yin (2017a). We adapt it and use a weighted sum of the history of the algorithm with decreasing weight as you go further back in time.
Definition 2. Asynchronicity error. Using the above parameters, we define:

\[ A_k = \sum_{j=1}^{\tau} c_j \|y_{k+1-j} - y_{k-j}\|^2 \]  

for \( c_i = \frac{6}{S} \) \( L^{1/2} \kappa^{3/2} \tau \sum_{j=i}^{\tau} \left(1 - \sigma^{1/2} S^{-1}\right)^{i-j-1} \psi^{-1}. \)  

(2.9) (2.10)

Here we define \( y_k = y_0 \) for all \( k < 0 \). The determination of the coefficients \( c_i \) is in general a very involved process of trial and error, intuition, and balancing competing requirements. The algorithm doesn’t depend on the coefficients, however; they are only an analytical tool.

We define \( E_k[X] \) as the expectation of \( X \) conditioned on \( (x_0, \ldots, x_k), (y_0, \ldots, y_k), (v_0, \ldots, v_k) \), and \( (i_0, \ldots, i_{k-1}) \). To simplify notation\(^4\), we assume that the minimizer \( x_* \) is 0, and that \( f(x_*) \) = 0 with no loss in generality. We define the Lyapunov function:

\[ \rho_k = \|v_k\|^2 + A_k + cf(x_k) \]

(2.11) for \( c = 2\sigma^{-1/2} S^{-1} (\beta \alpha^{-1}(1 - \alpha) + 1). \)  

(2.12)

We now present this paper’s first main contribution.

Theorem 1. Let \( f \) be \( \sigma \)-strongly convex with a gradient \( \nabla f \) that is \( L \)-Lipschitz with block Lipschitz constants \( \{L_i\}_{i=1}^{n} \). Let \( \psi \) defined in equation 2.2 satisfy \( \psi \leq \frac{3}{4} \) (i.e. \( \tau \leq \frac{1}{2\tau} S^{1/2} L^{1/2} L^{-3/4} \kappa^{-1/4} \)). Then for \( A2BCD \) we have:

\[ E_k[\rho_{k+1}] \leq \left(1 - (1 - \psi)\sigma^{1/2} S^{-1}\right)\rho_k. \]

To obtain \( E[\rho_k] \leq \epsilon \rho_0 \), it takes \( K_{A2BCD}(\epsilon) \) iterations for:

\[ K_{A2BCD}(\epsilon) = \left(\sigma^{-1/2} S + O(1)\right) \frac{\ln(1/\epsilon)}{1 - \psi}, \]

(2.13)

where \( O(\cdot) \) is asymptotic with respect to \( \sigma^{-1/2} S \rightarrow \infty \), and uniformly bounded.

This result is proven in Section B. A stronger result for \( L_i \equiv L \) can be proven, but this adds to the complexity of the proof; see Section E for a discussion. In practice, asynchronous algorithms are far more resilient to delays than the theory predicts. \( \tau \) can be much larger without negatively affecting the convergence rate and complexity. This is perhaps because we are limited to a worst-case analysis, which is not representative of the average-case performance.

Allen-Zhu et al. (2015) (Theorem 5.1) shows a linear convergence rate of \( 1 - 2/(1 + 2\sigma^{-1/2} S) \) for \( NU_{ACDM} \), which leads to the corresponding iteration complexity of \( K_{NU_{ACDM}}(\epsilon) = (\sigma^{-1/2} S + O(1)) \ln(1/\epsilon) \). Hence, we have:

\[ K_{A2BCD}(\epsilon) = \frac{1}{1 - \psi} (1 + o(1)) K_{NU_{ACDM}}(\epsilon) \]

When \( 0 \leq \psi \ll 1 \), or equivalently, when \( \tau \ll S^{1/2} L^{1/2} L^{-3/4} \kappa^{-1/4} \), the complexity of \( A2BCD \) asymptotically matches that of \( NU_{ACDM} \). Hence \( A2BCD \) combines state-of-the-art complexity with the faster iterations and superior scaling that asynchronous iterations allow. We now present some special cases of the conditions on the maximum delay \( \tau \) required for good complexity.

Corollary 3. Let the conditions of Theorem 1 hold. If all coordinate-wise Lipschitz constants \( L_i \) are equal (i.e. \( L_i = L_1 \), \( \forall i \)), then we have \( K_{A2BCD}(\epsilon) \sim K_{NU_{ACDM}}(\epsilon) \) when \( \tau \ll n^{1/2}\kappa^{-1/4}(L_1/L)^{3/4} \). If we further assume all coordinate-wise Lipschitz constants \( L_i \) equal \( L \). Then \( K_{A2BCD}(\epsilon) \sim K_{NU_{ACDM}}(\epsilon) = K_{ACDM}(\epsilon) \), when \( \tau \ll n^{1/2}\kappa^{-1/4} \).

\(^4\)We can assume \( x_* = 0 \) with no loss in generality since we may translate the coordinate system so that \( x_* \) is at the origin. We can assume \( f(x_*) = 0 \) with no loss in generality, since we can replace \( f(x) \) with \( f(x) - f(x_*) \). Without this assumption, the Lyapunov function simply becomes: \( \|v_k - x_*\|^2 + A_k + c(f(x_k) - f(x_*)). \)
Remark 1. Reduction to synchronous case. Notice that when $\tau = 0$, we have $\psi = 0$, $c_i \equiv 0$ and hence $A_k \equiv 0$. Thus $\texttt{A2BCD}$ becomes equivalent to $\texttt{NU_ACDM}$, the Lyapunov function $^5 \rho_k$ becomes equivalent to one found in Allen-Zhu et al. (2015)(pg. 9), and Theorem 1 yields the same complexity.

The maximum delay $\tau$ will be a function $\tau(p)$ of $p$, number of computing nodes. Clearly $\tau \geq p$, and experimentally it has been observed that $\tau = O(p)$ Leblond et al. (2017). Let gradient complexity $K(\epsilon, \tau)$ be the number of gradients required for an asynchronous algorithm with maximum delay $\tau$ to attain suboptimality $\epsilon$. $\tau(1) = 0$, since with only 1 computing node there can be no delay. This corresponds to the serial complexity. We say that an asynchronous algorithm attains a complexity speedup if $\frac{pK(\epsilon, \tau(0))}{K(\epsilon, \tau(p))}$ is increasing in $p$. We say it attains linear complexity speedup if $\frac{pK(\epsilon, \tau(0))}{K(\epsilon, \tau(p))} = \Omega(p)$.

In Theorem 1, we obtain a linear complexity speedup (for $p$ not too large), whereas no other prior attempt can attain even a complexity speedup with Nesterov acceleration.

In the ideal scenario where the rate at which gradients are calculated increases linearly with $p$, algorithms that have linear complexity speedup will have a linear decrease in wall-clock time. However in practice, when the number of computing nodes is sufficiently large, the rate at which gradients are calculated will no longer be linear. This is due to many parallel overhead factors including too many nodes sharing the same memory read/write bandwidth, and network bandwidth. However we note that even with these issues, we obtain much faster convergence than the synchronous counterpart experimentally.

2.1 Optimality

$\texttt{NU_ACDM}$ and hence $\texttt{A2BCD}$ are in fact optimal in some sense. That is, among a fairly wide class of coordinate descent algorithms $A$, they have the best-possible worst-case complexity to highest order. We extend the work in Lan & Zhou (2015) to encompass algorithms are asynchronous and have unequal $L_i$. For a subset $S \subseteq \mathbb{R}^d$, we let $IC(S)$ (inconsistent read) denote the set of vectors $v$ whose components are a combination of components of vectors in the set $S$. That is, $v = (v_1, v_2, \ldots, v_d)$ for some vectors $v_1, v_2, \ldots, v_d \in S$. Here $v_{i,j}$ denotes the $j$th component of vector $v_i$.

Definition 4. Asynchronous Randomized Incremental Algorithms. Consider the unconstrained minimization problem equation 1.1 for function $f$ satisfying the conditions stated in Section 1. We define the class $A$ as algorithms $G$ on this problem such that:

1. For each parameter set $(\sigma, L_1, \ldots, L_n, \epsilon, \tau(0))$, $G$ has an associated IID random variable $i_k$ with some fixed distribution $\mathbb{P}[i_k] = p_k = 1$. For the subset of $S \subseteq \mathbb{R}^d$, we let $IC(S)$ (inconsistent read) denote the set of vectors $v$ whose components are a combination of components of vectors in the set $S$. That is, $v = (v_1, v_2, \ldots, v_d)$ for some vectors $v_1, v_2, \ldots, v_d \in S$. Here $v_{i,j}$ denotes the $j$th component of vector $v_i$.

2. The iterates of $A$ satisfy: $x_{k+1} \in \text{span}(IC(X_k), \nabla_{i_0} f(IC(X_0)), \nabla_{i_1} f(IC(X_1)), \ldots, \nabla_{i_{n-1}} f(IC(X_{n-1})))$

This is a rather general class: $x_{k+1}$ can be constructed from any inconsistent reading of past iterates $IC(X_k)$, and any past gradient of an inconsistent read $\nabla_{i_0} f(IC(X_j))$.

Theorem 2. For any algorithm $G \in A$ that solves eq. 1.1, and parameter set $(\sigma, L_1, \ldots, L_n, \epsilon, \tau(0))$, there is a dimension $d$, a corresponding function $f$ on $\mathbb{R}^d$, and a starting point $x_0$, such that

$$\mathbb{E}||x_k - x_*||^2/||x_0 - x_*||^2 \geq \frac{1}{2}(1 - 4/(\sum_{j=1}^{n} \sqrt{L_j/\sigma} + 2n))^k$$

Hence $A$ has a complexity lower bound: $K(\epsilon) \geq \frac{1}{2}(1 + o(1))(\sum_{j=1}^{n} \sqrt{L_j/\sigma} + 2n) \ln(1/2\epsilon)$

Our proof in Section D follows very similar lines to Lan & Zhou (2015); Nesterov (2013).

^5Their Lyapunov function is in fact a generalization of the one found in Nesterov (2012).
3 ODE Analysis

In this section we present and analyze an ODE which is the continuous-time limit of \( A2BCD \). This ODE is a strongly convex, and asynchronous version of the ODE found in Su et al. (2014). For simplicity, assume \( L_i = L, \forall i \). We rescale (i.e. we replace \( f(x) \) with \( \frac{1}{\tau} f(x) \)) \( f \) so that \( \sigma = 1 \), and hence \( \kappa = L/\sigma = L \). Taking the discrete limit of synchronous \( A2BCD \) (i.e. accelerated \( RBCD \)), we can derive the following ODE\(^6\) (see Section equation C.1):

\[
\dot{Y} + 2n^{-1}\kappa^{-1/2} \dot{Y} + 2n^{-2}\kappa^{-1}\nabla f(Y) = 0
\]

(3.1)

We define the parameter \( \eta \triangleq n\kappa^{1/2} \), and the energy: \( E(t) = e^{n^{-1}\kappa^{-1/2}t}(f(Y) + \frac{1}{2}\|Y + \eta \dot{Y}\|^2) \). This is very similar to the Lyapunov function discussed in equation 2.11, with \( \frac{1}{2}\|Y(t) + \eta \dot{Y}(t)\|^2 \) fulfilling the role of \( \|v_k\|^2 \), and \( A_k = 0 \) (since there is no delay yet). Much like the traditional analysis in the proof of Theorem 1, we can derive a linear convergence result with a similar rate. See Section C.2.

Lemma 5. If \( Y \) satisfies equation 3.1, the energy satisfies \( E'(t) \leq 0, E(t) \leq E(0) \), and hence:

\[
f(Y(t)) + \frac{1}{4}\|Y(t) + n\kappa^{1/2} \dot{Y}(t)\|^2 \leq \left(f(Y(0)) + \frac{1}{4}\|Y(0) + \eta \dot{Y}(0)\|^2\right)e^{-n^{-1}\kappa^{-1/2}t}
\]

We may also analyze an asynchronous version of equation 3.1 to motivate the proof of our main theorem. Here \( \dot{Y}(t) \) is a delayed version of \( Y(t) \) with the delay bounded by \( \tau \).

\[
\dot{Y} + 2n^{-1}\kappa^{-1/2} \dot{Y} + 2n^{-2}\kappa^{-1}\nabla f(\dot{Y}) = 0,
\]

(3.2)

Unfortunately, this energy satisfies (see Section equation C.4, equation C.7):

\[
e^{-\eta^{-1}t} E'(t) \leq -\frac{1}{8}\eta \|\dot{Y}\|^2 + 3\kappa^2 \eta^{-1}\tau D(t), \text{ for } D(t) \triangleq \int_{t-\tau}^{t} \|\dot{Y}(s)\|^2 ds.
\]

Hence this energy \( E(t) \) may not be decreasing in general. But, we may add a continuous-time asynchronicity error (see Sun et al. (2017)), much like in Definition 2, to create a decreasing energy. Let \( c_0 \geq 0 \) and \( \tau > 0 \) be arbitrary constants that will be set later. Define:

\[
A(t) = \int_{t-\tau}^{t} c(t-s)\|\dot{Y}(s)\|^2 ds, \text{ for } c(t) \triangleq c_0\left(e^{-\tau t} + \frac{e^{-\tau t}}{1-e^{-\tau t}}(e^{-\tau t} - 1)\right).
\]

Lemma 6. When \( \tau r \leq \frac{1}{2} \), the asynchronicity error \( A(t) \) satisfies:

\[
e^{-\eta^{-1}t} \frac{d}{dt}(e^{\tau t}A(t)) \leq c_0\|\dot{Y}(t)\|^2 - \frac{1}{2}\tau^{-1}c_0D(t).
\]

See Section C.3 for the proof. Adding this error to the Lyapunov function serves a similar purpose in the continuous-time case as in the proof of Theorem 1 (see Lemma 11). It allows us to negate \( \frac{1}{2}\tau^{-1}c_0 \) units of \( D(t) \) for the cost of creating \( c_0 \) units of \( \|\dot{Y}(t)\|^2 \). This restores monotonicity.

Theorem 3. Let \( c_0 = 6\kappa^2\eta^{-1}\tau^2 \), and \( r = \eta^{-1} \). If \( \tau \leq \frac{1}{\sqrt{48}}n\kappa^{-1/2} \) then we have:

\[
e^{-\eta^{-1}t} \frac{d}{dt}\left(E(t) + e^{\tau^{-1}t}A(t)\right) \leq 0.
\]

(3.3)

Hence \( f(Y(t)) \) convergence linearly to \( f(x_*) \) with rate \( \mathcal{O}\left(\exp\left(-t/(n\kappa^{1/2})\right)\right) \)

---

\(^6\)For compactness, we have omitted the \( t \) from time-varying functions \( Y(t), \dot{Y}(t), \nabla Y(t) \), etc.
Notice how this convergence condition is similar to Corollary 3, but a little looser. The convergence condition in Theorem 1 can actually be improved to approximately match this (see Section E).

**Proof.**

\[
e^{-\eta t} \frac{d}{dt} \left( E(t) + e^{\eta t} A(t) \right) \leq \left( c_0 - \frac{1}{8} \eta \right) \| Y \|^2 + \left( 3\kappa^2 \eta^{-1} \tau - \frac{1}{2} \tau^{-1} c_0 \right) D(t)
\]

\[
= 6 \eta^{-1} \kappa^2 \left( \tau^2 - \frac{1}{48} \tau^2 \kappa^{-1} \right) \| Y \|^2 \leq 0 \tag{5.1}
\]

The preceding should hopefully elucidate the logic and general strategy of the proof of Theorem 1.

4 Related work

We now discuss related work that was not addressed in Section 1. Nesterov acceleration is a method for improving an algorithm’s iteration complexity’s dependence the condition number $\kappa$. Nesterov-accelerated methods have been proposed and discovered in many settings Nesterov (1983); Tseng (2008); Nesterov (2012); Lin et al. (2014); Lu & Xiao (2014); Shalev-Shwartz & Zhang (2016); Allen-Zhu (2017), including for coordinate descent algorithms (algorithms that use 1 gradient block $\nabla_i f$ or minimize with respect to 1 coordinate block per iteration), and incremental algorithms (algorithms for finite sum problems $\frac{1}{n} \sum_{i=1}^{n} f_i(x)$ that use 1 function gradient $\nabla f_i(x)$ per iteration). Such algorithms can often be augmented to solve composite minimization problems (minimization for objective of the form $f(x) + g(x)$, especially for nonsmooth $g$), or include constraints.

In Peng et al. (2016a), authors proposed and analyzed an asynchronous fixed-point algorithm called ARock, that takes proximal algorithms, forward-backward, ADMM, etc. as special cases. Work has also been done on asynchronous algorithms for finite sums in the operator setting Davis (2016); Johnstone & Eckstein (2018). In Hannah & Yin (2017b); Sun et al. (2017); Peng et al. (2016c); Cannelli et al. (2017) showed that many of the assumptions used in prior work (such as bounded delay $\tau < \infty$) were unrealistic and unnecessary in general. In Hannah & Yin (2017a) the authors showed that asynchronous iterations will complete far more iterations per second, and that a wide class of asynchronous algorithms, including asynchronous RBCD, have the same iteration complexity as their synchronous counterparts. Hence certain asynchronous algorithms can be expected to significantly outperform traditional ones.

In Xiao et al. (2017) authors propose a novel asynchronous catalyst-accelerated Lin et al. (2015) primal-dual algorithmic framework to solve regularized ERM problems. They structure the parallel updates so that the data that an update depends on is up to date (though the rest of the data may not be). However catalyst acceleration incurs a $\log(\kappa)$ penalty over Nesterov acceleration in general. In Allen-Zhu (2017), the author argues that the inner iterations of catalyst acceleration are hard to tune, making it less practical than Nesterov acceleration.

5 Numerical experiments

To investigate the performance of $A2BCD$, we solve the ridge regression problem. Consider the following primal and corresponding dual objective (see for instance Lin et al. (2014)):

\[
\min_{w \in \mathbb{R}^d} P(w) = \frac{1}{2n} \| A^T w - l \|^2 + \frac{\lambda}{2} \| w \|^2, \quad \min_{\alpha \in \mathbb{R}^n} D(\alpha) = \frac{1}{2d^2} \| A^T \| \alpha \|^2 + \frac{1}{2d} \| \alpha + l \|^2 \tag{5.1}
\]

where $A \in \mathbb{R}^{d \times n}$ is a matrix of $n$ samples and $d$ features, and $l$ is a label vector. We let $A = [A_1, \ldots, A_m]$ where $A_i$ are the column blocks of $A$. We compare $A2BCD$ (which is asynchronous accelerated), synchronous NU_ACDM (which is synchronous accelerated), and asynchronous RBCD
(which is asynchronous non-accelerated). Nodes randomly select a coordinate block according to equation 2.1, calculate the corresponding block gradient, and use it to apply an update to the shared solution vectors. synchronous NU_ACDM is implemented in a batch fashion, with batch size $p$ (1 block per computing node). Nodes in synchronous NU_ACDM implementation must wait until all nodes apply their computed gradients before they can start the next iteration, but the asynchronous algorithms simply compute with the most up-to-date information available.

We use the datasets $w1a$ (47272 samples, 300 features), $wxa$ which combines the data from from $w1a$ to $w8a$ (293201 samples, 300 features), and $alo1$ (108000 samples, 128 features) from LIBSVM Chang & Lin (2011). The algorithm is implemented in a multi-threaded fashion using C++11 and GNU Scientific Library with a shared memory architecture. We use 40 threads on two 2.5GHz 10-core Intel Xeon E5-2670v2 processors. See Section A.1 for a discussion of parameter tuning and estimation. The parameters for each algorithm are tuned to give the fastest performance, so that a fair comparison is possible.

A critical ingredient in the efficient implementation of A2BCD and NU_ACDM for this problem is the efficient update scheme discussed in Lee & Sidford (2013b;a). In linear regression applications such as this, it is essential to be able to efficiently maintain or recover $Ay$. This is because calculating block gradients requires the vector $A^T_i Ay$, and without an efficient way to recover $Ay$, block gradient evaluations are essentially 50% as expensive as full-gradient calculations. Unfortunately, every accelerated iteration results in dense updates to $y_k$ because of the averaging step in equation 2.6. Hence $Ay$ must be recalculated from scratch.

However Lee & Sidford (2013a) introduces a linear transformation that allows for an equivalent iteration that results in sparse updates to new iteration variables $p$ and $q$. The original purpose of this transformation was to ensure that the averaging steps (e.g. equation 2.6) do not dominate the computational cost for sparse problems. However we find a more important secondary use which applies to both sparse and dense problems. Since the updates to $p$ and $q$ are sparse coordinate-block updates, the vectors $Ap$ and $Aq$ can be efficiently maintained, and therefore block gradients can be efficiently calculated. The specifics of this efficient implementation are discussed in Section A.2.

In Table 5, we plot the sub-optimality vs. time for decreasing values of $\lambda$, which corresponds to increasingly large condition numbers $\kappa$. When $\kappa$ is small, acceleration doesn’t result in a significantly better convergence rate, and hence A2BCD and async-RBCD both outperform sync-NU_ACDM since they complete faster iterations at similar complexity. Acceleration for low $\kappa$ has unnecessary overhead, which means async-RBCD can be quite competitive. When $\kappa$ becomes large, async-RBCD is no longer competitive, since it has a poor convergence rate. We observe that A2BCD and sync-NU_ACDM have essentially the same convergence rate, but A2BCD is up to 4 – 5× faster than sync-NU_ACDM because it completes much faster iterations. We observe this advantage despite the fact that we are in an ideal environment for synchronous computation: A small, homogeneous, high-bandwidth, low-latency cluster. In large-scale heterogeneous systems with greater synchronization overhead, bandwidth constraints, and latency, we expect A2BCD’s advantage to be much larger.
Table 1: Sub-optimality $f(y_k) - f(x_*)$ (y-axis) vs time in seconds (x-axis) for A2BCD, synchronous NU_ACDM, and asynchronous RBCD for data sets w1a, wxa and aloi for various values of $\lambda$.

REFERENCES


A Efficient Implementation

An efficient implementation will have coordinate blocks of size greater than 1. This to ensure the efficiency of linear algebra subroutines. Especially because of this, the bulk of the computation for each iteration is computing $\nabla_i f(\hat{y}_k)$, and not the averaging steps. Hence the computing nodes only need a local copy of $y_k$ in order to do the bulk of an iteration’s computation. Given this gradient $\nabla_i f(\hat{y}_k)$, updating $y_k$ and $v_k$ is extremely fast ($x_k$ can simply be eliminated). It is natural to simply store $y_k$ and $v_k$ centrally, and update them when the delayed gradients $\nabla_i f(\hat{y}_k)$. Given the above, a write mutex over $(y, v)$ has minuscule overhead (which we confirm with experiments), and makes the labeling of iterates unambiguous. This also ensures that $v_k$ and $y_k$ are always up to date when $(y, v)$ are being updated. Whereas the gradient $\nabla_i f(\hat{y}_k)$ may at the same time be out of date, since it has been calculated with an outdated version of $y_k$. However a write mutex is not necessary in practice, and does not appear to affect convergence rates or computation time. Also it is possible to prove convergence under more general asynchronicity.

A.1 Parameter selection and tuning

When defining the coefficients, $\sigma$ may be underestimated, and $L, L_1, \ldots, L_n$ may be overestimated if exact values are unavailable. Notice that $x_k$ can be eliminated from the above iteration, and the block gradient $\nabla_i f(\hat{y}_k)$ only needs to be calculated once per iteration. A larger (or overestimated) maximum delay $\tau$ will cause a larger asynchronicity parameter $\psi$, which leads to more conservative step sizes to compensate.

To estimate $\psi$, one can first performed a dry run with all coefficient set to 0 to estimate $\tau$. All function parameters can be calculated exactly for this problem in terms of the data matrix and $\lambda$. We can then use these parameters and this tau to calculate $\psi$. $\psi$ and $\tau$ merely change the parameters, and do not change execution patterns of the processors. Hence their parameter specification doesn’t affect the observed delay. Through simple tuning though, we found that $\psi = 0.25$ resulted in good performance.

In tuning for general problems, there are theoretical reasons why it is difficult to attain acceleration without some prior knowledge of $\sigma$, the strong convexity modulus Arjevani (2017). Ideally $\sigma$ is pre-specified for instance in a regularization term. If the Lipschitz constants $L_i$ cannot be calculated directly (which is rarely the case for the classic dual problem of empirical risk minimization objectives), the line-search method discussed in Roux et al. (2012) Section 4 can be used.

A.2 Sparse update formulation

As mentioned in Section 5, authors in Lee & Sidford (2013a) proposed a linear transformation of an accelerated RBCD scheme that results in sparse coordinate updates. Our proposed algorithm can be given a similar efficient implementation. We may eliminate $x_k$ from $\text{A2BCD}$, and derive the equivalent iteration below:

$$
\begin{pmatrix}
  y_{k+1} \\
  v_{k+1}
\end{pmatrix} = \begin{pmatrix}
  1 - \alpha \beta & \alpha \beta \\
  1 - \beta & \beta
\end{pmatrix} \begin{pmatrix}
  y_k \\
  v_k
\end{pmatrix} - \begin{pmatrix}
  \alpha \sigma^{-1/2} L_i^{-1/2} + h(1 - \alpha) L_i^{-1} \nabla_i f(\hat{y}_k) \\
  \sigma^{-1/2} L_i^{-1/2} \nabla_i f(\hat{y}_k)
\end{pmatrix} \\
\triangleq C \begin{pmatrix}
  y_k \\
  v_k
\end{pmatrix} - Q_k
$$
where $C$ and $Q_k$ are defined in the obvious way. Hence we define auxiliary variables $p_k, q_k$ defined via:

$$
\begin{pmatrix}
  y_k \\
  v_k
\end{pmatrix} = C^k
\begin{pmatrix}
  p_k \\
  q_k
\end{pmatrix}
$$

(A.1)

These clearly follow the iteration:

$$
\begin{pmatrix}
  p_{k+1} \\
  q_{k+1}
\end{pmatrix} = \begin{pmatrix}
  p_k \\
  q_k
\end{pmatrix} - C^{-(k+1)}Q_k
$$

(A.2)

Since the vector $Q_k$ is sparse, we can evolve variables $p_k$, and $q_k$ in a sparse manner, and recover the original iteration variables at the end of the algorithm via A.1.

The gradient of the dual function is given by:

$$
\nabla D(y) = \frac{1}{\lambda d} \left( \frac{1}{d} A^T A y + \lambda (y + l) \right)
$$

As mentioned before, it is necessary to maintain or recover $Ay_k$ to calculate block gradients. Since $Ay_k$ can be recovered via the linear relation in equation A.1, and the gradient is an affine function, we maintain the auxiliary vectors $Ap^k$ and $Aq^k$ instead.

Hence we propose the following efficient implementation in Algorithm 1. We used this to generate the results in Table 5. We also note also that it can improve performance to periodically recover $v_k$ and $y_k$, reset the values of $p^k, q^k$, and $C$ to $v_k, y_k$, and $I$ respectively, and restarting the scheme (which can be done cheaply in time $O(d)$).

We let $B \in \mathbb{R}^{2 \times 2}$ represent $C^k$, and $b$ represent $B^{-1}$. $\otimes$ is the Kronecker product. Each computing node has local outdated versions of $p, q, Ap, Aq$ which we denote $\hat{p}, \hat{q}, \hat{A}p, \hat{A}q$ respectively. We also find it convenient to define:

$$
\begin{bmatrix}
  D_1^k \\
  D_2^k
\end{bmatrix} = \begin{bmatrix}
  \alpha \sigma^{-1/2} L_{ik}^{-1/2} + h(1-\alpha) L_{ik}^{-1} \\
  \sigma^{-1/2} L_{ik}^{-1/2}
\end{bmatrix}
$$

(A.3)
Algorithm 1 Shared-memory implementation of A2BCD

1: **Inputs:** Function parameters $A$, $\lambda$, $L$, $\{L_i\}_{i=1}^n$, $n$, $d$. Delay $\tau$ (obtained in dry run). Starting vectors $y$, $v$.
2: **Shared data:** Solution vectors $p$, $q$; auxiliary vectors $Ap$, $Aq$; sparsifying matrix $B$
3: **Node local data:** Solution vectors $\hat{p}$, $\hat{q}$, auxiliary vectors $\hat{Ap}$, $\hat{Aq}$, sparsifying matrix $\hat{B}$.
4: Calculate parameters $\psi$, $\alpha$, $\beta$, $h$ via 1. Set $k = 0$.
5: **Initializations:** $p \leftarrow y$, $q \leftarrow v$, $Ap \leftarrow Ay$, $Aq \leftarrow Av$, $B \leftarrow I$.
6: while not converged, each computing node asynchronous do
7: Randomly select block $i$ via equation 2.1.
8: Read shared data into local memory: $\hat{p} \leftarrow p$, $\hat{q} \leftarrow q$, $\hat{Ap} \leftarrow Ap$, $\hat{Aq} \leftarrow Aq$, $\hat{B} \leftarrow B$.
9: Compute block gradient: $\nabla_i f(\hat{y}) = \frac{1}{n} \lambda \left( \hat{B}_{1,1} \hat{Ap} + \hat{B}_{1,2} \hat{Aq} \right) + \lambda \left( \hat{B}_{1,1} \hat{p} + \hat{B}_{1,2} \hat{q} \right)$
10: Compute quantity $g_i = A^T_i \nabla_i f(\hat{y})$
11: Update $B \leftarrow \begin{bmatrix} 1 - \alpha \beta & \alpha \beta \\ \alpha \beta & 1 - \beta \end{bmatrix} \times B$, calculate inverse $b \leftarrow B^{-1}$.
12: $\begin{bmatrix} p \\ q \end{bmatrix} - b \begin{bmatrix} D_1^k \\ D_2^k \end{bmatrix} \odot \nabla_i f(\hat{y})$, $\begin{bmatrix} Ap \\ Aq \end{bmatrix} - b \begin{bmatrix} D_1^k \\ D_2^k \end{bmatrix} \odot g_i$
13: Increase iteration count: $k \leftarrow k + 1$
14: end while
15: Recover original iteration variables: $\begin{bmatrix} y \\ v \end{bmatrix} \leftarrow B \begin{bmatrix} p \\ q \end{bmatrix}$. Output $y$.

B Proof of the main result

We first recall a couple of inequalities for convex functions.

**Lemma 7.** Let $f$ be $\sigma$-strongly convex with $L$-Lipschitz gradient. Then we have:

$$f(y) \leq f(x) + (y - x, \nabla f(x)) + \frac{1}{2L} \|y - x\|^2, \forall x, y \tag{B.1}$$

$$f(y) \geq f(x) + (y - x, \nabla f(x)) + \frac{1}{2\sigma} \|y - x\|^2, \forall x, y \tag{B.2}$$

We also find it convenient to define the norm:

$$\|s\|_* = \sqrt{\sum_{i=1}^{n} L_i^{-1/2} \|s_i\|^2} \tag{B.3}$$
B.1 Starting point

First notice that using the definition equation 2.8 of \( v_{k+1} \) we have:
\[
\|v_{k+1}\|^2 = \|\beta v_k + (1 - \beta) y_k\|^2 - 2\sigma^{-1/2}L_{ik}^{-1/2}\langle \beta v_k + (1 - \beta) y_k, \nabla f(\hat{y}_k) \rangle + \sigma^{-1}L_{ik}^{-1}\|\nabla_i f(\hat{y}_k)\|^2
\]
\[
\mathbb{E}_k\|v_{k+1}\|^2 = \underbrace{\|\beta v_k + (1 - \beta) y_k\|^2 - 2\sigma^{-1/2}S^{-1} \langle \beta v_k + (1 - \beta) y_k, \nabla f(\hat{y}_k) \rangle}_{A} + \underbrace{S^{-1}\sum_{i=1}^{n} L_i^{-1/2}\|\nabla_i f(\hat{y}_k)\|^2}_{B} + \underbrace{\sum_{i=1}^{n} L_i^{-1/2}\|\nabla_i f(\hat{y}_k)\|^2}_{C}
\]

We have the following general identity:
\[
\|\beta x + (1 - \beta) y\|^2 = \beta\|x\|^2 + (1 - \beta)\|y\|^2 - \beta(1 - \beta)\|x - y\|^2, \forall x, y
\]

It can also easily be verified from equation 2.6 that we have:
\[
v_k = y_k + \alpha^{-1}(1 - \alpha)(y_k - x_k)
\]

Using equation B.5 on term A, equation B.6 on term B, and recalling the definition equation B.3 on term C, we have from equation B.4:
\[
\mathbb{E}_k\|v_{k+1}\|^2 = \underbrace{\|\beta v_k\|^2 + (1 - \beta)\|y_k\|^2 - \beta(1 - \beta)\|v_k - y_k\|^2 + S^{-1}\sigma^{-1/2}\|\nabla f(\hat{y}_k)\|^2}_{B} - \underbrace{2\sigma^{-1/2}S^{-1}\beta \alpha^{-1}(1 - \alpha)\|y_k - x_k, \nabla f(\hat{y}_k)\| - 2\sigma^{-1/2}S^{-1}\langle y_k, \nabla f(\hat{y}_k) \rangle}_{C}
\]

This inequality is our starting point. We analyze the terms on the second line in the next section.

B.2 The Cross Term

To analyze these terms, we need a small lemma. This lemma is fundamental in allowing us to deal with asynchronicity.

**Lemma 8.** Let \( \chi, A > 0 \). Let the delay be bounded by \( \tau \). Then:
\[
A\|\hat{y}_k - y_k\| \leq \frac{1}{2} \chi^{-1}A^2 + \frac{1}{2} \chi \tau \sum_{j=1}^{\tau} \|y_{k+1-j} - y_{k-j}\|^2
\]

**Proof.** See Hannah & Yin (2017a).

**Lemma 9.** We have:
\[
-\langle \nabla f(\hat{y}_k), y_k \rangle \leq -f(y_k) - \frac{1}{2} \sigma(1 - \psi)\|y_k\|^2 + \frac{1}{2} L\kappa \psi^{-1} L_{ik}^{-1/2} \sum_{j=1}^{\tau} \|y_{k+1-j} - y_{k-j}\|^2
\]
\[
\langle \nabla f(\hat{y}_k), x_k - y_k \rangle \leq f(x_k) - f(y_k)
\]
\[
+ \frac{1}{2} L\alpha(1 - \alpha)^{-1} \left( \kappa^{-1} \psi \beta \|v_k - y_k\|^2 + \kappa \psi^{-1} \beta^{-1} \tau \sum_{j=1}^{\tau} \|y_{k+1-j} - y_{k-j}\|^2 \right)
\]

The terms in bold in equation B.8 and equation B.9 are a result of the asynchronicity, and are identically 0 in its absence.

17
Proof. Our strategy is to separately analyze terms that appear in the traditional analysis of Nesterov (2012), and the terms that result from asynchronicity. We first prove equation B.8:

\[-\langle \nabla f(\hat{y}_k), y_k \rangle = -\langle \nabla f(y_k), y_k \rangle - \langle \nabla f(\hat{y}_k) - \nabla f(y_k), y_k \rangle \leq -f(y_k) - \frac{1}{2} \sigma \|y_k\|^2 + L \|\hat{y}_k - y_k\| \|y_k\|\]  

(B.10)
equation B.10 follows from strong convexity (equation B.2 with \(x = y_k\) and \(y = x_s\)), and the fact that \(\nabla f\) is \(L\)-Lipschitz. The term due to asynchronicity becomes:

\[L \|\hat{y}_k - y_k\| \|y_k\| \leq \frac{1}{2} L \kappa^{-1} \psi \|y_k\|^2 + \frac{1}{2} L \kappa^{-1} \psi^{-1} \tau \sum_{j=1}^{\tau} \|y_{k+1-j} - y_{k-j}\|^2\]

using Lemma 8 with \(\chi = \kappa \psi^{-1}, A = \|y_k\|.\) Combining this with equation B.10 completes the proof of equation B.8.

We now prove equation B.9:

\[\langle \nabla f(\hat{y}_k), x_k - y_k \rangle = \langle \nabla f(y_k), x_k - y_k \rangle + \langle \nabla f(\hat{y}_k) - \nabla f(y_k), x_k - y_k \rangle \leq f(x_k) - f(y_k) + L \|\hat{y}_k - y_k\| \|x_k - y_k\| \leq f(x_k) - f(y_k) + \frac{1}{2} L \left( \kappa^{-1} \beta \alpha^{-1} (1 - \alpha) \|x_k - y_k\|^2 + \kappa \psi^{-1} \beta^{-1} \alpha (1 - \alpha)^{-1} \tau \sum_{j=1}^{\tau} \|y_{k+1-j} - y_{k-j}\|^2 \right)\]

Here the last line follows from Lemma 8 with \(\chi = \kappa \psi^{-1} \beta^{-1} \alpha (1 - \alpha)^{-1}, A = nx_k - y_k\). We can complete the proof using the following identity that can be easily obtained from equation 2.6:

\[y_k - x_k = \alpha (1 - \alpha)^{-1} (\nu_k - y_k)\]

\[\square\]

B.3 Function-value term

Much like Nesterov (2012), we need a \(f(x_k)\) term in the Lyapunov function (see the middle of page 357). However we additionally need to consider asynchronicity when analyzing the growth of this term. Again terms due to asynchronicity are emboldened.

Lemma 10. We have:

\[E_k f(x_{k+1}) \leq f(y_k) - \frac{1}{2} h \left( 2 - h \left( 1 + \frac{1}{2} \sigma^{1/2} L^{-1/2} \psi \right) \right) S^{-1} \|\nabla f(\hat{y}_k)\|^2 + S^{-1} L \sigma^{1/2} \kappa \psi^{-1} \tau \sum_{j=1}^{\tau} \|y_{k+1-j} - y_{k-j}\|^2\]

Proof. From the definition equation 2.7 of \(x_{k+1}\), we can see that \(x_{k+1} - y_k\) is supported on block \(i_k\). Since each gradient block \(\nabla_i f\) is \(L_i\) Lipschitz with respect to changes to block \(i\), we can use...
We then apply Lemma 8 to this with
\[
 f(x_{k+1}) \leq f(y_k) + \langle \nabla f(y_k), x_{k+1} - y_k \rangle + \frac{1}{2} L_i \| x_{k+1} - y_k \|^2
\]

(from equation 2.7) = \[ \begin{align*}
 f(y_k) - h L_i^{-1} \langle \nabla f(y_k), \nabla f(\hat{y}_k) \rangle & + \frac{1}{2} h^2 L_i^{-1} \| \nabla f(\hat{y}_k) \|^2 \\
 & - \frac{1}{2} h(2 - h) L_i^{-1} \| \nabla f(\hat{y}_k) \|^2 \\
 E_k f(x_{k+1}) & \leq f(y_k) - h S^{-1} \sum_{i=1}^{n} L_i^{-1/2} (\nabla_i f(y_k) - \nabla_i f(\hat{y}_k)) - \frac{1}{2} h(2 - h) S^{-1} \| \nabla f(\hat{y}_k) \|^2 
\end{align*} \]

(B.11)

Here the last line followed from the definition equation B.3 of the norm \( \|\cdot\|_{1/2} \). We now analyze the middle term:

\[
 - \sum_{i=1}^{n} L_i^{-1/2} (\nabla_i f(y_k) - \nabla_i f(\hat{y}_k)) \\
 = - \sum_{i=1}^{n} L_i^{-1/4} (\nabla_i^4 f(y_k) - \nabla_i^4 f(\hat{y}_k)) \sum_{i=1}^{n} L_i^{-1/4} \nabla_i f(\hat{y}_k) \\
 \text{(Cauchy Schwarz)} \leq \left\| \sum_{i=1}^{n} L_i^{-1/4} (\nabla_i^4 f(y_k) - \nabla_i^4 f(\hat{y}_k)) \right\| \left\| \sum_{i=1}^{n} L_i^{-1/4} \nabla_i f(\hat{y}_k) \right\| \\
 = \left( \sum_{i=1}^{n} L_i^{-1/2} \right)^{1/2} \left( \sum_{i=1}^{n} L_i^{-1/2} \right)^{1/2} \\
 \text{(L \leq L_i, \forall i and equation B.3) \leq L^{-1/4} \| \nabla f(y_k) - \nabla f(\hat{y}_k) \|_* \| \nabla f(\hat{y}_k) \|_*} \\
 \text{\( \nabla f \) is \( L \)-Lipschitz)} \leq L^{-1/4} L \| y_k - \hat{y}_k \| \| \nabla f(\hat{y}_k) \|_* \\
\]

We then apply Lemma 8 to this with \( \chi = 2 h^{-1} \sigma^{1/2} \frac{1}{4} k^2 \gamma^{-1}, A = \| \nabla f(\hat{y}_k) \|_* \) to yield:

\[ \begin{align*}
 - \sum_{i=1}^{n} L_i^{-1/2} (\nabla_i f(y_k) - \nabla_i f(\hat{y}_k)) & \leq h^{-1} L \sigma^{1/2} \gamma^{-1} \sum_{j=1}^{\tau} \| y_{k+1-j} - y_{k-j} \|^2 \\
 & + \frac{1}{4} h \sigma^{1/2} L^{-1/2} \| \nabla f(\hat{y}_k) \|^2 
\end{align*} \]

(B.12)

Finally to complete the proof, we combine equation B.11, with equation B.12.

\[ \square \]

### B.4 Asynchronicity error

The previous inequalities produced difference terms of the form \( \| y_{k+1-j} - y_{k-j} \|^2 \). The following lemma shows how these errors can be incorporated into a Lyapunov function.

**Lemma 11.** Let \( 0 < r < 1 \) and consider the asynchronicity error and corresponding coefficients:

\[ A_k = \sum_{j=1}^{\infty} c_j \| y_{k+1-j} - y_{k-j} \|^2 \]

\[ c_i = \sum_{j=i}^{\infty} r^{i-j-1} s_j \]
This sum satisfies:
\[ E_k[A_{k+1} - rA_k] = c_1E_k\|y_{k+1} - y_k\|^2 - \sum_{j=1}^{\infty} s_j\|y_{k+1-j} - y_{k-j}\|^2 \]

**Remark 2. Interpretation.** This result means that an asynchronicity error term \( A_k \) can negate a series of difference terms \( -\sum_{j=1}^{\infty} s_j\|y_{k+1-j} - y_{k-j}\|^2 \) at the cost of producing an additional error \( c_1E_k\|y_{k+1} - y_k\|^2 \), while maintaining a convergence rate of \( r \). This essentially converts difference terms, which are hard to deal with, into a \( \|y_{k+1} - y_k\|^2 \) term which can be negated by other terms in the Lyapunov function. The proof is straightforward.

**Proof.**
\[
E_k[A_{k+1} - rA_k] = E_k \sum_{j=0}^{\infty} c_{j+1}\|y_{k+1-j} - y_{k-j}\|^2 - rE_k \sum_{j=1}^{\infty} c_j\|y_{k+1-j} - y_{k-j}\|^2 \\
= c_1E_k\|y_{k+1} - y_k\|^2 + E_k \sum_{j=1}^{\infty} (c_{j+1} - rc_j)\|y_{k+1-j} - y_{k-j}\|^2
\]

Noting the following completes the proof:
\[
c_{i+1} - rc_i = \sum_{j=i+1}^{\infty} r^{i+1-j-1}s_j - r \sum_{j=i}^{\infty} r^{i-j-1}s_j = -s_i \]

Given that \( A_k \) allows us to negate difference terms, we now analyze the cost \( c_1E_k\|y_{k+1} - y_k\|^2 \) of this negation.

**Lemma 12.** We have:
\[
E_k\|y_{k+1} - y_k\|^2 \leq 2\alpha^2\beta^2\|v_k - y_k\|^2 + 2S^{-1}L^{-1}\|\nabla f(\hat{y}_k)\|^2
\]

**Proof.**
\[
y_{k+1} - y_k = (\alpha v_{k+1} + (1 - \alpha)x_{k+1}) - y_k \\
= \alpha(\beta v_k + (1 - \beta)y_k - \sigma^{-1/2}L^{-1/2}_{ik}\nabla_i f(\hat{y}_k)) + (1 - \alpha)(y_k - hL^{-1}_{ik}\nabla_i f(\hat{y}_k)) - y_k \\
= \alpha\beta v_k + \alpha(1 - \beta)y_k - \alpha\sigma^{-1/2}L^{-1/2}_{ik}\nabla_i f(\hat{y}_k) - \alpha y_k - (1 - \alpha)hL^{-1}_{ik}\nabla_i f(\hat{y}_k) \\
= \alpha\beta(v_k - y_k) - \left(\alpha\sigma^{-1/2}L^{-1/2}_{ik} + h(1 - \alpha)L^{-1}_{ik}\right)\nabla_i f(\hat{y}_k) \\
\|y_{k+1} - y_k\|^2 \leq 2\alpha^2\beta^2\|v_k - y_k\|^2 + 2\left(\alpha\sigma^{-1/2}L^{-1/2}_{ik} + h(1 - \alpha)L^{-1}_{ik}\right)^2\|\nabla_i f(\hat{y}_k)\|^2
\]
Here equation B.13 following from equation 2.8, the definition of $v_{k+1}$. equation B.14 follows from the inequality $\|x + y\|^2 \leq 2\|x\|^2 + 2\|y\|^2$. The rest is simple algebraic manipulation.

$$\|y_{k+1} - y_k\|^2 \leq 2\alpha^2 \beta^2 \|v_k - y_k\|^2 + 2L_k^{-1} \left( \alpha \sigma^{-1/2} + h(1 - \alpha)L_k^{1/2} \right)^2 \|\nabla f(y_k)\|^2$$

$$= 2\alpha^2 \beta^2 \|v_k - y_k\|^2 + 2L_k^{-1} \left( \frac{1}{2} \sigma^{-1/2} \alpha + h(1 - \alpha) \right)^2 \|\nabla f(y_k)\|^2$$

$$= 2\alpha^2 \beta^2 \|v_k - y_k\|^2 + 2S^{-1} \left( \frac{1}{2} \psi - \sigma^{-1/2} S^{-1} L_k \right)$$

Finally, to complete the proof, we prove $L_k^{1/2} \sigma^{-1/2} \alpha + h(1 - \alpha) \leq 1$.

$$(L_k^{1/2} \sigma^{-1/2} \alpha + h(1 - \alpha)) = h + \alpha \left( L_k^{1/2} \sigma^{-1/2} - h \right)$$

(definitions of $h$ and $\alpha$: equation 2.3, and equation 2.5) $= 1 - \frac{1}{2} \sigma^{1/2} L_k^{1/2} \psi + \sigma^{1/2} S^{-1} \left( L_k^{1/2} \sigma^{-1/2} \right)$$

$$\leq 1 - \sigma^{1/2} L_k^{1/2} \left( \frac{1}{2} \psi - \sigma^{-1/2} S^{-1} L_k \right)$$

(B.15)

Rearranging the definition of $\psi$, we have:

$$S^{-1} = \frac{1}{92} \psi^2 L_k^{1/2} L_k^{-3/2} \kappa^{-1/2} \tau^{-2}$$

($\tau \geq 1$ and $\psi \leq \frac{1}{2}$) $\leq \frac{1}{18^2} L_k^{1/2} L_k^{-3/2} \kappa^{-1/2}$

Using this on equation B.15, we have:

$$L_k^{1/2} \alpha \sigma^{-1/2} + h(1 - \alpha) \leq 1 - \sigma^{1/2} L_k^{1/2} \left( \frac{1}{2} \psi - \frac{1}{18^2} L_k^{1/2} L_k^{-3/2} \kappa^{-1/2} \sigma^{-1/2} L_k \right)$$

$$= 1 - \sigma^{1/2} L_k^{1/2} \left( \frac{1}{2} \psi - \frac{1}{18^2} (L_k/L)^2 \right)$$

$$= 1 - \sigma^{1/2} L_k^{-1/2} \left( \frac{1}{24} - \frac{1}{18^2} \right) \leq 1.$$  

This completes the proof. 

\[\square\]

B.5 Master inequality

We are finally in a position to bring together all the all the previous results together into a master inequality for the Lyapunov function $\rho_k$ (defined in equation 2.11). After this lemma is proven, we will prove that the right hand size is negative, which will imply that $\rho_k$ linearly converges to 0 with rate $\beta$. 

---

21
Lemma 13. Master inequality. We have:

\[ E_k[\rho_{k+1} - \beta \rho_k] \]
\[
\leq + \| y_k \|^2 \times \left( 1 - \beta - \sigma^{-1/2} S^{-1} \sigma (1 - \psi) \right) \\
+ \| v_k - y_k \|^2 \times \beta \left( 2 \alpha^2 \beta c_1 + S^{-1} \beta L^{1/2} \kappa^{-1/2} \psi - (1 - \beta) \right) \\
+ f(y_k) \times \left( c - 2 \sigma^{-1/2} S^{-1} \beta \alpha^{-1} (1 - \alpha) + 1 \right) \\
+ f(x_k) \times \beta \left( 2 \sigma^{-1/2} S^{-1} \alpha^{-1} (1 - \alpha) - c \right) \\
+ \sum_{j=1}^{\tau} \| y_{k+1-j} - y_{k-j} \|^2 \times S^{-1} L \kappa \psi^{-1} \tau \sigma^{1/2} (2 \sigma^{-1} + c) - s \\
+ \| \nabla f(\hat{y}_k) \|^2 \times S^{-1} \left( \sigma^{-1} + 2 L^{-1} c_1 - \frac{1}{2} \epsilon h \left( 2 - h \left( 1 + \frac{1}{2} \sigma^{1/2} L^{-1/2} \psi \right) \right) \right)
\]

Proof.

\[ E_k \| v_{k+1} \|^2 - \beta \| v_k \|^2 \]
\[
(B.7) = (1 - \beta) \| y_k \|^2 - \beta (1 - \beta) \| v_k - y_k \|^2 + S^{-1} \sigma^{-1} \| \nabla f(\hat{y}_k) \|^2 \\
- 2 \sigma^{-1/2} S^{-1} \langle y_k, \nabla f(\hat{y}_k) \rangle \\
- 2 \sigma^{-1/2} S^{-1} \beta \alpha^{-1} (1 - \alpha) \langle y_k - x_k, \nabla f(\hat{y}_k) \rangle \\
\leq (1 - \beta) \| y_k \|^2 - \beta (1 - \beta) \| v_k - y_k \|^2 + S^{-1} \sigma^{-1} \| \nabla f(\hat{y}_k) \|^2 \\
(B.8) + 2 \sigma^{-1/2} S^{-1} \left( -f(y_k) - \frac{1}{2} \sigma (1 - \psi) \| y_k \|^2 + \frac{1}{2} L \kappa \psi^{-1} \tau \sum_{j=1}^{\tau} \| y_{k+1-j} - y_{k-j} \|^2 \right) \\
(B.9) - 2 \sigma^{-1/2} S^{-1} \beta \alpha^{-1} (1 - \alpha) (f(x_k) - f(y_k)) \\
+ \sigma^{-1/2} S^{-1} \beta L \left( \kappa^{-1} \psi \beta \| v_k - y_k \|^2 + \kappa \psi^{-1} \beta^{-1} \tau \sum_{j=1}^{\tau} \| y_{k+1-j} - y_{k-j} \|^2 \right)
\]

We now collect and organize the similar terms of this inequality.

\[
\leq + \| y_k \|^2 \times \left( 1 - \beta - \sigma^{-1/2} S^{-1} \sigma (1 - \psi) \right) \\
+ \| v_k - y_k \|^2 \times \beta \left( \sigma^{-1/2} S^{-1} \beta L \kappa^{-1} \psi - (1 - \beta) \right) \\
- f(y_k) \times 2 \sigma^{-1/2} S^{-1} (\beta \alpha^{-1} (1 - \alpha) + 1) \\
+ f(x_k) \times 2 \sigma^{-1/2} S^{-1} \beta \alpha^{-1} (1 - \alpha) \\
+ \sum_{j=1}^{\tau} \| y_{k+1-j} - y_{k-j} \|^2 \times 2 \sigma^{-1/2} S^{-1} L \kappa \psi^{-1} \tau \\
+ \| \nabla f(\hat{y}_k) \|^2 \times \sigma^{-1} S^{-1}
\]
Now finally, we add the function-value and asynchronicity terms to our analysis. We use Lemma 11 is with \( r = 1 - \sigma^{1/2}S^{-1} \), and
\[
    s_i = \begin{cases} 
    s = 6S^{-1}L^{1/2}k^{3/2}\psi^{-1}\tau, & 1 \leq i \leq \tau \\
    0, & i > \tau 
    \end{cases} \tag{B.18}
\]
Notice that this choice of \( s_i \) will recover the coefficient formula given in equation 2.9. Hence we have:
\[
    \mathbb{E}_k[cf(x_{k+1}) + A_{k+1} - \beta(cf(x_k) + A_k)] \\
    \quad \text{(Lemma 10)} \leq cf(y_k) - \frac{1}{2}c\kappa \left( 2 - \frac{1}{2} \left( 1 + \frac{1}{2} \sigma^{1/2}L^{1/2}\psi \right) \right) S^{-1} \| \nabla f(\hat{y}_k) \|^2 - \beta cf(x_k) \\
    \quad + S^{-1} L \sigma^{1/2} \kappa \psi^{-1} \tau \sum_{j=1}^{\tau} \| y_{k+1-j} - y_{k-j} \|^2 \\
    \quad \text{(Lemmas 11 and 12)} + c_1 \left( 2\alpha^2 \beta^2 \| v_k - y_k \|^2 + 2S^{-1}L^{-1} \| \nabla f(\hat{y}_k) \|^2 \right) \\
    \quad - \sum_{j=1}^{\infty} s_j \| y_{k+1-j} - y_{k-j} \|^2 + A_k(r - \beta) \tag{B.19}
\]
Notice \( A_k(r - \beta) \leq 0 \). Finally, combining equation B.17 and equation B.19 completes the proof. \( \square \)

In the next section, we will prove that every coefficient on the right hand side of equation B.16 is 0 or less, which will complete the proof of Theorem 1.

B.6 Proof of main theorem

**Lemma 14.** The coefficients of \( \| y_k \|^2 \), \( f(y_k) \), and \( \sum_{j=1}^{\tau} \| y_{k+1-j} - y_{k-j} \|^2 \) in Lemma 13 are non-positive.

**Proof.** The coefficient \( 1 - (1 - \psi)\sigma^{1/2}S^{-1} - \beta \| y_k \|^2 \) is identically 0 via the definition equation 2.4 of \( \beta \). The coefficient \( c - 2\sigma^{-1/2}S^{-1}(\beta \alpha^{-1}(1 - \alpha) + 1) \) of \( f(y_k) \) is identically 0 via the definition equation 2.12 of \( c \).

First notice from the definition equation 2.12 of \( c \):
\[
    c = 2\sigma^{-1/2}S^{-1}(\beta \alpha^{-1}(1 - \alpha) + 1) \\
    \text{(definitions of } \alpha, \beta) = 2\sigma^{-1/2}S^{-1} \left( (1 - \sigma^{1/2}S^{-1}(1 - \psi)) (1 + \psi) \sigma^{-1/2}S + 1 \right) \\
    = 2\sigma^{-1/2}S^{-1} \left( (1 + \psi) \sigma^{-1/2}S + \psi^2 \right) \\
    = 2\sigma^{-1} \left( (1 + \psi) + \psi^2 \sigma^{1/2}S^{-1} \right) \tag{B.21}
\]
\[
    c \leq 4\sigma^{-1} \tag{B.22}
\]
Here the last line followed since $\psi \leq \frac{1}{2}$ and $\sigma^{1/2}S^{-1} \leq 1$. We now analyze the coefficient of $\sum_{j=1}^{\tau} \|y_{k+1-j} - y_{k-j}\|^2$.

\[
S^{-1}L_\psi \psi^{-1} \tau \sigma^{1/2} (2\sigma^{-1} + c) - s
\]

(B.22) \leq 6L^{1/2} \kappa^{3/2} \psi^{-1} \tau - s

(definition equation B.18 of $s$)

\[\text{Lemma 15.} \quad \text{The coefficient } \beta(2\sigma^{-1/2}S^{-1}\alpha^{-1}(1 - \alpha) - c) \text{ of } f(x_k) \text{ in Lemma 13 is non-positive.}\]

\text{Proof.} \quad 2\sigma^{-1/2}S^{-1}\alpha^{-1}(1 - \alpha) - c

\[
(B.21) = 2\sigma^{-1/2}S^{-1}(1 + \psi)\sigma^{-1/2}S - 2\sigma^{-1}\left(1 + \psi\right) + \psi^2\sigma^{1/2}S^{-1}
\]

\[
= 2\sigma^{-1}\left(1 + \psi\right) - \left(1 + \psi\right) + \psi^2\sigma^{1/2}S^{-1}
\]

\[
= -2\psi^2\sigma^{-1/2}S^{-1} \leq 0 \quad \square
\]

\text{Lemma 16.} \quad \text{The coefficient } S^{-1}\left(\sigma^{-1} + 2L^{-1}c_1 - \frac{1}{2}\chi\left(2 - h\left(1 + \frac{1}{2}\sigma^{1/2}L^{-1/2}\psi\right)\right)\right) \text{ of } \|\nabla f(\hat{y}_k)\|_2^2 \text{ in Lemma 13 is non-positive.}\]

\text{Proof.} \quad \text{We first need to bound } c_1.

[(equation B.18 and equation 2.9)] \quad c_1 = s \sum_{j=1}^{\tau} \left(1 - \sigma^{1/2}S^{-1}\right)^{-j}

\[
\text{equation B.18} \quad \leq 6S^{-1}L^{1/2} \kappa^{3/2} \psi^{-1} \tau \sum_{j=1}^{\tau} \left(1 - \sigma^{1/2}S^{-1}\right)^{-j}
\]

\[
\leq 6S^{-1}L^{1/2} \kappa^{3/2} \psi^{-1} \tau^2 \left(1 - \sigma^{1/2}S^{-1}\right)^{-\tau}
\]

\text{It can be easily verified that if } x \leq \frac{1}{2} \text{ and } y \geq 0, \text{ then } (1 - x)^{-y} \leq \exp(2xy). \text{ Using this fact with } x = \sigma^{1/2}S^{-1} \text{ and } y = \tau, \text{ we have:}

\[
\leq 6S^{-1}L^{1/2} \kappa^{3/2} \psi^{-1} \tau^2 \exp\left(\tau\sigma^{1/2}S^{-1}\right)
\]

\text{(since } \psi \leq 3/7 \text{ and hence } \tau\sigma^{1/2}S^{-1} \leq \frac{1}{7}) \leq S^{-1}L^{1/2} \kappa^{3/2} \psi^{-1} \tau^2 \times 6 \exp\left(\frac{1}{7}\right)

\[
c_1 \leq 7S^{-1}L^{1/2} \kappa^{3/2} \psi^{-1} \tau^2
\]

(B.23)
We now analyze the coefficient of $\| \nabla f(\hat{y}_k) \|^2$
\[ \sigma^{-1} + 2L^{-1}c_1 - \frac{1}{2}ch\left(2 - h\left(1 + \frac{1}{2}\sigma^{1/2}L^{-1/2}\psi\right)\right) \]
(B.23 and 2.5) \leq \sigma^{-1} + 14S^{-1}L^{-1}L^{1/2}\kappa^{3/2}\psi^{-1}\tau^2 - \frac{1}{2}ch\left(1 + \frac{1}{4}\sigma^{1/2}L^{-1}\psi^2\right) \leq \sigma^{-1} + 14S^{-1}L^{-1}L^{1/2}\kappa^{3/2}\psi^{-1}\tau^2 - \frac{1}{2}ch 
(definition 2.2 of $\psi$) \[ \frac{1}{81}\sigma^{-1}\psi - \frac{1}{2}ch \]
(B.21, definition 2.5 of $h$) \[ \sigma^{-1}\left(1 + \frac{14}{81}\psi - \left(1 + \psi) + \psi^2\sigma^{1/2}S^{-1}\right)\left(1 - \frac{1}{2}\sigma^{1/2}L^{-1}\psi^2\right)\right) \]
$(\sigma^{1/2}L^{-1/2} \leq 0$ and $\sigma^{1/2}S^{-1} \leq 1$) \[ \sigma^{-1}\left(1 + \frac{14}{81}\psi - (1 + \psi)\left(1 - \frac{1}{2}\psi\right)\right) \]
\[ \sigma^{-1}\psi\left(\frac{14}{81} + \frac{1}{2}\psi - \frac{1}{2}\right) \]
$(\psi \leq \frac{1}{2}) \leq 0 \]

**Lemma 17.** The coefficient $\beta(2\alpha^2\beta c_1 + S^{-1}\beta L^{1/2}\kappa^{-1/2}\psi - (1 - \beta))$ of $\|v_k - y_k\|^2$ in 13 is non-positive.

**Proof.**
\[ 2\alpha^2\beta c_1 + \sigma^{1/2}S^{-1}\beta\psi - (1 - \psi)\sigma^{1/2}S^{-1} \]
(B.23) \[ \leq 14\alpha^2\beta S^{-1}L^{1/2}\kappa^{3/2}\psi^{-1}\tau^2 + \sigma^{1/2}S^{-1}\beta\psi - (1 - \psi)\sigma^{1/2}S^{-1} \leq 14\sigma S^{-3}L^{1/2}\kappa^{3/2}\psi^{-1}\tau^2 + \sigma^{1/2}S^{-1}\psi - (1 - \psi)\sigma^{1/2}S^{-1} \]
\[ \sigma^{1/2}S^{-1}(14S^{-2}L\kappa\tau^2\psi^{-1} + 2\psi - 1) \]
Here the last inequality follows since $\beta \leq 1$ and $\alpha \leq \sigma^{1/2}S^{-1}$. We now rearrange the definition of $\psi$ to yield the identity:
\[ S^{-2}\kappa = \frac{1}{94}L^2L^{-3}\tau^{-4}\psi^4 \]
Using this, we have:
\[ 14S^{-2}L\kappa\tau^2\psi^{-1} + 2\psi - 1 = \frac{14}{94}L^2L^{-2}\psi^3\tau^{-2} + 2\psi - 1 \leq \frac{14}{94}\left(\frac{3}{7}\right)^3 1^2 + \frac{6}{7} - 1 \leq 0 \]
Here the last line followed since $L \leq L$, $\psi \leq \frac{3}{7}$, and $\tau \geq 1$. Hence the proof is complete. 

**Proof of Theorem 1.** Using the master inequality 13 in combination with the previous Lemmas 14, 15, 16, and 17, we have:
\[ E_k[\rho_{k+1}] \leq \beta\rho_k = \left(1 - (1 - \psi)\sigma^{1/2}S^{-1}\right)\rho_k \]
When we have:

\[
\left(1 - (1 - \psi)\sigma^{1/2}S^{-1}\right)^k \leq \epsilon
\]

then the Lyapunov function \(\rho_k\) has decreased below \(\epsilon\rho_0\) in expectation. Hence the complexity \(K(\epsilon)\) satisfies:

\[
K(\epsilon) \ln(1 - (1 - \psi)\sigma^{1/2}S^{-1}) = \ln(\epsilon)
\]

\[
K(\epsilon) = \frac{-1}{\ln(1 - (1 - \psi)\sigma^{1/2}S^{-1})} \ln(1/\epsilon)
\]

Now it can be shown that for \(0 < x \leq \frac{1}{2}\), we have:

\[
\frac{1}{x} - 1 \leq \frac{-1}{\ln(1-x)} \leq \frac{1}{x} - \frac{1}{2}
\]

\[
\frac{-1}{\ln(1-x)} = \frac{1}{x} + O(1)
\]

Since \(n \geq 2\), we have \(\sigma^{1/2}S^{-1} \leq \frac{1}{2}\). Hence:

\[
K(\epsilon) = \frac{1}{1 - \psi} \left(\sigma^{-1/2}S + O(1)\right) \ln(1/\epsilon)
\]

An expression for \(K_{\text{ACDM}}(\epsilon)\), the complexity of \(\text{NU\_ACDM}\) follows by similar reasoning.

\[
K_{\text{NU\_ACDM}}(\epsilon) = \left(\sigma^{-1/2}S + O(1)\right) \ln(1/\epsilon)
\]  \hfill (B.24)

Finally we have:

\[
K(\epsilon) = \frac{1}{1 - \psi} \left(\sigma^{-1/2}S + O(1)\right) K_{\text{NU\_ACDM}}(\epsilon)
\]

\[
= \frac{1}{1 - \psi} \left(1 + o(1)\right) K_{\text{NU\_ACDM}}(\epsilon)
\]

which completes the proof. \(\square\)

\[\quad\]

C \hspace{1em} \text{Ordinary Differential Equation Analysis}

C.1 \hspace{1em} \text{Derivation of ODE for synchronous A2BCD}

If we take expectations with respect to \(E_k\), then synchronous (no delay) A2BCD becomes:

\[
y_k = \alpha v_k + (1 - \alpha)x_k
\]

\[
E_k x_{k+1} = y_k - n^{-1} \kappa^{-1} \nabla f(y_k)
\]

\[
E_k v_{k+1} = \beta v_k + (1 - \beta)y_k - n^{-1} \kappa^{-1/2} \nabla f(y_k)
\]

We find it convenient to define \(\eta = n^{1/2}\). Inspired by this, we consider the following iteration:

\[
y_k = \alpha v_k + (1 - \alpha)x_k \hspace{1em} \text{\quad (C.1)}
\]

\[
x_{k+1} = y_k - s^{1/2} \kappa^{-1/2} \eta^{-1} \nabla f(y_k) \hspace{1em} \text{\quad (C.2)}
\]

\[
v_{k+1} = \beta v_k + (1 - \beta)y_k - s^{1/2} \eta^{-1} \nabla f(y_k) \hspace{1em} \text{\quad (C.3)}
\]

\[\quad\]
for coefficients:
\[
\begin{align*}
\alpha &= \left(1 + s^{-1/2} \eta \right)^{-1} \\
\beta &= 1 - s^{1/2} \eta^{-1}
\end{align*}
\]

$s$ is a discretization scale parameter that will be sent to $0$ to obtain an ODE analogue of synchronous A2BCD. We first use equation B.6 to eliminate $v_k$ from from equation C.3.

\[
0 = -v_{k+1} + \beta v_k + (1-\beta)y_k - s^{1/2} \eta^{-1} \nabla f(y_k)
\]
\[
0 = -\alpha^{-1} y_{k+1} + \alpha^{-1}(1-\alpha)x_{k+1}
\]
\[
+ \beta(\alpha^{-1} y_k - \alpha^{-1}(1-\alpha)x_k) + (1-\beta)y_k - s^{1/2} \eta^{-1} \nabla f(y_k)
\]
(times by $\alpha$)
\[
0 = -y_{k+1} + (1-\alpha)x_{k+1}
\]
\[
+ \beta(y_k - (1-\alpha)x_k) + \alpha(1-\beta)y_k - \alpha s^{1/2} \eta^{-1} \nabla f(y_k)
\]
\[
= -y_{k+1} + y_k (\beta + \alpha(1-\beta))
\]
\[
+ (1-\alpha)x_{k+1} - x_k \beta(1-\alpha) - \alpha s^{1/2} \eta^{-1} \nabla f(y_k)
\]

We now eliminate $x_k$ using equation C.1:

\[
0 = -y_{k+1} + y_k (\beta + \alpha(1-\beta))
\]
\[
+ (1-\alpha) \left( y_k - s^{1/2} \eta^{-1} \kappa^{-1/2} \nabla f(y_k) \right) - \left( y_{k-1} - s^{1/2} \eta^{-1} \kappa^{-1/2} \nabla f(y_{k-1}) \right) \beta(1-\alpha)
\]
\[
- \alpha s^{1/2} \eta^{-1} \nabla f(y_k)
\]
\[
= -y_{k+1} + y_k (\beta + \alpha(1-\beta) + (1-\alpha)) - \beta(1-\alpha)y_{k-1}
\]
\[
+ s^{1/2} \eta^{-1} \nabla f(y_{k-1})(\beta-1)(1-\alpha)
\]
\[
- \alpha s^{1/2} \eta^{-1} \nabla f(y_k)
\]
\[
= (y_k - y_{k+1}) + \beta(1-\alpha)(y_k - y_{k-1})
\]
\[
+ s^{1/2} \eta^{-1} (\nabla f(y_{k-1})(\beta-1)(1-\alpha) - \alpha \nabla f(y_k))
\]

Now to derive an ODE, we let $y_k = Y(ks^{1/2})$. Then $\nabla f(y_{k-1}) = \nabla f(y_k) + O(s^{1/2})$. Hence the above becomes:

\[
0 = (y_k - y_{k+1}) + \beta(1-\alpha)(y_k - y_{k-1})
\]
\[
+ s^{1/2} \eta^{-1}(\beta-1)(1-\alpha) \nabla f(y_k) + O(s^{3/2})
\]
\[
0 = \left(-s^{1/2} \dot{Y} - \frac{1}{2} s \ddot{Y} \right) + \beta(1-\alpha) \left(s^{1/2} \dot{Y} - \frac{1}{2} s \ddot{Y} \right)
\]
\[
+ s^{1/2} \eta^{-1}(\beta-1)(1-\alpha) \nabla f(y_k) + O(s^{3/2})
\]  

(C.4)
We now look at some of the terms in this equation to find the highest-order dependence on $s$.

\[
\beta(1 - \alpha) = \left(1 - s^{1/2}\eta^{-1}\right) \left(1 - \frac{1}{1 + s^{-1/2}\eta}\right)
= \left(1 - s^{1/2}\eta^{-1}\right) \frac{s^{-1/2}\eta}{1 + s^{-1/2}\eta}
= \frac{s^{-1/2}\eta - 1}{s^{-1/2}\eta + 1}
= \frac{1 - s^{1/2}\eta^{-1}}{1 + s^{1/2}\eta^{-1}}
= 1 - 2s^{1/2}\eta^{-1} + \mathcal{O}(s)
\]

We also have:

\[
(\beta - 1)(1 - \alpha) - \alpha = \beta(1 - \alpha) - 1
= -2s^{1/2}\eta^{-1} + \mathcal{O}(s)
\]

Hence using these facts on equation C.4, we have:

\[
0 = \left(-s^{1/2}\dot{Y} - \frac{1}{2}s\ddot{Y}\right) + \left(1 - 2s^{1/2}\eta^{-1} + \mathcal{O}(s)\right)\left(s^{1/2}\dot{Y} - \frac{1}{2}s\ddot{Y}\right)
+ s^{1/2}\eta^{-1}\left(-2s^{1/2}\eta^{-1} + \mathcal{O}(s)\right)\n\n0 = -s^{1/2}\dot{Y} - \frac{1}{2}s\ddot{Y} + \left(s^{1/2}\dot{Y} - \frac{1}{2}s\ddot{Y} - 2s^{1/2}\eta^{-1}\dot{Y} + \mathcal{O}\left(s^{3/2}\right)\right)
\left(-2s^{1}\eta^{-2} + \mathcal{O}\left(s^{3/2}\right)\right)\n\n0 = -s\dot{Y} - 2s\eta^{-1}\dot{Y} - 2s\eta^{-2}\ddot{Y}\n\n0 = -\dot{Y} - 2\eta^{-1}\dot{Y} - 2\eta^{-2}\ddot{Y}
\]

Taking the limit as $s \to 0$, we obtain the ODE:

\[
\dot{Y}(t) + 2\eta^{-1}\dot{Y} + 2\eta^{-2}\nabla f(Y) = 0
\]

C.2 **Convergence proof for synchronous ODE**

\[
e^{-\eta^{-1}t}E(t) = \langle \nabla f(Y(t)), \dot{Y}(t) \rangle + \eta^{-1}f(Y(t))
+ \frac{1}{2}\langle Y(t) + \eta\dot{Y}(t), \dot{Y}(t) + \eta\ddot{Y}(t) \rangle + \eta^{-1}\frac{1}{4}\|Y(t) + \eta\dot{Y}(t)\|^2
\]

(strong convexity equation B.2) \leq \langle \nabla f(Y), \dot{Y} \rangle + \eta^{-1}\langle \nabla f(Y), Y \rangle - \frac{1}{2}\eta^{-1}\|Y\|^2
+ \frac{1}{2}\langle Y + \eta\dot{Y}, -\dot{Y} - 2\eta^{-1}\nabla f(Y) \rangle + \eta^{-1}\frac{1}{4}\|Y(t) + \eta\dot{Y}(t)\|^2
= -\frac{1}{4}\eta^{-1}\|Y\|^2 - \frac{1}{4}\eta\|\dot{Y}\|^2 \leq 0
\]
Hence we have $E'(t) \leq 0$. Therefore $E(t) \leq E(0)$. That is:

$$E(t) = e^{n_\tau - \kappa^{-1/2}t} \left( f(Y) + \frac{1}{4} \|Y + \eta \hat{Y}\|^2 \right) \leq E(0) = f(Y(0)) + \frac{1}{4} \|Y(0) + \eta \hat{Y}(0)\|^2$$  \hfill (C.5)

which implies:

$$f(Y(t)) + \frac{1}{4} \|Y(t) + \eta \hat{Y}(t)\|^2 \leq e^{n_\tau - \kappa^{-1/2}t} \left( f(Y(0)) + \frac{1}{4} \|Y(0) + \eta \hat{Y}(0)\|^2 \right)$$  \hfill (C.6)

C.3 Asynchronicity error lemma

This result is the continuous-time analogue of Lemma 11. First notice that $c(0) = c_0$ and $c(\tau) = 0$. We also have:

$$c'(t)/c_0 = -r e^{-rt} - r e^{-rt} \frac{e^{-r\tau}}{1 - e^{-r\tau}}$$

$$= -r \left( e^{-rt} + e^{-rt} \frac{e^{-r\tau}}{1 - e^{-r\tau}} \right)$$

$$= -r \left( e^{-rt} + (e^{-rt} - 1) \frac{e^{-r\tau}}{1 - e^{-r\tau}} \right)$$

$$c'(t) = -r c(t) - r c_0 \frac{e^{-r\tau}}{1 - e^{-r\tau}}$$

Hence using $c(\tau) = 0$:

$$A'(t) = c_0 \|\dot{Y}(t)\|^2 + \int_{t-\tau}^t c'(t-s) \|\dot{Y}(s)\|^2 ds$$

$$= c_0 \|\dot{Y}(t)\|^2 - r A(t) - r c_0 \frac{e^{-r\tau}}{1 - e^{-r\tau}} D(t)$$

Now when $x \leq \frac{1}{2}$, we have $\frac{e^{-\tau}}{1 - e^{-\tau}} \geq \frac{1}{2} x^{-1}$. Hence when $r \tau \leq \frac{1}{2}$, we have:

$$A'(t) \leq c_0 \|\dot{Y}(t)\|^2 - r A(t) - \frac{1}{2} r \tau c_0 D(t)$$

and the result easily follows.

C.4 Convergence analysis for the asynchronous ODE

We consider the same energy as in the synchronous case (that is, the ODE in equation 3.1). Similar to before, we have:

$$e^{-\eta^{-1}t} E'(t) \leq \langle \nabla f(Y), \dot{Y} \rangle + \eta^{-1} \langle \nabla f(Y), Y \rangle - \frac{1}{2} \eta^{-1} \|Y\|^2$$

$$+ \frac{1}{2} \langle Y + \eta \dot{Y}, -\dot{Y} - 2 \eta^{-1} \nabla f \left( \dot{Y} \right) \rangle + \eta^{-1} \frac{1}{4} \|Y(t) + \eta \hat{Y}(t)\|^2$$

$$= \langle \nabla f(Y), \dot{Y} \rangle + \eta^{-1} \langle \nabla f(Y), Y \rangle - \frac{1}{2} \eta^{-1} \|Y\|^2$$

$$+ \frac{1}{2} \langle Y + \eta \dot{Y}, -\dot{Y} - 2 \eta^{-1} \nabla f(Y) \rangle + \eta^{-1} \frac{1}{4} \|Y(t) + \eta \hat{Y}(t)\|^2$$

$$- \eta^{-1} \langle Y + \eta \dot{Y}, \nabla f \left( \dot{Y} \right) - \nabla f(Y) \rangle$$

$$= -\frac{1}{4} \eta^{-1} \|Y\|^2 - \frac{1}{4} \eta \|\dot{Y}\|^2 - \eta^{-1} \langle Y + \eta \dot{Y}, \nabla f \left( \dot{Y} \right) - \nabla f(Y) \rangle$$
where the final equality follows from the proof in Section C.2. Hence

\[ e^{-\eta^{-1}t}E'(t) \leq -\frac{1}{4}\eta^{-1}\|Y\|^2 + \frac{1}{4}\eta\|\dot{Y}\|^2 + L\eta^{-1}\|Y\|\|\dot{Y} - Y\| + L\|\dot{Y}\|\|\dot{Y} - Y\| \]  

(C.7)

Now we present an inequality that is similar to equation 8.

**Lemma 18.** Let \( A, \chi > 0 \). Then:

\[ \|Y(t) - \dot{Y}(t)\| A \leq \frac{1}{2}\chi \tau D(t) + \frac{1}{2}\chi^{-1}A^2 \]

**Proof.** Since \( \dot{Y}(t) \) is a delayed version of \( Y(t) \), we have: \( \dot{Y}(t) = Y(t - j(t)) \) for some function \( j(t) \geq 0 \) (though this can be easily generalized to an inconsistent read). Recall that for \( \chi > 0 \), we have

\[ ab \leq \frac{1}{2}(\chi a^2 + \chi^{-1}b^2) \]

Hence

\[
X(t) - \dot{X}(t) = \int_{s=t-j(t)}^t X'(s)ds \\
\|X(t) - \dot{X}(t)\| A = \left\| \int_{s=t-j(t)}^t X'(s)ds \right\| A \\
\leq \frac{1}{2}\chi \left\| \int_{s=t-j(t)}^t X'(s)ds \right\|^2 + \frac{1}{2}\chi^{-1}A^2 \\
(\text{Holder’s inequality}) \leq \frac{1}{2}\chi \left( \int_{s=t-j(t)}^t \|X'(s)\|^2 ds \right) \left( \int_{s=t-j(t)}^t 1 ds \right) + \frac{1}{2}\chi^{-1}A^2 \\
\leq \frac{1}{2}\chi \tau \left( \int_{s=t-j(t)}^t \|X'(s)\|^2 ds \right) + \frac{1}{2}\chi^{-1}A^2
\]

We use this lemma twice on \( \|Y\|\|\dot{Y} - Y\| \) and \( \|\dot{Y}\|\|\dot{Y} - Y\| \) in equation C.7 with \( \chi = 2L, A = \|Y\| \) and \( \chi = 4L\eta^{-1}, A = \|\dot{Y}\| \) respectively, to yield:

\[
e^{-\eta^{-1}t}E'(t) \leq -\frac{1}{4}\eta^{-1}\|Y\|^2 + \frac{1}{4}\eta\|\dot{Y}\|^2 + L\eta^{-1}\left(L\tau D(t) + \frac{1}{4}L^{-1}\|Y\|^2 \right) + L\left(2L\eta^{-1}\tau D(t) + \frac{1}{8}L^{-1}\eta\|\dot{Y}\|^2 \right) \\
= -\frac{1}{8}\eta\|\dot{Y}\|^2 + 3L^2\eta^{-1}\tau D(t)
\]

The proof of convergence is completed in Section 3.

**D Optimality proof**

For parameter set \( \sigma, L_1, \ldots, L_n, n \), we construct a block-separable function \( f \) on the space \( \mathbb{R}^{bn} \) (separated into \( n \) blocks of size \( b \)), which will imply this lower bound. Define \( \kappa_i = L_i/\sigma \). We define
the matrix \( A_i \in \mathbb{R}^{b \times b} \) via:

\[
A_i \triangleq \begin{pmatrix}
2 & -1 & 0 \\
-1 & 2 & \ddots & \ddots \\
0 & \ddots & \ddots & -1 & 0 \\
\ddots & & -1 & 2 & -1 \\
0 & -1 & \theta_i & \ddots & \ddots \\
\end{pmatrix}, \quad \text{for } \theta_i = \frac{\kappa_i^{1/2} + 3}{\kappa_i^{1/2} + 1}.
\]

Hence we define \( f_i \) on \( \mathbb{R}^b \) via:

\[
f_i = \frac{L_i - \sigma}{4} \left( \frac{1}{2} \langle x, A_i x \rangle - \langle e_1, x \rangle \right) + \frac{\sigma}{2} \| x \|^2
\]

which is clearly \( \sigma \)-strongly convex and \( L_i \)-Lipschitz on \( \mathbb{R}^b \). From Lemma 8 of Lan & Zhou (2015), we know that this function has unique minimizer

\[
x_{\ast,(i)} \triangleq (q_i, q_i^2, \ldots, q_i^b), \quad \text{for } q = \frac{\kappa_i^{1/2} - 1}{\kappa_i^{1/2} + 1}.
\]

Finally, we define \( f \) via:

\[
f(x) \triangleq \sum_{i=1}^n f_i(x_{(i)}).
\]

Now let \( e(i, j) \) be the \( j \)th unit vector of the \( i \)th block of size \( b \) in \( \mathbb{R}^{b n} \). For \( I_1, \ldots, I_n \in \mathbb{N} \), we define the subspaces

\[
V_i(I) = \text{span}\{e(i, 1), \ldots, e(i, I)\},
\]

\[
V(I_1, \ldots, I_n) = V(I_1) \oplus \ldots \oplus V_n(I_n).
\]

\( V(I_1, \ldots, I_n) \) is the subspace with the first \( I_1 \) components of block 1 nonzero, the first \( I_2 \) components of block 2 nonzero, etc. First notice that \( \text{IC}(V(I_1, \ldots, I_n)) = V(I_1, \ldots, I_n) \). Also, clearly, we have:

\[
\nabla_i f(V(I_1, \ldots, I_n)) \subset V(0, \ldots, 0, \min\{I_1 + 1, b\}, 0, \ldots, 0).
\]

\( \nabla_i f \) is supported on the \( i \)th block, hence why all the other indices are 0. The pattern of nonzeros in \( A \) means that the gradient will have at most 1 more nonzero on the \( i \)th block (see Nesterov (2013)).

Let the initial point \( x_0 \) belong to \( V(I_1, \ldots, I_n) \). Let \( I_{K,i} \) be the number of times we have had \( i_k = i \) for \( k = 0, \ldots, K - 1 \). By induction on condition 2 of Definition 4 using equation D.1, we have:

\[
x_k \in V(\min\{I_1 + I_{k,1}, b\}, \ldots, \min\{I_n + I_{k,m}, b\})
\]

Hence if \( x_0,(i) \in V_i(0) \) and \( k \leq b \), then

\[
\| x_k,(i) - x_{\ast,(i)} \|^2 \geq \min_{x \in V_i(I_{k,i})} \| x - x_{\ast,(i)} \|^2 = \sum_{j=I_{k,i}+1}^b q_i^{2j} = \left( q_i^{2I_{k,i}+2} - q_i^{2b+2} \right)/(1 - q_i^2)
\]

Therefore for all \( i \), we have:

\[
\mathbb{E}\| x_k \|^2 \geq \mathbb{E}\| x_k,(i) - x_{\ast,(i)} \|^2 \geq \mathbb{E}\left[ \left( q_i^{2I_{k,i}+2} - q_i^{2b+2} \right)/(1 - q_i^2) \right]
\]
To evaluate this expectation, we note:

\[ \mathbb{E}q_{t, \delta}^{2j} = \sum_{j=0}^{k} \binom{k}{j} p_j^j (1 - p_i)^{k-j} q_t^{2j} \]

\[ = (1 - p_i)^k \sum_{j=0}^{k} \binom{k}{j} (q_t^2 p_i (1 - p_i)^{-1})^j \]

\[ = (1 - p_i)^k (1 + q_t^2 p_i (1 - p_i)^{-1})^k \]

\[ = (1 - (1 - q_t^2) p_i)^k \]

Hence

\[ \mathbb{E}\|x_k - x^*\|^2 \geq (1 - (1 - q_t^2) p_i)^k - q_t^{2b} / (1 - q_t^2). \]

For any i, we may select the starting iterate \( x_0 \) by defining its block \( j = 1, \ldots, n \) via:

\[ x_{0, (j)} = (1 - \delta_{ij}) x_{\ast, (j)} \]

For such a choice of \( x_0 \), we have

\[ \|x_0 - x^*\|^2 = \|x_{\ast, (i)}\|^2 = q_t^2 + \ldots + q_t^{2b} = q_t^2 \frac{1 - q_t^{2b}}{1 - q_t^2} \]

Hence for this choice of \( x_0 \):

\[ \mathbb{E}\|x_k - x^*\|^2 / \|x_0 - x^*\|^2 \geq (1 - (1 - q_t^2) p_i)^k - q_t^{2b} / (1 - q_t^2) \]

Now notice:

\[ (1 - (1 - q_t^2) p_i)^k = (q_t^{i-2} - (q_t^{-2} - 1) p_i)^k q_t^{2k} \geq q_t^{2k} \]

Hence

\[ \mathbb{E}\|x_k - x^*\|^2 / \|x_0 - x^*\|^2 \geq (1 - (1 - q_t^2) p_i)^k (1 - q_t^{2b-2k}) / (1 - q_t^{2b}) \]

Now if we let \( b = 2k \), then we have:

\[ \mathbb{E}\|x_k - x^*\|^2 / \|x_0 - x^*\|^2 \geq (1 - (1 - q_t^2) p_i)^k (1 - q_t^{2k}) / (1 - q_t^{4k}) \]

\[ = (1 - (1 - q_t^2) p_i)^k / (1 + q_t^{2k}) \]

\[ \mathbb{E}\|x_k - x^*\|^2 / \|x_0 - x^*\|^2 \geq \frac{1}{2} \max (1 - (1 - q_t^2) p_i)^k \]

Now let us take the minimum of the right-hand side over the parameters \( p_i \), subject to \( \sum_{i=1}^{n} p_i = 1 \). The solution to this minimization is clearly:

\[ p_i = (1 - q_t^2)^{-1} / \left( \sum_{j=1}^{n} (1 - q_j^2)^{-1} \right) \]
Hence
\[
\mathbb{E}\|x_k - x^*\|^2 / \|x_0 - x^*\|^2 \geq \frac{1}{2} \left( 1 - \left( \sum_{j=1}^{n} (1 - q_j^2)^{-1} \right)^{-1} \right)^k
\]
\[
\sum_{j=1}^{n} (1 - q_j^2)^{-1} = \frac{1}{4} \sum_{j=1}^{n} \left( \kappa_i^{1/2} + 2 + \kappa_i^{-1/2} \right)
\]
\[
\geq \frac{1}{4} \left( \sum_{j=1}^{n} \kappa_i^{1/2} + 2n \right)
\]
\[
\mathbb{E}\|x_k - x^*\|^2 / \|x_0 - x^*\|^2 \geq \frac{1}{2} \left( 1 - \frac{4}{\sum_{j=1}^{n} \kappa_i^{1/2} + 2n} \right)^k
\]

Hence the complexity \(I(\epsilon)\) satisfies:
\[
\epsilon \geq \frac{1}{2} \left( 1 - \frac{4}{\sum_{j=1}^{n} \kappa_i^{1/2} + 2n} \right)^{I(\epsilon)}
\]
\[
I(\epsilon) \geq - \left( \ln \left( 1 - \frac{4}{\sum_{j=1}^{n} \kappa_i^{1/2} + 2n} \right) \right)^{-1} \ln(1/2\epsilon)
\]
\[
= \frac{1}{4} (1 + o(1)) \left( n + \sum_{j=1}^{n} \kappa_i^{1/2} \right) \ln(1/2\epsilon)
\]

\section{E Extensions}

As mentioned, a stronger result than Theorem 1 is possible. In the case when \(L_i = L\) for all \(i\), we can consider a slight modification of the coefficients:

\[
\alpha \triangleq \left( 1 + (1 + \psi)\sigma^{-1/2} S \right)^{-1} \quad \text{(E.1)}
\]
\[
y_k = \alpha v_k + (1 - \alpha) x_k, \quad \text{(E.4)}
\]
\[
\beta \triangleq 1 - (1 + \psi)^{-1} \sigma^{1/2} S^{-1} \quad \text{(E.2)}
\]
\[
x_{k+1} = y_k - hL^{-1}\nabla_i f(\hat{y}_k), \quad \text{(E.5)}
\]
\[
h \triangleq \left( 1 + \frac{1}{2} \sigma^{1/2} L^{-1/2} \psi \right)^{-1}, \quad \text{(E.3)}
\]
\[
v_{k+1} = \beta v_k + (1 - \beta) y_k - \sigma^{-1/2} L^{-1/2} \nabla_i f(\hat{y}_k). \quad \text{(E.6)}
\]

for the asynchronicity parameter:
\[
\psi = 6n^{1/2} \kappa^{-1} \times \tau \quad \text{(E.7)}
\]

This leads to complexity:
\[
K(\epsilon) = (1 + \psi) n \kappa^{1/2} \ln(1/\epsilon) \quad \text{(E.8)}
\]

Here there is no restriction on \(\psi\) as in Theorem 1, and hence there is no restriction on \(\tau\). Assuming \(\psi \leq 1\) gives optimal complexity to within a constant factor. Notice then that the resulting condition of \(\tau\)
\[
\tau \leq \frac{1}{6} n \kappa^{-1/2} \quad \text{(E.9)}
\]
now essentially matches the one in Theorem 3 in Section 3. While this result is stronger, it increases the complexity of the proof substantially. So in the interests of space and simplicity, we do not prove this stronger result.