UNIVERSITY OF CALIFORNIA
Los Angeles

Fundamental Results on Asynchronous Parallel Optimization Algorithms

A dissertation submitted in partial satisfaction
of the requirements for the degree
Doctor of Philosophy in Mathematics

by

Robert Rafaeil Hannah

2019

ABSTRACT OF THE DISSERTATION

Fundamental Results
on Asynchronous Parallel
Optimization Algorithms

by

Robert Rafaeil Hannah
Doctor of Philosophy in Mathematics
University of California, Los Angeles, 2019
Professor Wotao Yin, Chair

In this thesis, we present a body of work on the performance and convergence properties of asynchronous-parallel algorithms completed over the course of my doctorate degree (Hannah, Feng, and Wotao Yin 2018; Hannah and Wotao Yin 2017b; T. Sun, Hannah, and Wotao Yin 2017; Hannah and Wotao Yin 2017a). Asynchronous algorithms eliminate the costly synchronization penalty of traditional synchronous-parallel algorithms. They do this by having computing nodes utilize the most recently available information to compute updates. However, it's not immediately clear whether the trade-off of eliminating synchronization penalty at the cost of using outdated information is favorable.

We first give a comprehensive theoretical justification of the performance advantages of asynchronous algorithms, which we summarize as "Faster Iterations, Same Quality" (Hannah and Wotao Yin 2017a). Under a well-justified model, we show that asynchronous algorithms complete "Faster Iterations". Using renewal theory, we demonstrate how network delays, heterogeneous sub-problem difficulty and computing power greatly hinder synchronous algorithms, but have no impact on their asynchronous counterparts. We next prove the first exact convergence rate results for a variety of synchronous algorithms including synchronous ARock and synchronous randomized block coordinate descent (sync-RBCD). This allows us to make a fair comparison between these algorithms and their asynchronous counterparts.

Finally we show that a variety of asynchronous algorithms have a convergence rate that essentially matches the previously derived exact rates for synchronous counterparts so long as the delays are not too large. Hence asynchronous algorithms complete faster iteration that are of the "Same Quality" as synchronous algorithms. Therefore we conclude that a wide variety of asynchronous algorithms will always outcompete their synchronous counterparts if the delays are not too large, and especially at scale.

Next we present the first asynchonous Nesterov-accelerated algorithm that attains a speedup: `A2BCD` (Hannah, Feng, and Wotao Yin 2018). We first prove that `A2BCD` attains `NU_ACDM`'s complexity to highest order. `NU_ACDM` is a state-of-the-art accelerated coordinate descent algorithm (Allen-Zhu, Qu, et al. 2016). Then we show that both `A2BCD` and `NU_ACDM` both have optimal complexity. Hence because `A2BCD` has faster iterations, and optimal complexity, it should be the fastest coordinate descent algorithm. We verify this with numerical experiments comparing `A2BCD` with `NU_ACDM`. We find that `A2BCD` is up to 4-5x faster than `NU_ACDM`, and hence conclude that our algorithm is the current fastest coordinate descent algorithm that exists. Finally we derive a second-order ODE, which is the continuous-time limit of `A2BCD`. The ODE analysis motivates and clarifies our proof strategy.

Lastly, we present earlier foundational work that comprises the basis of the technical innovations that made the previous results possible (Hannah and Wotao Yin 2017b). We show that ARock and its many special cases may converge even under unbounded delays (both stochastic and deterministic). These results sidestep longstanding impossibility results derived in the 1980s by making slightly stronger assumptions. They were also an early demonstration of the power of meticulous Lyapunov-function construction techniques pioneered in this body of work.

The dissertation of Robert Rafaeil Hannah is approved.

Lieven Vandenberghe

Deanna Needell

Stanley J. Osher

Wotao Yin, Committee Chair

University of California, Los Angeles

2019

To my parents, William and Mary Hannah, who over decades supported me and made this possible.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

I wanted take this opportunity to thank my family, friends, collaborators and my committee.

I am particularly indebted to my adviser, Professor Wotao Yin, who despite a multitude of responsibilities has been the best mentor and collaborator that I could have asked for. I was extremely fortunate that I met Prof. Yin years ago when he came to UCLA. Over the years his advice, wisdom, and understanding have guided me to great success in my research, and life in general.

I also wanted to thank the other members of my committee, Prof. Stanley Osher, Prof. Deanna Needell, and Prof. Vandenberghe for their advice, time and feedback. Your perspective has greatly improved my work.

I also wanted to thank my collaborators: Dr. Lin Xiao, and Dr. Zeyuan Allen-Zhu for giving me the opportunity to collaborate with them at Microsoft Research; Prof. Ernest Ryu for his frequent advice, and for greatly improving my Scaled Relative Graph framework; Fei Feng, Prof. Daniel O'Connor, Tao Sun, and Yanli Liu for their hard work and insights on our shared projects.

Further, I wanted to thank many others who I've interacted with over the years and have helped steer my research in a positive direction, including Dr. Zhimin Peng, Prof. Damek Davis, Dr. Yat Tin Show, Dr. Tianyu Wu, Dr. Kun Yuan, Prof. Yangyang Xu, and many others.

Finally, I wanted to thank my parents William and Mary Hannah who made this possible through tremendous personal sacrifice over many years – staying up late helping me with homework, quizzing me for mathematics tests on the train, sending me to Sydney Grammar School, and generally providing a loving environment that allowed me to reach my potential.

VITA

2007-2012	Bachelor of Advanced Science (Mathematics, and Physics), University of Sydney, Australia.
2013-2019	Teaching and Research Assistant, Department of Mathematics University of California, Los Angeles, Los Angeles, CA. Teaching assistant graduate-level ODEs and PDEs.
2017	Software Engineering Internship (Bing Ads, Core AI) Microsoft Corporation, Washington.
2018	Research Intern (Machine Learning and Optimization Group) Microsoft Corporation, Washington.

PUBLICATIONS

Ernest K. Ryu, Robert Hannah, and Wotao Yin. "Scaled Relative Graph: Nonexpansive Operators via 2D Euclidean Geometry." In: (Feb. 26, 2019). arXiv: 1902.09788 [math]

Robert Hannah, Fei Feng, and Wotao Yin. "A2BCD: Asynchronous Acceleration with Optimal Complexity." In: International Conference on Learning Representations. Sept. 27, 2018

Robert Hannah, Yanli Liu, et al. "Breaking the Span Assumption Yields Fast Finite-Sum Minimization." In: *Advances in Neural Information Processing Systems 31*. Curran Associates, Inc., 2018, pp. 2312–2321

Robert Hannah and Wotao Yin. "More Iterations per Second, Same Quality – Why Asynchronous Algorithms May Drastically Outperform Traditional Ones." In: (Aug. 17, 2017). arXiv: 1708.05136

Tao Sun, Robert Hannah, and Wotao Yin. "Asynchronous Coordinate Descent under More Realistic Assumptions." In: *Advances in Neural Information Processing Systems 30*. 2017, pp. 6183–6191

Robert Hannah and Wotao Yin. "On Unbounded Delays in Asynchronous Parallel Fixed-Point Algorithms." In: *Journal of Scientific Computing* (Dec. 12, 2017), pp. 1–28

Part I

Introduction

CHAPTER 1

Introduction

The confluence of a combination of factors has lead to the increasing importance and interest in parallelization. Broadly, we have an enormous growth in demand for computation, while at the same time we are running into fundamental barriers to increasing the power of serial and serial-like computing systems.

On the demand side, the most obvious of these factors is the explosion in the availability and size of data sets. Larger data sets allow us to fit more complex models to this data, which requires ever larger amounts of computation, memory and communication. Another related factor is clearly the increasing scale and sophistication of the operations of human endeavor. The larger a system, the larger and more complicated the space of possible decisions is, and the more there is to gain from fractional improvements. For, say, a small bookstore, beyond several obvious decisions to increase revenue, there comes a point at which optimizing further is not worth the additional effort. However, even a one percent increase in revenue for, say, Amazon would have a multi-billion dollar impact. Nothing is too small to not be worth optimizing.

On the supply side, we have both technological and theoretical factors. The power of an individual core, after 30 years' exponential growth, has stopped increasing significantly since 2005. Before this point, running the same serial algorithm on the same problem would become faster year-over-year because of the increasing power of individual cores, however this is no longer the case. Moving forward, CPUs will only become faster through the addition of more cores rather than more powerful cores (Sutter 2005; Sutter 2011). Moreover, even if Moore's law had continued to hold on a single chip, the growth rate of the size of data sets has vastly outstripped the growth rate of computational power for quite some time.

We are also at the point at which serial or parallel-agnostic algorithms are reaching their limits in many settings. There are now many serial algorithms that are essentially optimal in many contexts – for instance Nesterov's accelerated gradient method in the full-gradient setting (Yurii Nesterov 1983), `NU_ACDM` in the coordinate setting (Allen-Zhu, Qu, et al. 2016), and Katyusha in the finite-sum setting (Allen-Zhu 2017). Thankfully parallel optimization is far less well understood, and we have many less-explored avenues for increasing algorithmic efficiency.

Parallel optimization is fundamentally more complex than serial optimization. This is because many factors that are insignificant or even non-existent for serial algorithms can become the dominant cost at scale in parallel systems. And clearly, all limiting factors to the efficiency of serial systems are necessarily potentially limiting factors in parallel systems as well. These parallel factors include communication bandwidth, latency, iteration efficiency, network topology, heterogeneity, and many others – some surely undiscovered. The computational complexity of a serial optimization algorithm has always been a reasonable proxy for the algorithm's speed (with many exceptions however). In contrast, for large parallel systems this is not even approximately the case.

Parallel systems exhibit fundamental trade offs, whereby the fastest algorithm will strongly depend on the specific computing system that we are using. So for instance, there is a fundamental trade-off between memory and communication in efficient matrix-matrix multiplication. It is well-known that modern computing systems are vastly over-indexed on floating point operation speed relative to inter-computing-node bandwidth, and hence it can become a dominant cost that should be minimized. In (Solomonik and Demmel 2011) it is shown that having redundant local memory allows computing nodes to "avoid communication", allowing them to complete the multiplication much faster while actually not reducing the total number of floating point operations.

There are a multitude of unresolved questions related to building high-performance solvers. One extremely important factor is delayed communication between computing nodes. This frequently becomes a bottleneck and dominant cost at scale. In this work we present a body of work that helps clarify and resolve this thorny problem in the context of optimization

solvers.

1.1 Motivation

The vast majority of parallel algorithms are *synchronous* algorithms. For instance the synchronous-parallel Gauss-Jacobi algorithm divides the problem space $\mathbb{R}^N$ into p coordinate blocks. At every iteration, these blocks are updated by a corresponding set of p processors, and each processor's update is communicated to every other processor. Synchronous algorithms are simpler to analyze and implement. They are often mathematically equivalent to a serial algorithm, and hence retain the same convergence guarantees. However, they have major drawbacks, such as synchronization penalty. At each iteration, all processors must wait for the results of the slowest processor to be received in order to begin the next iteration.

Synchronous algorithms may become impractical at scale, or on a busy network. Network latency is a major problem and bottleneck for parallel algorithms. Over a 20-25 year period on a wide range of systems, latency has improved by a factor of $20 - 40$ whereas CPU speeds have improved by a factor of 1000 (Rumble et al. 2011). This means that synchronizing at every step can be extremely expensive, and the divergence between processing speeds and latency will make this problem worse over time.

Moreover, these relatively modest improvements in latency refer to the hardware's maximum performance. Latency and bandwidth are much worse in large data centers, which are typically very congested: Spikes in traffic can cause latency to increase temporarily by a factor of 20 (Rumble et al. 2011). Congestion also causes packet loss: Some data may fail to reach all parties, and must be sent again. If any computing node in a synchronous-parallel system experiences congestion issues or packet-loss, the entire system must wait for that one node. In addition, dedicated access to computing nodes often cannot be guaranteed. Nodes may have unexpected or unpredictable demands placed on them by others user, may temporarily go offline, etc. causing further unpredictable delays. The more processors in the system, the more likely that at least of the computing nodes will experience these kinds of problems in each iteration, leading the entire system to frequently grind to a halt.

In addition, sometimes the structure of the problem makes synchronous-parallel solvers inefficient. For instance, it may not be feasible to break a problem into subproblems of equal difficulty. If the computing nodes have roughly equal computational power, nodes that are assigned easier subproblems will frequently be waiting on nodes assigned harder subproblems. The more heterogeneous the difficulty of subproblems, the more problematic this issue becomes.

What is needed is a more flexible framework for parallel optimization: One that is resilient to latency, unpredictable and congested networks, packet loss, heterogeneous subproblem difficulty, and other practical issues.

1.2 Advantages of asynchronous algorithms

A node in an asynchronous algorithm, instead of waiting to receive results from all other nodes, simply computes its next update using the most recent information it has received. Using outdated information will often still result in convergence if the asynchronous algorithm is properly designed.

Latency, congestion, and random delays will no longer cripple the system, because processors can make progress without waiting on the results of the slowest processor. Asynchronous algorithms are resilient to packet-loss, unexpected drains on computing power, the loss of a node, and many other common problems on large congested networks. The speed of asynchronous algorithms is more related to the aggregate computing power and bandwidth of the system, rather than the speed of the slowest processor. In addition, the algorithms discussed in this work dynamically balances load with random coordinate block assignment: Processors take on as much work as they are currently able to, and no workload tuning is required.

There is, however, a trade-off: Using outdated information means the error decreases less per iteration. However more iterations can occur per second because of vastly reduced synchronization penalty. Promising empirical obtained in (Z. Peng et al. 2016) suggest that this trade-off is a favorable one.

1.3 Overview

In this work, we present a series of results on the performance and convergence of asynchronous parallel algorithms developed in a number of recent papers (Hannah, Feng, and Wotao Yin 2018; Hannah and Wotao Yin 2017b; T. Sun, Hannah, and Wotao Yin 2017; Hannah and Wotao Yin 2017a). Part I is introductory. In Chapter 2, we review relevant literature on asynchronous algorithms, coordinate methods, and Nesterov acceleration. In Chapter 3, we review relevant theoretical background and notation for our results. In Chapter 4, we define ARock and our model of asynchronicity more precisely. We also outline the main thesis of this body of work: That asynchronous algorithms complete faster iterations, and suffer no complexity penalty for using outdated information. We also outline the general convergence proof strategy. The remaining parts of this work – Part II, Part III, and Part IV – present the main results of (Hannah and Wotao Yin 2017a; Hannah, Feng, and Wotao Yin 2018; Hannah and Wotao Yin 2017b) respectively.

Part II presents the results of (Hannah and Wotao Yin 2017a), which relate to linear convergence of various (non-accelerated) asynchronous algorithms. Chapter 5 develops a model of the iteration time in synchronous and asynchronous systems, and investigates factors that lead asynchronous algorithms to complete much faster iterations. Chapter 6 proves exact/ sharp convergence rates for various synchronous algorithms, such as synchronous RBCD, so that we are able to make a fair comparison to their asynchronous counterparts. Finally, in Chapter 7 we show that ARock, and hence all of its special cases, has essentially the same complexity as in Chapter 6 so long as the delay is not too large. Hence it suffers no complexity penalty for using outdated information. This holds true even for potentially unbounded delays. Taking the results of this part together, we conclude that a wide variety of asynchronous algorithms will vastly outperform their synchronous counterparts.

Part III presents the results of (Hannah, Feng, and Wotao Yin 2018), which echo those of Part II. We propose and prove the convergence of the **A**synchronous **A**ccelerated Nonuniform Randomized **B**lock **C**oordinate **D**escent algorithm (`A2BCD`), the first asynchronous Nesterov-accelerated algorithm that achieves optimal complexity. `A2BCD` is based on `NU_ACDM`, which

was previously the fastest existing coordinate descent algorithm. Chapter 8 defines A2BCD, and states the main results of this part. In Chapter 9, we prove that A2BCD attains NU_ACDM's state-of-the-art iteration complexity to highest order, so long as delays are not too large. This is significant because it was an open question whether it was possible to make an asynchronous accelerated algorithm that had good complexity. The proof is very different from that of (Allen-Zhu, Qu, et al. 2016), and involves significant technical innovations and complexity related to the analysis of asynchronicity. In Chapter 10, we 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. In Chapter 11, we derive a second-order ordinary differential equation (ODE), which is the continuous-time limit of A2BCD. This extends the ODE found in (Su, Boyd, and Candes 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. The ODE analysis motivates and clarifies the our proof strategy of the main result. In Chapter 12, we confirm with numerical experiments on a small-scale shared-memory architecture that A2BCD is the current fastest coordinate descent algorithm. We find that A2BCD can approximately solve the (dual) ridge regression problem up to $4-5\times$ faster than NU_ACDM for various data sets from LIBSVM (Chang and C.-J. Lin 2011). We also discuss critical elements of an efficient implementation, including the sparse-update reformulation of A2BCD and parameter tuning.

Lastly, in Part IV, we present earlier foundational work that comprises the basis of the technical innovations that made the previous results possible (Hannah and Wotao Yin 2017b). We consider ARock on a merely nonexpansive operator. In contrast to the previous parts, in this regime it is well-known that only weak convergence is possible in general. We extend the results of (Z. Peng et al. 2016) to show that ARock converges weakly to a solution, even under unbounded delays. We consider stochastic delays in Chapter 13, and deterministic unbounded delays in Chapter 14. These results were the first general convergence results for asynchronous unbounded delay. They sidestepped earlier impossibility results (D. P. Bertsekas and J. N. Tsitsiklis 1997) by making slightly stronger assumptions. They were also an early demonstration of the power of meticulous Lyapunov-function construction techniques

pioneered in this body of work.

CHAPTER 2

Literature Review

In this chapter we review previous work related to our results.

2.1 Earlier work

Asynchronous algorithms were first proposed in (Chazan and Miranker 1969) to solve linear systems. Since then, asynchronous algorithms have been applied to many fields including nonlinear systems, differential equations, consensus problems, and optimization. General convergence results and theory were developed later in (D. P. Bertsekas 1983; D. P. Bertsekas and J. N. Tsitsiklis 1997; P. Tseng, D. Bertsekas, and J. Tsitsiklis 1990; Z. Q. Luo and P. Tseng 1992; Z.-Q. Luo and Paul Tseng 1993; P. Tseng 1991) for partially and totally asynchronous systems.

Coordinate algorithms update individual coordinates of a solution vector $(x_1, \ldots, x_m)$ one at a time: first coordinate $i(0)$, then $i(1)$, etc. Until relatively recently, authors assumed that this sequence of coordinates $(x_1, \ldots, x_m)$ is a deterministic sequence with very little restriction. However, this imposes stronger restrictions on the problem. In asynchronous algorithms, the update to the solution vector x^k is computed using information from an old/outdated point that is $j(k)$ iterates in the past. These delays $j(k)$ are usually also assumed to be deterministic, but this appears to be relatively less restrictive. In (D. P. Bertsekas and J. N. Tsitsiklis 1997), the authors describe two basic classes of deterministic asynchronous scenarios that appeared in the literature.

Definition 1. Totally asynchronous iteration. Every block, x_i, is updated infinitely many times. Information from iteration k (i.e. the components of x^k) is only used a finite

number of times.

Total asynchronicity is a very weak condition that leads to convergence results with limited applicability, though there do exist applications to linear problems and strictly convex network flow problems (D. P. Bertsekas and J. N. Tsitsiklis 1997; P. Tseng, D. Bertsekas, and J. Tsitsiklis 1990). For instance, the asynchronous linear iteration $x \mapsto Ax + b$ will only converge in general if the largest eigenvalue of $|A|$ (the matrix obtained by taking an absolute value of every entry) is strictly less than 1 (Chazan and Miranker 1969; D. P. Bertsekas and J. N. Tsitsiklis 1997).

Definition 2. Partially asynchronous iteration. There exists an integer B such that every component, x_i, is updated at least once every B steps; and the information used by the processors cannot be older than B steps (bounded delay).

Partially asynchronous algorithms have better convergence properties. For instance, from (P. Tseng 1991):

Theorem 1. *For strongly convex f with ∇f Lipschitz, there is a step size γ_1 such that for any step size $0 < \gamma < \gamma_1$, asynchronous gradient descent with partial asynchronicity converges at least linearly to a minimum, with rate $\mathcal{O}\Big((1 - c\gamma)^k\Big)$ for some constant c.*

However, the formulas for c or γ_1 are complicated, and the authors did not include them. These constants are also tiny, because one needs to assume the worst-case scenario. The maximum delay B needs to be known in advance to determine the step size.

2.2 More recent work

Stochastic asynchronous algorithms began to appear recently, a popular example being "Hogwild!" (Recht et al. 2011). These algorithms always assume a bounded delay ($j(k) \leq \tau$ for all k and i), and that the sequence of blocks $i(k)$ is chosen independently and identically with $\mathbb{P}[i(k) = j] = p_j$ for fixed nonzero probabilities p_j.

Many works on asynchronous algorithms consider conditions for a linear speedup. However in Part II we show that the complexity of many algorithms is asymptotically equal to that of their synchronous counterparts, which is a much stronger result than linear speedup. All work except (Z. Peng et al. 2016; Hannah and Wotao Yin 2017b; Johnstone and Eckstein 2018; Davis 2016) pertain exclusively to the function-value setting. We work in the operator setting mostly, which means that our results apply to a wider variety of algorithms.

In (Avron, Druinsky, and Gupta 2014), the authors prove linear convergence for an asynchronous stochastic linear solver. In (Ji Liu et al. 2015), the authors prove function-value convergence for asynchronous randomized block coordinate descent (RBCD). In each step, one of the m coordinates is randomly chosen and updated with a gradient descent step. They prove $\mathcal{O}(1/k)$ convergence for f convex with ∇f Lipschitz, and linear convergence when f is also strongly convex. This was extended in (J. Liu and Wright 2015) to composite objective functions. For condition number κ, they report a per-iteration linear convergence rate of $r = 1 - \frac{1}{2m\kappa}$. This implies an iteration complexity approximately 8 times higher than our result in Part II. For linear speedup, they require a bounded delay of $\tau < \infty$ that satisfies $\tau = \mathcal{O}(m^{1/2})$, and $\tau = \mathcal{O}(m^{1/4})$ for composite objectives. Our corresponding condition for bounded delay is $\tau = \mathcal{O}(m^q)$ for $0 \leq q < \frac{1}{2}$ for both composite and non-composite objectives. However, as mentioned, our results hold also for unbounded delays.

In (Z. Peng et al. 2016), the authors propose the ARock algorithm, and prove its convergence under bounded delays. They prove linear speedup for $\tau = \mathcal{O}\left(m^{1/4}\right)$. In (Zhimin Peng, Xu, et al. 2017) authors prove function-value linear convergence of an asynchronous block proximal gradient algorithm under unbounded delays. However in both cases, it is unclear how the iteration complexity they obtain compares to the corresponding synchronous algorithm. Work has also been done on asynchronous algorithms for finite sums in the operator setting (Davis 2016; Johnstone and Eckstein 2018). In (Hannah and Wotao Yin 2017b; T. Sun, Hannah, and Wotao Yin 2017; Zhimin Peng, Xu, et al. 2017; 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 (Mania et al. 2017), authors achieve a linear speedup for $\tau = \mathcal{O}\left(m^{1/6}\right)$, but with complexity $\mathcal{O}(\kappa^2 \ln(1/\epsilon))$ that is $\Omega(\kappa)$ times larger than ours. In (Lian, H. Zhang, et al. 2016), the authors review a number of asynchronous algorithm analyses and collect conditions necessary for linear speedup on a fixed problem. But as mentioned, we prove a much stronger result than linear speedup. Several months after (Hannah and Wotao Yin 2017a) appeared online, (Dutta et al. 2018) made similar arguments about the theoretical advantages of asynchronous algorithms for stochastic gradient descent.

There is also a rich body of work on asynchronous SGD. In the distributed setting, (Z. Zhou et al. 2018) showed global convergence for stochastic variationally coherent problems even when the delays grow at a polynomial rate. In (Lian, W. Zhang, 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 (T. Liu et al. 2018), authors obtained an asymptotic rate of convergence for asynchronous momentum SGD on streaming PCA, which provides insight into the trade-off between asynchrony and momentum.

2.3 Asynchronous acceleration and coordinate descent

Coordinate descent methods, in which a chosen coordinate block $i(k)$ is updated at every iteration, are a popular way to solve minimization problems. Randomized block coordinate descent (`RBCD`, (Y. Nesterov 2012)) updates a uniformly randomly chosen coordinate block $i(k)$ with a gradient-descent-like step: $x_{k+1} = x_k - (1/L_{i(k)})\nabla_{i(k)} f(x_k)$ (for coordinate Lipschitz constants $L_1, \ldots, L_m$). The complexity $K(\epsilon)$ of an algorithm is defined as the number of iterations required to decrease the error $\mathbb{E}(f(x_k) - f(x_*))$ to less than $\epsilon(f(x_0) - f(x_*))$. Randomized coordinate descent on σ-strongly convex, coordinate smooth f has a complexity of $K(\epsilon) = \mathcal{O}(m(\bar{L}/\sigma)\ln(1/\epsilon))$ (for $\bar{L} = n^{-1}\sum_{i=1}^m L_i$).

Using a series of averaging and extrapolation steps, *accelerated* `RBCD` (Y. Nesterov 2012) improves `RBCD`'s iteration complexity $K(\epsilon)$ to $\mathcal{O}(m\sqrt{\bar{L}/\sigma}\ln(1/\epsilon))$, which leads to much faster convergence when $\frac{\bar{L}}{\sigma}$ is large. This rate is optimal when all L_i are equal (Lan and Y. Zhou

2017). Finally, using a special probability distribution for the random block index $i(k)$, the non-uniform accelerated coordinate descent method (Allen-Zhu, Qu, et al. 2016) (`NU_ACDM`) can further decrease the complexity to $\mathcal{O}(\sum_{i=1}^{m} \sqrt{L_i/\sigma}\ln(1/\epsilon))$, which can be up to $\sqrt{m}$ times faster than accelerated `RBCD`, by Cauchy-Schwarz. `NU_ACDM` was the state-of-the-art coordinate descent algorithm for solving minimization problems until (Hannah, Feng, and Wotao Yin 2018).

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.

The work of (Meng et al. 2016) claims to obtain square-root speedup for an asynchronous accelerated SVRG in the case of finite sum minimization $f(x) = n^{-1}\sum_{i=1}^{n} f_i(x)$. In the case where all n component functions have the same Lipschitz constant L, the complexity they obtain reduces to $(n+\kappa)\ln(1/\epsilon)$ for $\kappa = \mathcal{O}(\tau n^2)$ (Corollary 4.4). Hence authors do not even obtain accelerated rates. Their convergence condition is $\tau < \frac{1}{4\Delta^{1/8}}$ for sparsity parameter Δ. 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, Huang, and Z. Lin 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 τ times larger than the serial complexity. Since τ 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, accelerated SVRG under sparsity assumptions. And while we think this is quite likely, they have not yet provided proof.

2.4 Unbounded delay

There were some unbounded delay results before (Hannah and Wotao Yin 2017b) in the stochastic unconstrained convex optimization setting (John C. Duchi, Chaturapruek, and Ré 2015; Sra et al. 2016; Agarwal and John C Duchi 2011) . It is hard to compare results from a different optimization setting to our results. However we note the following: We obtain point convergence ($x^k \rightharpoonup x^*$) rather than function-value convergence ($f\left(x^k\right) \to f(x^*)$) for convex f that is not necessarily strongly convex. The deterministic unbounded delay criterion in Theorem 9 is weaker than all other delay assumptions. The step size in these papers converges to 0 as $k \to \infty$, which is an inevitable part of the problem setting. This makes asynchronicity error less of a problem. Nonetheless, in (Hannah and Wotao Yin 2017b), we are able to prove convergence in our setting with a step size rule that is only a function of the delay distribution despite unbounded delays (Theorem 6). The step size rule is invariant in k, and does not converge to 0. Theorem 9 features a step size that adapts to current delay conditions, once again invariant in k, which is cited as a key advantage of (Sra et al. 2016).

Our result in Theorem 9 can be seen as a halfway point between partial and total asynchrony. Using a slightly stronger assumption than total asynchronicity, we are able to prove a much stronger convergence result.

CHAPTER 3

Preliminaries and Background

In this work, $\mathbb{H}$ will always denote a Hilbert space, with norm $\|\diamond\|$ and inner product $\langle \diamond, \diamond \rangle$. Frequently we will consider coordinate algorithms. Given $\mathbb{H}$, we may split the space into m orthogonal blocks of coordinates, and hence write any $x \in \mathbb{H}$ as:

$$x = (x_1, \ldots, x_m)$$

Here x_i denotes the ith block of x. In general, subscripts will denote blocks, and superscripts will denote iterations. So for instance x_i^k will denote the ith block of the kth iterate of some algorithm. In the same way, we can write a gradient as $\nabla f(x) = (\nabla_1 f(x), \ldots, \nabla_m f(x))$, where $\nabla_i f(x)$ is the ith block of the gradient. P_i will denote the projection onto the ith block: $P_i(x_1, \ldots, x_m) = (0, \ldots, 0, x_i, 0, \ldots, 0)$.

3.1 Convex and smooth functions

For a more thorough discussion of convex and smooth functions, see (Yurii Nesterov 2013; Y. Nesterov 2012). Most the the inequalities that follow are derived in these sources. A function $f : \mathbb{H} \to \mathbb{R} \cup \{\infty\}$ is **convex** if:

$$f(\theta x + (1-\theta)y) \leq \theta f(x) + (1-\theta) f(y), \forall\, x, y \in \mathbb{H},\, \theta \in (0,1)$$

We say f if **proper** if $f(x) < \infty$ at some point x. The **domain** of f denoted $\mathrm{dom}(f)$ is the set of points x for which $f(x) < \infty$. For any differentiable convex function f, we have:

$$f(y) \geq f(x) + \langle \nabla f(x), y - x \rangle, \forall x, y$$

We say that f is **μ-strongly convex** for $\mu > 0$ if $f(x) - \frac{1}{2}\mu\|x\|^2$ is convex. For any differentiable μ-strongly convex f, we have (see (Yurii Nesterov 2013)):

$$f(y) \geq f(x) + \langle \nabla f(x), y - x \rangle + \frac{1}{2}\mu\|x - y\|^2, \forall x, y \tag{3.1.1}$$

In many cases, a convex function f may not be differentiable. However, we may define the **subdifferential** ∂f at a point f via:

$$\partial f(x) = \{u | f(y) \geq f(x) + \langle y - x, u \rangle, \forall x, y\}$$

The subdifferential is a generalization of the gradient for functions that are not differentiable.

We say that f is **L-smooth** if it is differentiable with L-Lipschitz gradient ∇f. That is:

$$\|\nabla f(y) - \nabla f(x)\| \leq L\|y - x\|, \forall x, y$$

For such functions, we have (see (Yurii Nesterov 2013)):

$$f(y) \leq f(x) + \langle \nabla f(x), y - x \rangle + \frac{1}{2}L\|y - x\|^2, \forall x, y \tag{3.1.2}$$

$$\langle \nabla f(y) - \nabla f(x), y - x \rangle \leq L\|y - x\|^2, \forall x, y \tag{3.1.3}$$

For constant $L_1, \ldots, L_m$, we say that f is L_i-coordinate smooth if we have:

$$\|\nabla_i f(y) - \nabla_i f(x)\| \leq L_i\|y - x\|, \forall x, y, i$$

For such functions, we have (see (Y. Nesterov 2012)):

$$f(x + P_i u) \leq f(x) + \langle \nabla f(x), P_i u \rangle + \frac{1}{2}L_i\|P_i u\|^2, \forall x, u, i$$

If f is L_i-coordinate smooth, then clearly it is also smooth with parameter $\sum_{i=1}^m L_i$ (in the worst case).

If a function f is both μ-strongly convex and L-smooth, we have (see (Yurii Nesterov 2013)):

$$\langle \nabla f(y) - \nabla f(x), y - x \rangle \geq \frac{\mu L}{\mu + L}\|y - x\|^2 + \frac{1}{\mu + L}\|\nabla f(y) - \nabla f(x)\|^2$$

Given a proper function $f : \mathbb{H} \to \mathbb{R} \cup \{\infty\}$, the **conjugate** $f^* : \mathbb{H} \to \mathbb{R} \cup \{\infty\}$ is defined via (see (Bauschke and P. L. Combettes 2011) for an in-depth introduction to the conjugate):

$$f^*(u) = \sup\langle x, u \rangle - f(x)$$

3.2 Operators

For a primer on operators, we suggest (Bauschke and P. L. Combettes 2011; Ryu and Boyd 2015). We write $A : \mathbb{H} \rightrightarrows \mathbb{H}$ to denote at **operator**. An operator maps points $x \in \mathbb{H}$ to subsets $A(x) \subset \mathbb{H}$. We will omit the brackets and simply write Ax from now. $\text{dom}(A)$ denotes the set of points $x \in \mathbb{H}$ that do not map to the empty set. We say that an operator is **single-valued** if for each x, Ax is either a singleton or the empty set. Otherwise we say A is **multi-valued**. If A is single-valued, we can identify it with the function $\tilde{A} : \text{dom}(A) \to \mathbb{H}$.

An operator $A : \mathbb{H} \rightrightarrows \mathbb{H}$ can be identified with its graph $\text{gra}(A) \subset \mathbb{H}^2$, which is defined as:

$$\text{gra}(A) = \{(x, y) | y \in Ax\}$$

Using this identification, we can define the inverse of A denotes A^{-1} via its graph:

$$\text{gra}\left(A^{-1}\right) = \{(y, x) | y \in Ax\}$$

For operator A and $\gamma > 0$, we define the **resolvent** $J_{\gamma A}$ and the **reflected resolvent** $R_{\gamma A}$ via:

$$J_{\gamma A} = (I + \gamma A)^{-1}$$
$$R_{\gamma A} = 2J_{\gamma A} - I$$

Hence I is the **identity** function $Ix = x$. The resolvent and reflected resolvent are used in operator splitting methods, which will be discussed later.

An operator is L**-Lischitz** for $L > 0$ if it is single-valued, and

$$\|\nabla f(y) - \nabla f(x)\| \leq L\|y - x\|, \forall x, y \in \text{dom}(f)$$

It is called **nonexpansive** if it is 1-Lipschitz, and an $L-$**contraction** if it is L Lipschitz for $L < 1$. We say an operator A is β**-cocoercive** if it is single-valued and

$$\beta\|Ay - Ax\|^2 \leq \langle Ay - Ax, y - x\rangle, \forall x, y$$

We say an operator A is α**-averaged** if it can be written as $A = (1 - \theta)I + \theta R$ for $\theta \in [0, 1]$ and R nonexpansive. We say an operator A is μ**-strongly monotone** if:

$$\mu\|x - y\|^2 \leq \langle v - u, y - x\rangle, \forall u \in Ax, v \in Ay$$

and we say A is merely **monotone** if it is 0-strongly monotone.

Lemma 3. *Let $S = I - T$, where T is an r-Lipschitz operator. Then for all $x, y \in \mathbb{R}^d$ we have:*

$$\langle Sy - Sx, y - x\rangle \geq \frac{1}{2}\|Sy - Sx\|^2 + \frac{1}{2}\Big(1 - r^2\Big)\|y - x\|^2 \tag{3.2.1}$$

Proof.

$$\begin{aligned}
(T \text{ is } r\text{-Lipschitz})\ r^2\|y - x\|^2 &\geq \|Ty - Tx\|^2 \\
&= \|(I - S)y - (I - S)x\|^2 \\
&= \|Sy - Sx\|^2 - 2\langle Sy - Sx, y - x\rangle + \|y - x\|^2 \\
(\text{rearrange})\ \langle Sy - Sx, y - x\rangle &\geq \frac{1}{2}\|Sy - Sx\|^2 + \frac{1}{2}\Big(1 - r^2\Big)\|y - x\|^2
\end{aligned}$$

□

3.3 KM iterations

Take a nonexpansive operator $T : \mathbb{H} \to \mathbb{H}$. Frequently we wish to find a **fixed point** of such operators. That is, a point x^* such that $Tx^* = x^*$. Frequently, finding such a fixed point may be equivalent to some original problem that we wish to solve. For instance, given convex and L-smooth f, the minimizers x^* of this function are exactly the fixed points of the nonexpansive operator $I - \frac{2}{L}\nabla f$.

Starting at $x^0 \in \mathbb{H}$, the simple **Picard iteration**

$$x^{k+1} = Tx^k$$

will converge linearly to a fixed point x^* with rate $\mathcal{O}\Big(L^k\Big)$ if T is an L-contraction. However if T is merely nonexpansive, this sequence may never converge. For instance $T = -I$ leads to the non-convergent sequence: $x^0, -x^0, x^0, -x^0, \ldots$. However, for $\lambda \in (0, 1)$, consider the Krasnosel'skiĭ–Mann iteration (KM):

$$x^{k+1} = ((1 - \lambda)I + \lambda T)x^k.$$

By averaging T with the identity, the sequence x^k weakly converges to a fixed point of T (see (Bauschke and P. L. Combettes 2011)). Though KM may look unfamiliar, gradient descent is equivalent to KM with operator $T = I - \frac{2}{L}\nabla f$.

An **epoch** of an algorithm is essentially a number of iterations that corresponds to one evaluation of Sx. So for instance, m iterations of ARock corresponds to 1 epoch, since each iteration involves computing $S_{i(k)}$, which is $1/m$ of the computational effort of computing the a full evaluation Sx.

3.4 Special cases of KM

The KM iteration takes many popular algorithms as special cases. In the following table, we demonstrate how common optimization algorithms are simply special cases of KM using the appropriate fixed-point operator. In this table, the gradients ∇f, ∇g, ∇h are Lipschitz with constants L_f, L_g, L_h respectively. We assume f, g, h are all convex. $\text{Proj}_C(x)$ denotes to projections of $x \in \mathbb{H}$ onto a convex set C. 1 denotes the vector 1s. $\mathbf{1}$ denotes $1 \otimes I$, where $\otimes$ is the tensor product. $[v_1; v_2; \ldots; v_n]$ denotes a column vector composed of $v_1, v_2, \ldots$ respectively. A and B are linear operators, $d \in \mathbb{N}\backslash\{0\}$ is a dimension, C is a convex set, and b is a vector.

Columns 1 and 2 contain the optimization problem and a common algorithm used to solve it. Column 3 gives the nonexpansive fixed-point operator T corresponding to the algorithm. A fixed point of T corresponds to a solution to the original optimization problem. When you apply the KM iteration to T, you obtain the algorithm in column 2. Column 4 contains assumptions necessary for convergence. The derivations of the algorithms and operators, as well as the proof of nonexpansiveness, are out of the scope of this paper. We refer the interested reader to (Bauschke and P. L. Combettes 2011; Davis and Wotao Yin 2017; Hannah and Wotao Yin 2017b).

Table 3.1: Selection of common algorithms that are special cases of KM iteration, and their corresponding fixed-point operator.

Optimization problem	**Algorithm**	**Nonexpansive fixed-point operator T**	**Assumption**
$\min\ f(x)$	Gradient descent	$I - \gamma\nabla f$	$\gamma \in (0, \frac{2}{L_f}]$
$\min\ f(x)$	Proximal point	$J_{\gamma\partial f}$	$\gamma > 0$
$\min f(x) + g(x)$	Forward backward	$J_{\gamma\partial f} \circ (I - \gamma\nabla g)$	$\gamma \in (0, \frac{2}{L_g}]$
$\min\{g(x) : x \in C\}$	Projected gradient	$\mathrm{Proj}_C \circ (I - \gamma\nabla g)$	$\gamma \in (0, \frac{2}{L_g}]$
$\min f(x) + g(x)$	Peaceman-Rachford	$R_{\gamma\partial f} \circ R_{\gamma\partial g}$	$\gamma > 0$
$\min \sum_{i=1}^{d} f_i(x)$	Parallel Peaceman-Rachford	$(\frac{2}{d}\mathbf{1}\mathbf{1}^T - I) \circ R_{\gamma\partial\mathbf{f}}$ where $\mathbf{f} = [f_1; \ldots; f_d] : \mathbb{H}^d \to \mathbb{R}^d$	$\gamma > 0$
$\min f(x) + g(x)$	Douglas-Rachford	$\frac{1}{2}I + \frac{1}{2}R_{\gamma\partial f} \circ R_{\gamma\partial g}$	$\gamma > 0$
$\min f(x) + g(x) + h(x)$	Davis-Yin	$I - J_{\gamma\partial g} + J_{\gamma\partial f} \circ (2J_{\gamma\partial g} - I - \gamma\nabla h \circ J_{\gamma\partial g})$	$\gamma \in (0, \frac{2}{L_h}]$
$\min\{f(x)+g(z) : Ax + Bz = b\}$	ADMM	$\frac{1}{2}I + \frac{1}{2}R_{\gamma\partial F} \circ R_{\gamma\partial G}$, where $F(y) := f^*(A^T y)$, $G(y) := g^*(B^T y) - b^T y$	$\gamma > 0$

3.5 Duality and finite sums

In this paper, we only consider coordinate algorithms. This may seem like a limitation. However it is not very restrictive for the following reasons. We aim to study parallel optimization algorithms. For parallelism to be possible, there has to be some kind of way to split an algorithm into sub-problems. The most common ways are to split an algorithms over its coordinate, and to split over functions for finite sum problems:

$$f(x) = n^{-1} \sum_{i=1}^{n} f_i(x)$$

While we are only considering the former splitting, by duality, we can apply coordinate methods to finite-sum problems.

Consider the finite sum problem for convex f_i:

$$P(x) = n^{-1} \sum_{i=1}^{n} f_i(A_i \cdot x) + \frac{1}{2}\sigma\|x\|^2$$

A_i can be viewed as data vectors of some sort. We create n auxillary variables w_i, each corresponding to a function f. Under the constraint $w_i = A_i \cdot x$, we can write this as:

$$P(x, w) = n^{-1} \sum_{i=1}^{n} f_i(w_i) + \frac{1}{2}\sigma\|x\|^2$$

Defining the data matrix A via $A^T = [A_1, \dots, A_n]$, we can form the Lagrangian:

$$L(x, w, \alpha) = n^{-1} \sum_{i=1}^{n} f_i(w_i) + \frac{1}{2}\sigma\|x\|^2 + \langle Ax - w, \alpha \rangle$$

and hence the dual function. We first minimize with respect to x:

$$\begin{aligned}
\nabla_x L &= \sigma x + A^T\alpha = 0 \\
x &= -\sigma^{-1} A^T \alpha \\
\min_x L &= n^{-1} \sum_{i=1}^{n} f_i(w_i) + \frac{1}{2}\sigma^{-1}\left\|A^T\alpha\right\|^2 + \left\langle -\sigma^{-1} AA^T\alpha - w, \alpha \right\rangle \\
&= n^{-1} \sum_{i=1}^{n} f_i(w_i) - \frac{1}{2}\sigma^{-1}\left\|A^T\alpha\right\|^2 + \langle -w, \alpha \rangle
\end{aligned}$$

Then we minimize with respect to w.

$$\min_w \min_x L = n^{-1} \min_{w_i} \sum_{i=1}^{n} (f_i(w_i) + \langle -w_i, n\alpha_i \rangle) - \frac{1}{2}\sigma^{-1}\left\|A^T\alpha\right\|^2$$

$$D(\alpha) = -n^{-1} \sum_{i=1}^{n} f_i^*(n\alpha_i) - \frac{1}{2}\sigma^{-1} \left\| A^T \alpha \right\|^2$$

Clearly each coordinate i of the dual function $D(\alpha)$ corresponds to function i of the primal problem. Hence coordinate methods on the dual correspond to finite-sum methods on the primal. So given a finite-sum problem, we may apply the coordinate methods of this paper to the corresponding dual problem. So our results are quite general.

CHAPTER 4

Asynchronicity

In this section we discuss definitions related to asynchronicity. Most of the analysis in this paper is done on ARock, since it is such a general algorithm.

4.1 The ARock algorithm

ARock is essentially an asynchronous-parallel block-coordinate version of KM iteration. A shared solution vector $x = (x_1, \ldots, x_m) \in \mathbb{R}^d$ is updated by a collection of p computing nodes. We let subscripts $i \in \{1, \ldots, m\}$ denote blocks of a vector, and superscripts $k \in \{0, 1, \ldots\}$ denote iteration number. At iteration k, a block $i(k)$ of the solution vector x^k is randomly chosen. This block is then updated with a KM style iteration, and the other blocks are left unchanged. Let $S = I - T$ and $Sx = (S_1 x, \ldots, S_m x)$ where $S_j x$ denotes the j'th block of Sx.

Definition 4. The ARock Algorithm. Let $\eta^k \in \mathbb{R}$ be a series of step lengths and $i(k) \in \{1, \ldots, m\}$ be a series of block indices. Let T be a nonexpansive operator with at least one fixed point x^*, and $S = I - T$. Then the ARock algorithm (Z. Peng et al. 2016) is defined via the iteration:

$$\text{for } i = 1, \ldots, m, \quad x_i^{k+1} \leftarrow \begin{cases} x_i^k - \eta^k S_i(\hat{x}^k), & i = i(k), \\ x_i^k, & i \neq i(k), \end{cases} \tag{4.1.1}$$

Here $\hat{x}^k$ is the **delayed iterate**, which represents a possibly outdated version of the iteration vector x^k used to make an update. This will be defined shortly.

Much like in Section 3.4, different choices of T lead to different asynchronous algorithms. Since ARock is an asynchronous randomized block-coordinate algorithm, T_{GD} leads to

asynchronous RBCD, T_{FB} leads to asynchronous proximal RBCD/ forward backward, etc. If the fixed-point framework is unfamiliar, it may be helpful to mentally replace S with $\gamma\nabla f$, and view ARock as asynchronous RBCD.

4.2 Setup and assumptions

In this section, we describe the delayed iterates, and the block index more precisely. We define a series of **delay vectors** $\vec{j}(0), \vec{j}(1), \vec{j}(2), \ldots$ in $\mathbb{N}^m$, corresponding to $x^0, x^1, x^2, \ldots$ respectively. The components of the delay vector $\vec{j}(k) = (j(k,1), j(k,2), \ldots, j(k,m))$ represent the staleness of the components of the solution vector x^k. Hence we define the delayed iterate as:

$$\hat{x}^k = \left(\hat{x}_1^k, \hat{x}_2^k, \ldots, \hat{x}_m^k\right) = \left(x_1^{k-j(k,1)}, x_2^{k-j(k,2)}, \ldots, x_m^{k-j(k,m)}\right). \tag{4.2.1}$$

We define the **current delay** $j(k)$ as $j(k) = \max_i\{j(k,i)\}$. So bounded delay corresponds to assuming $j(k) \leq \tau$ for some $\tau < \infty$. The delay vectors depend on the model of asynchronicity chosen. To simplify notation, for $\vec{j} \in \mathbb{N}^m$ it becomes convenient to define: $x^{k-\vec{j}} = \left(x^{k-j_1}, x^{k-j_2}, \ldots, x^{k-j_m}\right)$. And hence we have: $x^{k-\vec{j}(k)} = \hat{x}^k$.

We find it convenient to define $X^k = \left(x^0, \ldots, x^k\right)$ and $J^k = \left(\vec{j}(0), \vec{j}(1), \ldots, \vec{j}(k)\right)$. $\sigma(a, b, c, \ldots)$ represents the sigma algebra generated by $a, b, c, \ldots$. Throughout this paper unless stated otherwise, we make the following assumption about the block sequence $i(k)$.

Assumption 1. IID block sequence. $i(k)$, is a series of uniform[1] IID random variables that takes values $1, 2, \ldots, m$ each with probability $1/m$. $i(k)$ is independent of $\sigma\left(X^k, J^k\right)$. That is, $i(k)$ is independent of previous delays and iterates jointly.

Only a few papers that we are aware of make progress in eliminating this assumption that $i(k)$ is independent of the delays (T. Sun, Hannah, and Wotao Yin 2017; Leblond, Pedregosa, and Lacoste-Julien 2017; Cannelli et al. 2017). The assumption may be necessary for good convergence rates. Also removing the assumption of an IID random block sequence

[1]Nonuniform probabilities are a simple extension. However for simplicity we assume a uniform distribution.

is problematic. A cyclic choice as in (R. Sun and Ye 2016; Chow, T. Wu, and W. Yin 2017) leads to at least an m-times slowdown of the algorithm in the worst case for smooth minimization (R. Sun and Ye 2016). The block sequence will be IID if we allow computing nodes to randomly update any block chosen in a uniform IID fashion, and updating each block is of equal computational difficulty. Future work may involve finding an intermediate scenarios between IID and cyclic block choices that still results in adequate rates.

4.3 Faster Iterations + Same Quality = Faster Algorithms

This article will prove several results that comprise evidence that asynchronous algorithms will drastically outperform synchronous ones at scale. Our argument involves a series of steps, each backed up with several theoretical results. This argument and corresponding results was first presented in (Hannah and Wotao Yin 2017a).

1. **Faster iterations:** We show that asynchronous algorithms complete much faster iterations. In particular, modeling the iteration time as a renewal process with random delays, we show that asynchronous algorithms complete iterations effectively $\Theta(\ln(p))$ times faster than their synchronous counterparts.

2. **Same quality:** We show that a wide variety of asynchronous algorithms have essentially the same iteration complexity as their synchronous counterparts. Surprisingly, this is true even in the presence of potentially unbounded delays in information, so long as the delays are not too large on average.

Taking these two facts together, we can conclude that a large variety of asynchronous algorithms will drastically outperform their synchronous counterparts on large-scale applications.

4.4 Asynchronicity error

The error $\left\|x^k - x^*\right\|^2$ for ARock is not monotonic because of the asynchronicity. As in (Hannah and Wotao Yin 2017b), following on from (Z. Peng et al. 2016), the authors propose an **asynchronicity error** term to add to the classical error:

$$\underbrace{\xi^k}_{\text{Total error}} = \underbrace{\left\|x^k - x^*\right\|^2}_{\text{Classical error}} + \underbrace{\frac{1}{m}\sum_{i=1}^{\infty} c_i \left\|x^{k+1-i} - x^{k-i}\right\|^2}_{\text{Asynchronicity error}} \tag{4.4.1}$$

Here $(c_1, c_2, \ldots)$ is a decreasing sequence of positive coefficients. This Lyapunov function is actually monotonic in expectation for carefully chosen coefficients and step size for ARock. See (Hannah and Wotao Yin 2017b) for discussion of why ξ^k is the most natural error to consider when proving convergence. Choosing the coefficients that yields strong convergence results is highly nontrivial, and is part of the technical innovation of this paper. The coefficients will depends on the problem parameters, such a condition number, the number of coordinates; and also the characteristics of the delays.

4.5 General strategy for constructing Lyapunov Functions

The general strategy for building Lyapunov functions is as follows. This has wide applicability in optimization, and not just asynchronous algorithms.

Remark 1. General Strategy.

1. Let ξ^k initially be the classical error $\left\|x^{k+1} - x^*\right\|^2$ (or $f(x^k) - f(x^*)$, or similar). We will adaptively change ξ, until we have a useful Lyapunov function. Calculate the expectation of the classical error $\mathbb{E}\left[\left\|x^{k+1}\right\|^2 \middle| \sigma\left(X^k, J^k\right)\right]$ and take inequalities.

2. If this produces residual terms (for instance $\left\|x^{k+1-i} - x^{k-i}\right\|^2$) that we cannot eliminate, add a general linear combination of these terms to ξ^k. In this case, we add $\frac{1}{m}\sum_{i=1}^{\infty} c_i \left\|x^{k+1-i} - x^{k-i}\right\|^2$ to obtain the the asynchronicity error shown above.

3. Repeat steps 1 and 2 until we gain "closure". I.e. The *positive terms* in the inequality for $\mathbb{E}\left[\xi^{k+1} \middle| \sigma\left(X^k, J^k\right)\right]$ are the same as the terms found in ξ^k.

4. *Negative terms* are not problematic because they serve to decrease the expectation of ξ^{k+1}. They should not be eliminated because they can give useful information.

5. Vary the coefficients of the Lyapunov function to enable a useful comparison between $\mathbb{E}\left[\xi^{k+1}\middle|\sigma\left(X^k, J^k\right)\right]$ and ξ^k.

Which inequalities to take and which residual terms to create is a matter of trial and error. Some choices lead to dead ends, whereas others lead to a viable proof.

Part II

Faster Iterations, Same Quality

In this part, we present the results of (Hannah and Wotao Yin 2017a).

CHAPTER 5

Faster Iterations

In this section, we model how asynchronous algorithms complete faster iterations using renewal processes. Though the theoretical arguments are rather simple, we choose to include this because it is an important justification for asynchronous algorithms' utility.

5.1 Implementation setup

We consider a system of p computing nodes that update a shared solution vector x which is stored at a central server. The computing nodes are first sent the solution vector which is then stored in their local memory as $\hat{x}$. Then they randomly chose a block i to update, calculate an update $S_i\hat{x}$, and send this update to the server. The server receives and applies these updates as they arrive via: $x \leftarrow x - \eta S_i\hat{x}$.

We now compare and contrast the synchronous and asynchronous version of ARock. In the synchronous implementation, at every iteration, the server sends each computing node the same vector x. Only when the updates from every node are computed, received and applied by the server can the next iteration begin. Hence the lateness of even one computing node will prevent the server from sending the latest solution vector to the computing nodes so they can compute the next update.

In the asynchronous implementation, computing nodes will be sent the latest solution vector as soon as they are done computing their latest update. They may also compute multiple updates with the same solution vector to reduce bandwidth. Hence computing nodes act independently without any central coordination, and without waiting for other nodes to complete their updates. The server will simply apply updates as they are received.

Many updates may occur between the time a node is sent the solution vector and the time the computed update is applied. Therefore effectively, the solution vector is updated with outdated information.

5.2 Iteration time model

Following (Serpedin and Chaudhari 2009) (p. 43) and adding modifications, we model the time taken for node l's update as follows:

$$P_l = R_l + C(l, i_l, m) + S_l.$$

Here i_l is the block of the solution vector that node l updates. $C(l, i, m)$ represents the non-random portion of the update time. This includes computation time, and the time delay to send and received vectors over the network due to bandwidth limitations. $C(l, i, m)$ is a function of i because different blocks may have different sizes and update difficulties. C_l varies with computing node l because different nodes have different characteristics such as computing power. R_l is the random delay involved in receiving the solution vector from the central server, and S_l is the random delay involved in sending an update back to the server. R_l and S_l are assumed IID with exponential distribution of mean λ_l. This exponential model for the random portion of the delay has extensive theoretical and empirical justifications (see (Serpedin and Chaudhari 2009), pp. 44-45 for a discussion of the evidence).

In the following sections, we examine critical factors that result in asynchronous algorithms completing faster iterations. For simplicity, we consider each factor in isolation.

5.3 The effect of random delays

We first consider the effect of random delays on the synchronization penalty. For simplicity, assume that $C(l, i, m)$ is constant over i and l, and hence we can write this function as $C(1, 1, m)$. Also assume we have $\lambda_l = \lambda$ for all l. This situation would occur if all blocks were of equal difficulty to update, and all nodes had the same computational power and network delay distribution. This is the ideal scenario, and yet we will observe a growing

synchronization penalty with scale.

Because all nodes must finish updating for the next iteration to start, the iteration time P for the synchronous system is given by:

$$P = C(1,1,m) + \max_{l=1,2,\ldots,p}\{R_l + S_l\}. \tag{5.3.1}$$

Hence we have (using (Eisenberg 2008)):

$$\mathbb{E}P - C(1,1,m) \geq \mathbb{E}\Big(\max_{l=1,\ldots,p}\{R_l\}\Big) = \lambda\sum_{l=1}^{p}\frac{1}{l} \geq \lambda\ln(p).$$

Now consider the time $\mathcal{T}_{\text{Sync}}(K)$ required for K epochs, which corresponds to $\lceil Km/p\rceil$ iterations. This is because each step of synchronous ARock requires p evaluations of $S_i x$, which in total is p/m of the computational effort for a full evaluation of Sx. Let[1] $P^1, P^2, \ldots \sim P$:

$$\mathbb{E}\mathcal{T}_{\text{Sync}}(K) = \mathbb{E}\sum_{k=1}^{\lceil Km/p\rceil} P^k \geq \frac{Km}{p}\mathbb{E}P \geq \frac{Km}{p}(C(1,1,m) + \lambda\ln(p)).$$

Hence for small values of p, the expected time to reach K epochs will decrease inverse linearly with the number of nodes p. However as p becomes larger, there is at least a $\Theta(\ln(p))$ penalty in how long this will take compared to a linear speedup.

Using the same model, we now show that asynchronous algorithms have no such $\Theta(\ln(p))$ scaling penalty. The time taken for node l to complete k iterations is given by:

$$S_l^k = \sum_{j=1}^{k} P_l^j \tag{5.3.2}$$

where $P_l^k \sim P$. This is actually a **renewal process** with **interarrival time** P_l (see (Mitov and Omey 2014; Kella and Stadje 2006)). However if we consider the total number of iterations completed by all nodes together, then the time of the k'th iteration S^k is known as a **superposition of renewal processes**. From (Kella and Stadje 2006) 1.4, as $k \to \infty$ we have:

$$\frac{\mathbb{E}S^k}{k} \to \frac{\mathbb{E}P}{p}, \tag{5.3.3}$$

[1]We write $A \sim B$ for random variables A and B if these variables have the same distribution.

(by convergence in the previous step) $$\mathbb{E}S^k = k\frac{(C(1,1,m)+2\lambda)}{p}(1+o_{m,\lambda,p}(1)). \quad (5.3.4)$$

The subscripts in $o_{m,\lambda,p}(1)$ denote that this term converges to 0 as $k \to \infty$ in a way that depends on m, p, and λ. Hence the expected time to complete K epochs is given by:

$$\mathbb{E}\mathcal{T}_{\text{Async}}(K) = \frac{Km}{p}(C(1,1,m)+2\lambda)(1+o_{m,\lambda,p}(1)) \quad (5.3.5)$$

as $K \to \infty$. Hence it can be seen that asynchronous algorithms do not have a $\ln(p)$ penalty. Hence for large enough K, asynchronous algorithms will compute at least $\Theta(\ln(p))$ more epochs per second than synchronous algorithms.

5.4 Heterogeneous update difficulty

Sometimes a parallel problem *cannot* be split into m blocks in a way that updating each block is of equal difficulty (as was previously assumed in this subsection). This can cause significant synchronization penalty in the synchronous case, but has no such effect on asynchronous algorithms because computing nodes do not have to wait for slower nodes or blocks to complete.

Let us assume for the moment that there is no random component of the update time for a single node, and that all nodes have the same computational power. This means that the update time for node l at iteration k is simply:

$$P_l^k = C(1, i_l, m) \quad (5.4.1)$$

where i_l is the block that node l updates at iteration k. Assume also that at every iteration, each node l will chose a uniformly random block to update, and hence i_l is a uniform random variable on $\{1, 2, \ldots, m\}$. For the synchronous algorithm, we have an update time:

$$P = \max_{l=1,2,\ldots,p}\{C(1, i_l, m)\} \quad (5.4.2)$$

Clearly then, as p increases, we have:

$$\mathbb{E}P \to \max_i C(1, i, m) \quad (5.4.3)$$

That is, the update time is determined by the most difficult block to update. Hence the expected time for K synchronous epochs is:

$$\mathbb{E}\mathcal{T}_{\text{Sync}}(K) \sim \frac{Km}{p}\left(\max_i C(i,m) + o_m(1)\right) \tag{5.4.4}$$

as $p \to \infty$.

Now consider an asynchronous algorithm. The update time of a single node is:

$$\mathbb{E}P = \mathbb{E}C(1, i_l, m) = \frac{1}{m}\sum_{i=1}^{m} C(1,i,m) \tag{5.4.5}$$

Yet again we have a superposition of renewal processes, and hence from (Kella and Stadje 2006), we have as $k \to \infty$:

$$\frac{\mathbb{E}S^k}{k} \to \frac{\mathbb{E}P}{p} \tag{5.4.6}$$

$$\mathbb{E}S^k = \frac{k}{p}\left(\frac{1}{m}\sum_{i=1}^{m} C(1,i,m)\right)(1 + o_{m,p}(1)) \tag{5.4.7}$$

Hence the expected time for K epochs is given by:

$$\mathbb{E}\mathcal{T}_{\text{Async}}(K) = \frac{Km}{p}\left(\frac{1}{m}\sum_{i=1}^{m} C(1,i,m)\right)(1 + o_{m,p}(1)) \tag{5.4.8}$$

as $K \to \infty$. Notice that the time taken for an asynchronous algorithm is determined by the *average difficulty of updating a block*. Compare this to synchronous algorithms where the *most difficult block* determines the time complexity. If the difficulty of blocks is highly heterogeneous, asynchronous algorithms may complete far faster iterations, even without considering network effects.

5.5 Heterogeneous computing node power

If the computing nodes have very different computing powers, the faster nodes in synchronous systems will always have to wait for slower nodes. Assume for the moment that blocks are of equal difficulty, and there is no network delay. The update time for node l is then given by:

$$P_l = C(l, 1, m) \tag{5.5.1}$$

Hence the time for K synchronous epochs (which is Km/p iterations) is given by:

$$\mathcal{T}_{\text{Sync}}(K) = \frac{Km}{p}\left(\max_{l=1,2,\ldots,p} C(l,1,m)\right) \tag{5.5.2}$$

On the other hand, for asynchronous systems, the time for K epochs (which corresponds to Km iterations) is again a superposition of renewal processes. We have (by (Kella and Stadje 2006), 1.3):

$$\mathcal{T}_{\text{Async}}(K) = S^{Km} = \frac{Km}{p}\left(\frac{1}{p}\sum_{l=1}^{p} C(l,1,m)^{-1}\right)^{-1}(1 + o_{m,\lambda,p}(1)) \tag{5.5.3}$$

Hence we can see that for synchronous algorithms, this time depends on the *power of the weakest computing node*. Whereas for asynchronous algorithms, the time depends on the *average computing power* of the nodes.

5.6 Summary

Hence for several reasons, asynchronous algorithms will completely significantly more iterations in the same time period as compared to synchronous algorithms. Most of the rest of this paper focuses on proving that in several situations these iterations make essentially the same progress towards a solution.

CHAPTER 6

Sharp Iteration Complexity for Synchronous Algorithms

In this section we prove sharp iteration complexity results for some synchronous algorithms. We need sharp rates and complexities in order to make a fair comparison between synchronous and asynchronous ARock. Most authors derive *upper bounds* on rates instead, which do not give a true measure of an algorithm's performance. We prove results only for ARock, RBCD, and proximal RBCD. Similar sharp complexity results along the same lines are possible for the other special cases, such as Douglass-Rachford, proximal point, etc. In the interests of space, we do not prove an exhaustive list. We will show later that synchronous and asynchronous ARock have the same complexity asymptotically, despite the use of outdated information.

An algorithm is said to **linearly converge** if the error $\mathbb{E}E(k) = \mathcal{O}\left(R^k\right)$ for $0 < R < 1$. R is called the **linear convergence rate**. The **epoch complexity** $I(\epsilon)$ is the number of epochs required to decrease the error $\mathbb{E}E(k)$ below $\epsilon E(0)$, where $E(0)$ is the initial error. This error could be the distance from the solution $\left\|x^k - x^*\right\|^2$, the suboptimality $f(x^k) - f^*$, etc.

Consider an algorithm A that solves a problem class P. Define $\bar{R}_A(f)$ as the smallest linear convergence rate that A attains for a specific problem instance $f \in P$. That is, $\bar{R}_A(f) \triangleq \inf\left\{R | E(k) = \mathcal{O}\left(R^k\right)\right\}$. If an algorithm converges with rate R, this may be an overestimate, whereas $\bar{R}_A(f)$ can be viewed as the true speed of convergence. The **sharp convergence rate** $R_A(P)$ of algorithm A over problem class P is defined as the largest convergence rate $\bar{R}_A(f)$ that A attains for any $f \in P$. That is:

$$R_A(P) \triangleq \max_{f \in P} \bar{R}_A(f) \tag{6.0.1}$$

This the worst case convergence rate of an algorithm A over the problem class. In other words, it is the best convergence rate the can be guaranteed by the algorithm A over this class. For instance, consider gradient descent on the class of L-smooth, μ-strongly convex functions f. The sharp convergence rate $R_A(\mu, L)$ is defined as:

$$R_A(\mu, L) = \max_{\text{f is } L\text{-smooth, } \mu\text{-strongly convex}} \bar{R}_A(f) \tag{6.0.2}$$

Clearly the sharp rate is a function of μ and L. The **sharp epoch complexity** is defined in a similar way. It is the smallest epoch complexity that can be guaranteed for an algorithm A over a problem class P.

6.1 Synchronous ARock

First we define synchronous ARock. At every step, each of the p computing nodes are given a random block to update with a KM-style iteration. Hence we have:

$$x^{k+1} = x^k - \eta^k P^k S x^k \tag{6.1.1}$$

Here P^k is a projection onto a random subset of $\{1, 2, \ldots, m\}$ of size p (we assume $p \leq m$). We note that each block has a $\frac{p}{m}$ probability of being updated on a given iteration.

Proposition 5. Convergence rate of synchronous KM iterations. *Let T be an r-Lipschitz operator for $0 < r < 1$. Consider the synchronous KM iteration defined in Equation (6.1.1) for $1 \leq p \leq m$. The sharp convergence rate is given by:*

$$R_{KM}(\eta, r) = 1 - \frac{p}{m} + \frac{p}{m}(\max\{|1 - \eta(1 - r)|, |1 - \eta(1 + r)|\})^2 \tag{6.1.2}$$

$\eta^k = 1$ optimizes this rate. This step size yields the following optimal convergence rate (6.1.3), and optimal iteration complexity (6.1.4) respectively:

$$R = 1 - \frac{p}{m}\left(1 - r^2\right) \tag{6.1.3}$$

$$I(\epsilon) = \left(\frac{1}{1 - r^2} - \theta\frac{p}{m}\right)\ln(1/\epsilon) \tag{6.1.4}$$

Here $\theta \in [1/2, 1]$. Lastly this rate and complexity are sharp, and occur for at least 1 operator.

For problems of interest, the first term $1/(1-r^2)$ will usually dominate the second. We are interested in huge-scale problems, which will usually have $m \gg p$ or $r \approx 1$. Hence if we have either $r \to 1$ or $p/m \to 0$, then:

$$I(\epsilon) = (1+o(1))\frac{1}{1-r^2}\ln(1/\epsilon) \tag{6.1.5}$$

We will eventually prove that ARock has essentially the same iteration complexity. It becomes convenient later to define

$$I_{\text{Sync}}(\epsilon) = \frac{1}{1-r^2}\ln(1/\epsilon) \tag{6.1.6}$$

which is the sharp complexity of synchronous ARock to highest order.

Proof of Proposition 5. Taking conditional expectation on the following

$$\left\|x^{k+1}\right\|^2 = \left\|x^k\right\|^2 - 2\eta^k\left\langle x^k, P^kSx^k\right\rangle + \left(\eta^k\right)^2\left\|P^kSx^k\right\|^2$$

with respect to $i(k)$ yields

$$\begin{aligned}
\mathbb{E}\left[\left\|x^{k+1}\right\|^2\middle|x^k\right] &= \left\|x^k\right\|^2 - 2\eta^k\frac{p}{m}\left\langle x^k, Sx^k\right\rangle + \left(\eta^k\right)^2\frac{p}{m}\left\|Sx^k\right\|^2 \\
\text{(By Lemma 3)} &\le \left\|x^k\right\|^2 - \eta^k\frac{p}{m}\left(\left\|Sx^k\right\|^2 + \left(1-r^2\right)\left\|x^k\right\|^2\right) + \left(\eta^k\right)^2\frac{p}{m}\left\|Sx^k\right\|^2 \\
&= \left(1-\eta^k\frac{p}{m}\left(1-r^2\right)\right)\left\|x^k\right\|^2 - \eta^k\frac{p}{m}\left(1-\eta^k\right)\left\|Sx^k\right\|^2,
\end{aligned} \tag{6.1.7}$$

When $\eta^k \le 1$, (6.1.7) yields:

$$\begin{aligned}
\text{(By } (1-r)\text{-strong monotonicity of } S) &\le \left(1-\eta^k\frac{p}{m}\left(1-r^2\right)\right)\left\|x^k\right\|^2 - (1-r)^2\eta^k\frac{p}{m}\left(1-\eta^k\right)\left\|x^k\right\|^2 \\
&= \left(1-\frac{p}{m}+\frac{p}{m}\left(1-\eta^k(1-r)\right)^2\right)\left\|x^k\right\|^2.
\end{aligned} \tag{6.1.8}$$

Now when $\eta^k \ge 1$, (6.1.7) yields:

$$\mathbb{E}\left[\left\|x^{k+1}\right\|^2\middle|x^k\right] \tag{6.1.9}$$

$$\begin{aligned}
(S \text{ is } (1+r)\text{-Lipschitz}) &\le \left(1-\eta^k\frac{p}{m}\left(1-r^2\right)\right)\left\|x^k\right\|^2 - \eta^k\frac{p}{m}\left(1-\eta^k\right)(1+r)^2\left\|x^k\right\|^2 \\
&= \left(1-\frac{p}{m}+\frac{p}{m}\left(1-\eta^k(1+r)\right)^2\right)\left\|x^k\right\|^2.
\end{aligned} \tag{6.1.10}$$

It can be verified that every single inequality for $\eta^k \leq 1$ is an equality for $T = rI$, and every single inequality for $\eta^k \geq 1$ is an equality for $T = -rI$. Therefore the inequalities (6.1.8) and (6.1.10) give a sharp rate of convergence for $\eta^k \leq 1$ and $\eta^k > 1$ respectively. These expressions match (6.1.2). This rate is clearly optimized when $\eta^k = 1$, and matches the rate given in Equation (6.1.3).

Let's now look at the corresponding iteration complexity. $\frac{m}{p}$ iterations correspond to 1 epoch, hence:

$$\begin{aligned}
\epsilon &= \left(1 - \frac{p}{m}\left(1 - r^2\right)\right)^{I(\epsilon)\frac{m}{p}} \\
I(\epsilon) &= \frac{p}{m}\left(\frac{\ln(1/\epsilon)}{-\ln\left(1 - \frac{p}{m}(1 - r^2)\right)}\right) \\
&= \left(\frac{1}{1 - r^2} - \theta\frac{p}{m}\right)\ln(1/\epsilon)
\end{aligned}$$

where $\theta \in \left[\frac{1}{2}, 1\right]$. The last line follows from the inequality:

$$1 - x \leq \frac{-x}{\ln(1 - x)} \leq 1 - \frac{1}{2}x, \text{ for } 0 \leq x \leq 1$$

for $x = \frac{p}{m}(1 - r^2)$. The derived complexity matches Equation (6.1.4), hence the proof is complete. □

6.2 Sharp Complexity Results for RBCD

Proposition 5 allows us to obtain a sharp convergence rate and epoch complexity for synchronous RBCD. Define the operator $T_{\text{GD}} = I - \frac{2}{\mu+L}\nabla f$. Clearly the synchronous KM in (6.1.1) is equivalent to synchronous RBCD with step size size $\frac{2\eta}{\mu+L}$.

Corollary 6. Sharp Convergence Rate of Synchronous RBCD. *The optimal step size for synchronous RBCD is $\frac{2}{\mu+L}$. This step size yields the following optimal convergence rate (6.2.1), and optimal iteration complexity (6.2.2) respectively:*

$$R = 1 - 4\frac{p}{m}\frac{\kappa}{(\kappa+1)^2} \qquad = 1 - 4\frac{p}{m\kappa}(1 + \mathcal{O}(1/\kappa)) \tag{6.2.1}$$

$$I(\epsilon) = \frac{1}{4}(\kappa + \mathcal{O}(1))\ln(1/\epsilon) \tag{6.2.2}$$

as $\kappa \to \infty$. Lastly, this convergence rate is sharp, and occurs for at least 1 function.

Among other things, we show that asynchronous RBCD has epoch iteration complexity that is asymptotically equal to $\frac{1}{4}\kappa \ln(1/\epsilon)$, which is the complexity of synchronous RBCD. This appears to be a new result that extends recent work in (Taylor, Hendrickx, and Glineur 2018) which proved the special case of $m = 1$, $p = 1$. Standard RBCD corresponds to $p = 1$.

Proof of Corollary 6. We consider the operator $T_{\text{GD}} = I - \gamma \nabla f$. The corresponding synchronous KM iteration is synchronous RBCD:

$$x^{k+1} = x^k - \eta\gamma P^k \nabla f\left(x^k\right)$$

Let's let the sharp rate of convergence be denoted by $R_{\text{GD}}(\eta, \gamma)$. Clearly for any $\lambda > 0$, we have: $R_{\text{GD}}(\eta, \gamma) = R_{\text{GD}}(\eta\lambda, \gamma/\lambda)$. Let $\lambda = \gamma(L + \mu)/2$. So we need only calculate $R_{\text{GD}}\left(\eta\left(\gamma/\frac{2}{L+\mu}\right), \frac{2}{L+\mu}\right)$ to determine $R_{\text{GD}}(\eta, \gamma)$. Hence we need only consider the operator $T_{\text{GD}} = I - \frac{2}{L+\mu}\nabla f$, or equivalently, the case when $\gamma = \frac{2}{L+\mu}$. For $T_{\text{GD}} = I - \frac{2}{L+\mu}\nabla f$, we have (recalling Thm. 2.1.12 of (Yurii Nesterov 2013)):

$$\begin{aligned}
\|T(y) - T(x)\|^2 &= \|y - x\|^2 - \frac{4}{\mu + L}\langle \nabla f(y) - \nabla f(x), y - x\rangle + \left(\frac{2}{L+\mu}\right)^2 \|\nabla f(y) - \nabla f(x)\|^2 \\
&\leq \|y - x\|^2 - \frac{4}{\mu + L}\left(\frac{\mu L}{\mu + L}\|x - y\|^2 + \frac{1}{\mu + L}\|\nabla f(y) - \nabla f(x)\|^2\right) \\
&+ \left(\frac{2}{L+\mu}\right)^2 \|\nabla f(y) - \nabla f(x)\|^2 \\
&= \|y - x\|^2\left(1 - \frac{4\mu L}{(\mu + L)^2}\right) = \|y - x\|^2\left(1 - \frac{4\kappa}{(1+\kappa)^2}\right) \\
&= \|y - x\|^2\left(1 - \frac{2}{\kappa + 1}\right)^2
\end{aligned}$$

Hence T_{GD} is r-Lipschitz for $r = 1 - \frac{2}{\kappa+1}$. This combined with the fact that synchronous RBCD is a special case of synchronous KM implies: $R_{\text{GD}}\left(\eta, \frac{2}{L+\mu}\right) \leq R_{\text{KM}}\left(\eta, 1 - \frac{2}{\kappa+1}\right)$.

On the other hand, $f = \frac{1}{2}\mu\|x\|^2$ yields $T_{\text{GD}} = rI$ and $f = \frac{1}{2}L\|x\|^2$ yields $T_{\text{GD}} = -rI$. That is, the worst-case examples of Proposition 5 are attainable for any step size η. Hence $R_{\text{GD}}\left(\eta, \frac{2}{L+\mu}\right) = R_{\text{KM}}\left(\eta, 1 - \frac{2}{\kappa+1}\right)$. This allows us to use (6.1.2) to determine the sharp rate

of convergence for synchronous gradient descent for step size $\frac{2\eta}{L+\mu}$:

$$R_{\mathrm{KM}}\left(\eta, 1-\frac{2}{\kappa+1}\right)=1-\frac{p}{m}+\frac{p}{m}\left(\max\left\{\left|1-\eta\frac{2}{\kappa+1}\right|,\left|1-\eta\frac{2\kappa}{\kappa+1}\right|\right\}\right)^2 \tag{6.2.3}$$

As before, this rate is optimized when $\eta=1$. With this step size, we have the corresponding optimal convergence rate:

$$R=1-\frac{p}{m}\left(1-r^2\right)=1-\frac{p}{m}\frac{4\kappa}{(\kappa+1)^2}=1-4\frac{p}{m\kappa}(1+\mathcal{O}(1/\kappa))$$

which matches Equation (6.2.1).

This allows us to then determine the optimal iteration complexity:

$$\begin{aligned}I(\epsilon)&=\left(\frac{1}{1-r^2}-\theta\frac{p}{m}\right)\ln(1/\epsilon)=\left(\frac{(\kappa+1)^2}{4\kappa}-\theta\frac{p}{m}\right)\ln(1/\epsilon)=\frac{1}{4}\left(\kappa+2+\frac{1}{\kappa}-4\theta\frac{p}{m}\right)\ln(1/\epsilon)\\&=\frac{1}{4}(\kappa+\mathcal{O}(1))\ln(1/\epsilon)\end{aligned}$$

This matches Equation (6.2.2). Hence the results are proven. □

Clearly other sharp complexity results easily follow from Proposition 5. Consider the objective:

$$F(x)=f(x_1,\ldots,x_m)+\sum_{i=1}^{n}g(x_i) \tag{6.2.4}$$

where each g_i is convex and subdifferentiable. Consider the synchronous proximal RBCD algorithm given by:

$$x^{k+1}=\left(1-\eta^k\right)x^k+\eta^k P^k(I+\gamma\partial g)^{-1}\circ(I-\gamma\nabla f)\left(x^k\right) \tag{6.2.5}$$

The optimal parameter choice yields the same complexity as synchronous RBCD.

Corollary 7. Sharp convergence rate for synchronous proximal RBCD. *The synchronous proximal RBCD algorithm defined in (6.2.5) has an optimal parameter choice of $\gamma=\frac{2}{L+\mu}$, $\eta^k=1$. This choice yields an optimal convergence rate of (6.1.3), and an optimal complexity of (6.1.4).*

Proof of Corollary 7. Consider the operator $T_{\mathrm{FB}} = (I + \gamma\partial g)^{-1} \circ (I - \gamma\nabla f)$. Let the sharp convergence rate be denoted as $R_{\mathrm{FB}}(\eta, \gamma)$. When $g = 0$, T_{FB} reduces to T_{GD} from before. Hence synchronous RBCD is a special case of synchronous proximal RBCD, and we have $R_{\mathrm{FB}}(\eta, \gamma) \geq R_{\mathrm{GD}}(\eta, \gamma)$. If we let $\gamma = \frac{2}{L+\mu}$ and $\eta = 1$, then T_{FB} is $\left(1 - \frac{2}{\kappa+1}\right)$-Lipschitz since $(I + \gamma\partial g)^{-1}$ is nonexpansive. Since synchronous proximal forward backward is a special case of synchronous KM, we have $R_{\mathrm{KM}}\left(1, 1 - \frac{2}{\kappa+1}\right) \geq R_{\mathrm{FB}}\left(1, \frac{2}{L+\mu}\right)$. However from the previous proof, we also have $R_{\mathrm{KM}}\left(1, 1 - \frac{2}{\kappa+1}\right) = R_{\mathrm{GD}}\left(1, \frac{2}{L+\mu}\right)$, which is the optimal rate for synchronous RBCD. Putting these facts together, we conclude that $\gamma = \frac{2}{L+\mu}$ and $\eta = 1$ are also the optimal parameter choices for synchronous proximal RBCD, and lead to the same optimal convergence rate and optimal complexity. □

CHAPTER 7

Same Quality: Stochastic unbounded delays

In this section, we bolster part 2 of the main argument in Section 4.3. We have just derived the sharp convergence rate and complexity for synchronous ARock. We now prove that ARock, and hence all of its special cases, has essentially the same complexity as synchronous ARock – even under unbounded delays.

7.1 Main result

In this section we consider stochastic unbounded delays.

Assumption 2. Stochastic unbounded delays. The sequence of delay vectors $\vec{j}(0), \vec{j}(1), \vec{j}(2), \ldots$ is an independent sequence of random variables, and $\vec{j}(k)$ is independent of $\left(x^0, x^1, x^2, \ldots, x^k\right)$.

This assumption is very general. The distribution of delays may change each iteration in any way that is independent of $\sigma(X^k)$. This assumption includes deterministic bounded delay. This corresponds to $j(k)$ being a trivial random variable that takes 1 value with probability 1.

The convergence rate will depend on the distribution of the delays. Distributions with larger delays in general will lead to worse performance. Let $P_l \in [0,1]$ be constants such that:

$$P_l \geq \mathbb{P}\Big[j(k) \geq l \Big| \sigma\left(X^k\right)\Big], \forall k \tag{7.1.1}$$

We let ρ be defined by:

$$\rho = 1 - \frac{1}{m}\left(1 - r^2\right) \tag{7.1.2}$$

This is the sharp linear convergence rate for synchronous ARock with $p = 1$, and optimal step size. This is the best best rate we can hope to achieve. We also define the probability moment:

$$M = \sum_{l=1}^{\infty} P_l^{1/2} \rho^{-l/2} \tag{7.1.3}$$

This moment quantifies how large the delay is, similar to τ. In the case of deterministic bounded delay, it is easy to show that $\tau \leq M \leq e\tau$ for $\tau \leq m$.

We also define the asynchronicity parameter:

$$\psi = 5Mm^{-1/2}$$

This quantifies how strongly asynchronicity will affect convergence. The larger ψ is, the worse the complexity will be, and the more conservative a step size we must take. For the special case of deterministic bounded delay, we have $\psi \leq 5e\tau m^{-1/2}$. Using ψ, we define the step size:

$$\eta_1 = \left(1 + \frac{3}{5}\psi\right)^{-1} \tag{7.1.4}$$

We state our results in terms of the Lyapunov function ξ^k discussed in Chapter 4:

$$\underbrace{\xi^k}_{\text{Total error}} = \underbrace{\left\|x^k - x^*\right\|^2}_{\text{Classical error}} + \underbrace{\frac{1}{m}\sum_{i=1}^{\infty} c_i \left\|x^{k+1-i} - x^{k-i}\right\|^2}_{\text{Asynchronicity error}}$$

Theorem 2. Linear convergence for stochastic delays. *Let Assumption 1 and Assumption 2 hold. Let the Lyapunov function coefficients be given by* $c_i = 2m^{1/2} \sum_{l=i}^{\infty} P_l^{1/2} \rho^{i-l/2-1}$. *Let* M *be finite, and* $\eta^k = \eta_1$. *Then we have the following linear convergence rate and epoch complexity respectively:*

$$\mathbb{E}\left[\xi^{k+1} \middle| \sigma\left(X^k\right)\right] \leq \left(1 - \frac{1}{m}\left(1 - r^2\right)\frac{1}{1+\psi}\right)\xi^k, \tag{7.1.5}$$

$$I_{Async}(\epsilon) \leq (1+\psi)\frac{1}{1-r^2}\ln(1/\epsilon) \tag{7.1.6}$$

$$= (1+\psi) I_{Sync}(\epsilon) \tag{7.1.7}$$

This theorem is proven in Section 7.6 after a series of results built up in this section. For $\psi \ll 1$, the complexities of asynchronous and synchronous ARock essentially become the

same. Hence ARock suffers no significant complexity penalty for using outdated information, so long as that information is not too old.

We now present some important special cases. This corollary easily follows from Theorem 2, Corollary 6, and Corollary 7.

Corollary 8. *Consider asynchronous RBCD and asynchronous proximal RBCD with step size* $h = \frac{2}{\left(1+\frac{3}{5}\psi\right)(\mu+L)}$. *Their linear convergence rate and epoch complexity are given by the following:*

$$R = 1 - \frac{4\kappa}{m(\kappa+1)^2(1+\psi)} \tag{7.1.8}$$

$$I(\epsilon) = \frac{1}{4}(\kappa + \mathcal{O}(1))(1+\psi)\ln(1/\epsilon) \tag{7.1.9}$$

When there is no asynchronicity, the Lyapunov function will reduce to the classical error $\xi^k = \left\|x^k - x^*\right\|^2$, and $\eta_1 = 1$. Also the convergence rates and iteration complexity will reduce to the sharp values obtained in Proposition 5 for $p = 1$.

When we have deterministic bounded delay, we have complexity:

$$I(\epsilon) = \frac{1}{4}(\kappa + \mathcal{O}(1))\left(1 + 5e\tau m^{-1/2}\right)\ln(1/\epsilon) \tag{7.1.10}$$

Hence in this setting synchronous and asynchronous RBCD have essentially the same complexity when $\tau \ll m^{1/2}$.

Our main results remain true for unbounded delay. Almost all work on asynchronous algorithms except for (Hannah and Wotao Yin 2017b; Zhimin Peng, Xu, et al. 2017) assumed bounded delays. However in practice, there may be no way to rule out arbitrarily large delays. So this assumption is impractical. Even when the delay is bounded, our results may imply a much better complexity, since we may have $\tau \gg M$. This would occur when very large delays can occur, but are quite rare.

We now prove Theorem 2 in a way that emphasizes the reasons and intuition behind our approach – especially the strategic way in which the coefficients are chosen.

7.2 Preliminaries

Let x^* be any solution, and set $x^* = 0$ with no loss in generality, to make the notation more compact[1]. We let the step size η^k be $\sigma\left(X^k\right)$-measurable. The starting point of our analysis is the following:

$$\begin{aligned}\mathbb{E}\left[\left\|x^{k+1}\right\|^2|\sigma\left(X^k,J^k\right)\right] &= \mathbb{E}\left[\left\|x^k-\eta^k S_{i(k)}\hat{x}^k\right\|^2|\sigma\left(X^k,J^k\right)\right]\\ &= \left\|x^k\right\|^2+\mathbb{E}\left[-2\eta^k\left\langle x^k,S_{i(k)}\hat{x}^k\right\rangle+\left(\eta^k\right)^2\left\|S_{i(k)}\hat{x}^k\right\|^2|\sigma\left(X^k,J^k\right)\right]\\ &= \left\|x^k\right\|^2\underbrace{-2\frac{\eta^k}{m}\left\langle x^k,S\hat{x}^k\right\rangle}_{\text{cross term}}+\frac{\left(\eta^k\right)^2}{m}\left\|S\hat{x}^k\right\|^2. \end{aligned} \tag{7.2.1}$$

Here the expectation is taken over only the block index $i(k)$ (Recall Assumption 1).

7.3 The cross term

We now discuss our general strategy on how to proceed from Equation (7.2.1). To obtain a linear convergence result, we first need to negate $\left\|S\hat{x}^k\right\|^2$, which can be thought of as a "waste" term. We also need to generate a $-\left\|x^k\right\|^2$ term for linear convergence. We can convert $-\left\langle S\hat{x}^k,x^k\right\rangle$ into $-\left\langle S\hat{x}^k,\hat{x}^k\right\rangle$, which produces $-\left\|S\hat{x}^k\right\|^2$ and $-\left\|\hat{x}^k\right\|^2$ terms by Lemma 3. The first term allows us to eliminate the $\left\|S\hat{x}^k\right\|^2$ waste. We then convert $-\left\|\hat{x}^k\right\|^2$ into $-\left\|x^k\right\|^2$, which is needed for linear convergence. This is the rationale behind Lemma 10 ahead.

However this process of conversion produces other "waste" terms, which can be seen in the second line of Lemma 10. We use the Lyapunov function to deal with this waste, by incorporating the waste terms directly into the error that we consider (see Section 7.4). This is the final piece of the puzzle that leads to linear convergence.

First however, we need Lemma 9 to allow us to quantify the error associated with converting $-\left\langle S\hat{x}^k,x^k\right\rangle$ to $-\left\langle S\hat{x}^k,\hat{x}^k\right\rangle$, and the error associated with converting $-\left\|\hat{x}^k\right\|^2$ to $-\left\|x^k\right\|^2$ mentioned above.

[1]This can be achieved by translating the origin of the coordinate system to x^*. Hence $\left\|x^k\right\|$ is the distance from the solution.

Lemma 9. *Let $a > 0$, $j(k)$ be the current delay, η^k be the current step size, and $\epsilon_1, \epsilon_2, \ldots > 0$ be a series of parameters. Then we have:*

$$a\left\|x^k - \hat{x}^k\right\| \leq \frac{1}{2}a^2\eta^k\left(\sum_{i=1}^{j(k)}\frac{1}{\epsilon_i}\right) + \frac{1}{2}\frac{1}{\eta^k}\sum_{i=1}^{j(k)}\left(\epsilon_i\left\|x^{k+1-i} - x^{k-i}\right\|^2\right) \tag{7.3.1}$$

Proof. See (Hannah and Wotao Yin 2017b; Z. Peng et al. 2016) for a simple proof using Cauchy-Schwarz. □

Lemma 10 generates some positive parameters $\epsilon_1, \epsilon_2, \ldots > 0$ and $\delta_1, \delta_2, \ldots > 0$. It's not immediately clear what these parameters should be set to. However we will see in Section 7.5 if they are properly chosen, they can be used to construct a Lyapunov function that will allow us to prove a fast linear convergence rate.

We will make use of Lemma 9 twice with parameter sets $(\epsilon_1, \epsilon_2, \ldots)$ and $(\delta_1, \delta_2, \ldots)$ respectively. To simplify notation, we define:

$$E_j = \sum_{i=1}^{j}\frac{1}{\epsilon_i} \qquad\qquad D_j = \sum_{i=1}^{j}\frac{1}{\delta_i} \tag{7.3.2}$$

We also define the following convergence rate function:

$$R(\eta, \gamma) = 1 - (\eta/m)\left(1 - r^2\right)(1 - \eta/\gamma) \tag{7.3.3}$$

Note that we have $R < 1$ when $0 < \eta < \gamma$. The rate is optimized when $\eta = (1/2)\gamma$. Also $\rho \leq R$ for $\eta \leq 1$.

Lemma 10. *Let Assumption 1 hold. Let $\epsilon_1, \epsilon_2, \ldots > 0$ and $\delta_1, \delta_2, \ldots > 0$ be a sequence of parameters. Let η^k be $\sigma\left(X^k\right)$-measurable. ARock yields the following inequality:*

$$\begin{aligned}\mathbb{E}\left[\left\|x^{k+1}\right\|^2\middle|\sigma\left(X^k, J^k\right)\right] \leq R(\eta^k, 1/D_{j(k)})\left\|x^k\right\|^2 - \frac{\eta^k}{m}\left\|S\hat{x}^k\right\|^2\left(1 - \eta^k\left(1 + E_{j(k)}\right)\right) \\ + \underbrace{\frac{1}{m}\sum_{i=1}^{j(k)}\left(\delta_i\left(1 - r^2\right) + \epsilon_i\right)\left\|x^{k+1-i} - x^{k-i}\right\|^2}_{\textit{Conversion errors}}\end{aligned}$$

Hence we can see that for sufficiently small step size, the $\left\|S\hat{x}^k\right\|^2$ waste is eliminated.

Proof.

$$
\begin{aligned}
&-2\frac{\eta^k}{m}\left\langle x^k, S\hat{x}^k\right\rangle \\
&= -2\frac{\eta^k}{m}\left\langle \hat{x}^k, S\hat{x}^k\right\rangle - 2\frac{\eta^k}{m}\left\langle x^k - \hat{x}^k, S\hat{x}^k\right\rangle \\
&\leq -\frac{\eta^k}{m}\left(\left\|S\hat{x}^k\right\|^2 + \left(1-r^2\right)\|\hat{x}\|^2\right) + 2\frac{\eta^k}{m}\left\|x^k - \hat{x}^k\right\| \cdot \left\|S\hat{x}^k\right\|, \text{ (Lemma 3)} \\
&\leq -\frac{\eta^k}{m}\left(\left\|S\hat{x}^k\right\|^2 + \left(1-r^2\right)\left\|\hat{x}^k\right\|^2\right) + 2\frac{\eta^k}{m}\left(\frac{1}{2}\left\|S\hat{x}^k\right\|^2\eta^k E_{j(k)} + \frac{1}{2}\frac{1}{\eta^k}\sum_{i=1}^{j(k)}\left(\epsilon_i\left\|x^{k+1-i} - x^{k-i}\right\|^2\right)\right) \\
&= -\frac{\eta^k}{m}\left(1-r^2\right)\left\|\hat{x}^k\right\|^2 + \underbrace{\frac{1}{m}\sum_{i=1}^{j(k)}\epsilon_i\left\|x^{k+1-i} - x^{k-i}\right\|^2}_{\text{Conversion error}} - \frac{\eta^k}{m}\left\|S\hat{x}^k\right\|^2\left(1-\eta^k E_{j(k)}\right) \qquad (7.3.4)
\end{aligned}
$$

The final inequality followed from Lemma 9. Now let's examine $-\left\|\hat{x}^k\right\|^2$, which we convert to a $-\left\|x^k\right\|^2$ term (and some conversion error) for linear convergence.

$$
\begin{aligned}
-\left\|\hat{x}^k\right\|^2 &= -\left\|x^k\right\|^2 - 2\left\langle \hat{x}^k - x^k, x^k\right\rangle - \left\|x^k - \hat{x}^k\right\|^2 \\
&\leq -\left\|x^k\right\|^2 + 2\left\|\hat{x}^k - x^k\right\|\left\|x^k\right\| \\
\text{(Lemma 9)} &\leq -\left\|x^k\right\|^2 + \left\|x^k\right\|^2\eta^k D_{j(k)} + \frac{1}{\eta^k}\sum_{i=1}^{j(k)}\left(\delta_i\left\|x^{k+1-i} - x^{k-i}\right\|^2\right) \\
&= -\left(1-\eta^k D_{j(k)}\right)\left\|x^k\right\|^2 + \frac{1}{\eta^k}\sum_{i=1}^{j(k)}\left(\delta_i\left\|x^{k+1-i} - x^{k-i}\right\|^2\right)
\end{aligned}
$$

Hence substituting this into (7.3.4), we have

$$
\begin{aligned}
-2\frac{\eta^k}{m}\left\langle x^k, S\hat{x}^k\right\rangle &\leq -\frac{\eta^k}{m}\left(1-r^2\right)\left(1-\eta^k D_{j(k)}\right)\left\|x^k\right\|^2 + \frac{\eta^k}{m}\left(1-r^2\right)\frac{1}{\eta^k}\sum_{i=1}^{j(k)}\left(\delta_i\left\|x^{k+1-i} - x^{k-i}\right\|^2\right) \\
&\quad + \frac{1}{m}\sum_{i=1}^{j(k)}\epsilon_i\left\|x^{k+1-i} - x^{k-i}\right\|^2 - \frac{\eta^k}{m}\left\|S\hat{x}^k\right\|^2\left(1-\eta^k D_{j(k)}\right) \\
&= -\frac{\eta^k}{m}\left(1-r^2\right)\left(1-\eta^k D_{j(k)}\right)\left\|x^k\right\|^2 + \underbrace{\frac{1}{m}\sum_{i=1}^{j(k)}\left(\delta_i\left(1-r^2\right) + \epsilon_i\right)\left\|x^{k+1-i} - x^{k-i}\right\|^2}_{\text{Conversion error}} \\
&\quad - \frac{\eta^k}{m}\left\|S\hat{x}^k\right\|^2\left(1-\eta^k D_{j(k)}\right)
\end{aligned}
$$

Using (7.2.1) immediately yields the result. □

7.4 The Lyapunov function

We now consider how the Lyapunov function defined in Equation (4.4.1) changes in size from step to step. The reason that a Lyapunov function is needed is to deal with the $\left\|x^{k+1-i}-x^{k-i}\right\|^2$ terms. They cannot be easily negated like $\left\|S\hat{x}^k\right\|^2$ terms, and so must be incorporated into the Lyapunov function.

Lemma 11. *Let the conditions of Lemma 10 and Assumption 2 hold. Define*

$$\eta_1 = \left(1+\frac{c_1}{m}+\left\|\frac{1}{\epsilon_i}\right\|_{\ell^1}\right)^{-1} \tag{7.4.1}$$

$$\eta_2 = \left(\sum_{i=1}^{\infty}\frac{P_i}{\delta_i}\right)^{-1} \tag{7.4.2}$$

Let η^k be $\sigma\left(X^k\right)$-measurable, and $\eta^k \leq \eta_1$. Then ARock satisfies:

$$\mathbb{E}\left[\xi^{k+1}\middle|\sigma\left(X^k\right)\right] \leq \left\|x^k\right\|^2 R\left(\eta^k,\eta_2\right) + \frac{1}{m}\sum_{i=1}^{\infty}\left(\left(\epsilon_i+\left(1-r^2\right)\delta_i\right)P_i + c_{i+1}\right)\left\|x^{k+1-i}-x^{k-i}\right\|^2$$

Notice that we have defined η_1 and η_2 in terms of the unspecified parameters $(\epsilon_1,\epsilon_2,\ldots)$ and $(\delta_1,\delta_2,\ldots)$. Eventually, we will set $\epsilon_i = m^{1/2}P_i^{-1/2}\rho^{i/2}$ and $\delta_l = m^{1/2}P_l^{-1/2}\rho^{l/2}(1-r^2)^{-1}$ for reasons that will be explained in Section 7.6. With this parameter choice, the definition of η_1 will match eq. (7.1.4).

Proof.

$$\mathbb{E}\left[\xi^{k+1}\middle|\sigma\left(X^k,J^k\right)\right] = \underbrace{\mathbb{E}\left[\left\|x^{k+1}\right\|^2\middle|\sigma\left(X^k,J^k\right)\right]}_{A} + \underbrace{\frac{c_1}{m}\mathbb{E}\left[\left\|x^{k+1}-x^k\right\|^2\middle|\sigma\left(X^k,J^k\right)\right]}_{B} \tag{7.4.3}$$

$$+ \underbrace{\frac{1}{m}\sum_{i=1}^{\infty}c_{i+1}\left\|x^{k+1-i}-x^{k-i}\right\|^2}_{C} \tag{7.4.4}$$

We obtain a bound on A from Lemma 10. B follows by the definition of ARock:

$$B = \frac{c_1}{m}\frac{\left(\eta^k\right)^2}{m}\left\|S\hat{x}^k\right\|^2.$$

C contains no expectation because it is $\sigma\big(X^k, J^k\big)$ measurable. Hence we have:

$$\begin{aligned}
&\mathbb{E}\Big[\xi^{k+1}\Big|\sigma\big(X^k, J^k\big)\Big] \\
&\leq \left\|x^k\right\|^2\left(1-\frac{\eta^k}{m}\big(1-r^2\big)\big(1-\eta^k\big(D_{j(k)}\big)\big)\right)-\frac{\eta^k}{m}\left\|S\hat{x}^k\right\|^2\left(1-\eta^k\left(1+\frac{c_1}{m}+E_{j(k)}\right)\right) \\
&+\frac{1}{m}\sum_{i=1}^{j(k)}\big(\epsilon_i+\big(1-r^2\big)\delta_i\big)\left\|x^{k+1-i}-x^{k-i}\right\|^2+\frac{1}{m}\sum_{i=1}^{\infty}c_{i+1}\left\|x^{k+1-i}-x^{k-i}\right\|^2
\end{aligned} \tag{7.4.5}$$

Notice that $E_j=\sum_{i=1}^{j}1/\epsilon_i\leq\|1/\epsilon_i\|_{\ell^1}$ for all j, and that therefore the step size condition $\eta^k\leq\eta_1$ eliminates the $\left\|S\hat{x}^k\right\|^2$ term.

Now it becomes necessary to take expectations over the delay distribution (by taking the expectation with respect to $\sigma\big(X^k\big)$ instead of $\sigma\big(X^k, J^k\big)$). Notice that for a positive sequence $(\gamma_1,\gamma_2,\ldots)$, we have: $\mathbb{E}\Big[\sum_{i=1}^{j(k)}\gamma_i\Big|\sigma\big(X^k\big)\Big]\leq\sum_{i=1}^{\infty}P_i\gamma_i$. This yields:

$$\begin{aligned}
\mathbb{E}\Big[\xi^{k+1}\Big|\sigma\big(X^k\big)\Big]\leq\left\|x^k\right\|^2&\left(1-\frac{\eta^k}{m}\big(1-r^2\big)\left(1-\eta^k\left(\sum_{i=1}^{\infty}\frac{P_i}{\delta_i}\right)\right)\right) \\
&+\frac{1}{m}\sum_{i=1}^{\infty}\big(\epsilon_i+\big(1-r^2\big)\delta_i\big)P_i\left\|x^{k+1-i}-x^{k-i}\right\|^2+\frac{1}{m}\sum_{i=1}^{\infty}c_{i+1}\left\|x^{k+1-i}-x^{k-i}\right\|^2
\end{aligned}$$

which completes the proof. □

7.5 Linear convergence

The right-hand side in Lemma 11 closely resembles ξ^k. Ideally, we have:

$$\mathbb{E}\Big[\xi^{k+1}\Big|\sigma\big(X^k\big)\Big]\leq\gamma\xi^k \tag{7.5.1}$$

for some $0<\gamma<1$ as small as possible, and some choice of parameters $(\epsilon_1,\epsilon_2,\ldots)$, $(\delta_1,\delta_2,\ldots)$ and coefficients $(c_1,c_2,\ldots)$. In this section we will derive such a result by carefully chosing these parameters. However, we need the following lemma in order to derive a coefficient formula.

Lemma 12. Coefficient formula. *Let $0<\rho<1$ and let $(s_1,s_2,\ldots)$ be a positive sequence. Consider the coefficient formula:*

$$c_i=\sum_{l=i}^{\infty}s_l\rho^{-(l-i+1)}. \tag{7.5.2}$$

If $c_1 < \infty$, then we have $c_i \downarrow 0$ and:

$$\rho c_i = c_{i+1} + s_i \tag{7.5.3}$$

Proof.

$$\rho c_i = \sum_{l=i}^{\infty} s_l \rho^{-(l-i)} = \sum_{l=i+1}^{\infty} s_l \rho^{-(l-(i+1)+1)} + s_i = c_{i+1} + s_i$$

Clearly this implies $c_i \downarrow 0$, since $\rho < 1$ and coefficients are nonnegative. □

Recall that ρ is defined in eq. (7.1.2). We now present a general linear convergence result. Afterwards in Section 7.6, we will make the precise parameter choice that yields optimal complexity.

Proposition 13. Linear convergence for stochastic delays. *Let Assumption 1 hold. Let $\eta^k \leq \eta_1$, and let $\epsilon_1, \epsilon_2, \ldots > 0$ and $\delta_1, \delta_2, \ldots > 0$ be a sequence of parameters. Let*

$$\sum_{l=i}^{\infty} \left(\epsilon_l + \left(1 - r^2\right)\delta_l\right) P_l \rho^{-l} < \infty \tag{7.5.4}$$

With the choice of coefficients[2]*:*

$$c_i = \sum_{l=i}^{\infty} \left(\epsilon_l + \left(1 - r^2\right)\delta_l\right) P_l \rho^{-(l-i+1)} \tag{7.5.5}$$

We have the following linear convergence result:

$$\mathbb{E}\left[\xi^{k+1} \middle| \sigma\left(X^k\right)\right] \leq R\left(\eta^k, \eta_2\right)\xi^k \tag{7.5.6}$$

Proof. By applying Lemma 12 with $s_l = P_l(\epsilon_l + (1 - r^2)\delta_l)$, we obtain:

$$\rho c_i = \left(\epsilon_i + \left(1 - r^2\right)\delta_i\right) P_i + c_{i+1}$$

Hence from Lemma 11, we have:

$$\begin{aligned}
\mathbb{E}\left[\xi^{k+1} \middle| \sigma\left(X^k\right)\right] &\leq \left\|x^k\right\|^2 R\left(\eta^k, \eta_2\right) + \frac{1}{m} \sum_{i=1}^{\infty} \left(\left(\epsilon_i + \left(1 - r^2\right)\delta_i\right) P_i + c_{i+1}\right) \left\|x^{k+1-i} - x^{k-i}\right\|^2 \\
&\leq \left\|x^k\right\|^2 R\left(\eta^k, \eta_2\right) + \frac{1}{m} \sum_{i=1}^{\infty} \rho c_i \left\|x^{k+1-i} - x^{k-i}\right\|^2 \leq \max\left(\rho, R\left(\eta^k, \eta_2\right)\right)\xi^k = \rho \xi^k
\end{aligned}$$

The last line follows, because $0 \leq \eta^k \leq \eta_1 \leq 1$ implies $\rho \leq R(\eta^k, \eta_2)$. □

[2] *This formula will eventually match the formula $c_i = 2m^{1/2} \sum_{l=i}^{\infty} P_l^{1/2} \rho^{i-l/2-1}$ given in Theorem 2 when the parameters $\epsilon_1, \epsilon_2, \ldots > 0$ and $\delta_1, \delta_2, \ldots > 0$ are chosen later.*

7.6 Proof of Theorem 2

Recall we have the following step size restriction $\eta^k \leq (1 + c_1/m + \|1/\epsilon_i\|_{\ell^1})^{-1}$ (with η_1 defined in eq. (7.4.1)) coupled with the convergence rate:

$$R\big(\eta^k, \eta_2\big) = 1 - \frac{\eta^k}{m}\big(1 - r^2\big)\big(1 - \eta^k \eta_2^{-1}\big)$$

for $\eta_2^{-1} = \sum_{i=1}^{\infty} P_i/\delta_i$. We now prove Theorem 2. However we do so in a way that justifies the choice of parameters that we use.

For $(\epsilon_1, \epsilon_2, \ldots)$, there is an unambiguous best choice. We maximize η_1 over the sequence ϵ_i, by letting $\epsilon_i = \sqrt{m} P_i^{-1/2} \rho^{i/2}$. All thing being equal, increasing η_1 allows for a better convergence rate by increasing the range of possible step sizes. This leads to:

$$\eta_1^{-1} = 1 + m^{-1}\big(1 - r^2\big) \sum_{l=1}^{\infty} \delta_l P_l \rho^{-l} + 2m^{-1/2} M \tag{7.6.1}$$

for M defined in (7.1.3). η_2 is unchanged.

For δ_l, we are faced with a trade-off. Larger δ_l increases η_2, which improves the convergence rate. Smaller δ_l increases η_1, which as discussed is advantageous. To solve this trade-off, we set $\delta_l = m^{1/2} P_l^{-1/2} \rho^{l/2} (1 - r^2)^{-1}$, which leads to:

$$\begin{aligned}
\eta_1^{-1} &= 1 + 3m^{-1/2} M \\
\eta_2^{-1} &= m^{-1/2}\big(1 - r^2\big) \sum_{i=1}^{\infty} P_l^{3/2} \rho^{-l/2} \\
&\leq m^{-1/2}\big(1 - r^2\big) M
\end{aligned}$$

This choice essentially maximizes η_2 subject to keeping the asymptotic value of η_1 essentially the same. Let $\phi = M m^{-1/2}$. Hence the convergence rate is:

$$\begin{aligned}
R(\eta_1, \eta_2) &= 1 - (\eta_1/m)\big(1 - r^2\big)\big(1 - \eta_1 \eta_2^{-1}\big) \\
&= 1 - \frac{1}{m}\big(1 - r^2\big)\frac{1}{1 + 3\phi}\left(1 - \frac{(1 - r^2)\phi}{1 + 3\phi}\right) \\
&\leq 1 - \frac{1}{m}\big(1 - r^2\big)\frac{1 + 2\phi}{(1 + 3\phi)^2} \\
&\leq 1 - \frac{1}{m}\big(1 - r^2\big)\frac{1}{1 + 5\phi}
\end{aligned}$$

Now we calculate the corresponding epoch complexity $I(\epsilon)$:

$$\epsilon = \left(1 - \frac{1}{m}\big(1 - r^2\big)\frac{1}{1+5\phi}\right)^{mI(\epsilon)}$$
$$I(\epsilon) = -\frac{1}{m}\ln(1/\epsilon)/\ln\left(1 - \frac{1}{m}\big(1 - r^2\big)\frac{1}{1+5\phi}\right)$$

Note that $-x/\ln(1-x) \leq 1 - x$. Applying this with $x = \frac{1}{m}(1-r^2)\frac{1}{1+5\phi}$ yields:

$$\begin{aligned} I(\epsilon) &\leq (1+5\phi)\frac{1}{1-r^2}\ln(1/\epsilon) \\ &= (1+5\phi)I_{\mathrm{Sync}}(\epsilon) \end{aligned}$$

With our parameter choice, the coefficients of the Lyapunov function are given by:

$$\begin{aligned} c_i &= \sum_{l=i}^{\infty}\Big(\epsilon_l + \big(1-r^2\big)\delta_l\Big)P_l\rho^{-(l-i+1)} \\ &= \sum_{l=i}^{\infty}\Big(m^{1/2}P_i^{-1/2}\rho^{i/2} + \big(1-r^2\big)m^{1/2}P_l^{-1/2}\rho^{l/2}\big(1-r^2\big)^{-1}\Big)P_l\rho^{-(l-i+1)} \\ &= 2m^{1/2}\sum_{l=i}^{\infty}P_l^{1/2}\rho^{-(l/2-i+1)} \end{aligned}$$

This completes the proof.

Part III

Asynchronous Acceleration

In this part, we present the results of (Hannah, Feng, and Wotao Yin 2018).

CHAPTER 8

Asynchronous Acceleration

In this section, we propose and prove the convergence of the **A**synchronous **A**ccelerated Nonuniform Randomized **B**lock **C**oordinate **D**escent 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:

$$\min_{x\in\mathbb{R}^d} f(x) = f\Big(x_{(1)},\ldots,x_{(n)}\Big) \tag{8.0.1}$$

where f is σ-strongly convex for $\sigma > 0$ with L-Lipschitz gradient $\nabla f = (\nabla_1 f, \ldots, \nabla_n f)$. f is assumed to be L_i-coordinate smooth in this section. 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 section. 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 and 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 Section 3.5). Hence `A2BCD` can be used as an asynchronous Nesterov-accelerated finite-sum algorithm.

8.1 Summary of Results

We prove that `A2BCD` attains `NU_ACDM`'s state-of-the-art iteration complexity to highest order for solving (8.0.1), so long as delays are not too large. The proof is very different from that of (Allen-Zhu, Qu, et al. 2016), 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 8.3). This extends results in (Lan and Y. Zhou 2017) 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. In Chapter 12 we confirm with numerical experiments on a small-scale shared-memory architecture that A2BCD is the current fastest coordinate descent algorithm. We find that A2BCD can approximately solve the (dual) ridge regression problem up to $4-5\times$ faster than NU_ACDM for various data sets from LIBSVM (Chang and C.-J. Lin 2011). We also discuss critical elements of an efficient implementation, including the sparse-update reformulation of A2BCD and parameter tuning.

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.

The work of (Meng et al. 2016) 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 $(n+\kappa)\ln(1/\epsilon)$ for $\kappa = \mathcal{O}(\tau n^2)$ (Corollary 4.4). Hence authors do not even obtain accelerated rates. Their convergence condition is $\tau < \frac{1}{4\Delta^{1/8}}$ for sparsity parameter Δ. 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, Huang, and Z. Lin 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 τ times larger than the serial complexity. Since τ 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, 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 Chapter 11). This extends the ODE found in (Su, Boyd, and Candes 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.

8.2 Main Theoretical 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" (Zhimin Peng, Tianyu Wu, et al. 2016). 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 (Q. Lin, Lu, and Xiao 2014)).

To calculate the $k+1$'th iteration of the algorithm from iteration k, we use only one block of the gradient $\nabla_{i(k)} f$. 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 (T. Sun, Hannah, and Wotao Yin 2017; Leblond, Pedregosa, and Lacoste-Julien 2017; Cannelli et al. 2017).

Definition 14. Asynchronous Accelerated Randomized Block Coordinate Descent (`A2BCD`). Let f be σ-strongly convex, and let its gradient ∇f be L-Lipschitz with block coordinate Lipschitz parameters L_i. We define the **condition number** $\kappa = L/\sigma$, and let $\underline{\mathrm{L}} = \min_i L_i$. Using these parameters, we sample $i(k)$ in an independent and identically distributed (IID) fashion according to

$$\mathbb{P}[i(k) = j] = L_j^{1/2}/S, \quad j \in \{1, \dots, n\}, \quad \text{for } S = \textstyle\sum_{i=1}^{n} L_i^{1/2}. \tag{8.2.1}$$

Let τ be the maximum asynchronous delay. We define the dimensionless **asynchronicity parameter** ψ, which is proportional to τ, and quantifies how strongly asynchronicity will affect convergence:

$$\psi = 9\Big(S^{-1/2}\underline{\mathrm{L}}^{-1/2}L^{3/4}\kappa^{1/4}\Big) \times \tau \tag{8.2.2}$$

We use the above system parameters and ψ to define the coefficients α, β, and γ via eqs. (8.2.3) to (8.2.5). Hence `A2BCD` algorithm is defined via the iterations: eqs. (8.2.6) to (8.2.8).

$$\alpha \triangleq \Big(1 + (1+\psi)\sigma^{-1/2}S\Big)^{-1} \tag{8.2.3}$$

$$\beta \triangleq 1 - (1-\psi)\sigma^{1/2}S^{-1} \tag{8.2.4}$$

$$h \triangleq 1 - \frac{1}{2}\sigma^{1/2}\underline{\mathrm{L}}^{-1/2}\psi. \tag{8.2.5}$$

$$y_k = \alpha v_k + (1-\alpha)x_k, \tag{8.2.6}$$

$$x_{k+1} = y_k - hL_{i(k)}^{-1}\nabla_{i(k)}f(\hat{y}_k), \tag{8.2.7}$$

$$v_{k+1} = \beta v_k + (1-\beta)y_k - \sigma^{-1/2}L_{i(k)}^{-1/2}\nabla_{i(k)}f(\hat{y}_k). \tag{8.2.8}$$

See Section 12.1 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.

We again use this asynchronicity error, as in previous sections. However, the coefficients in this setting will be quite different.

Definition 15. Asynchronicity error. Using the above parameters, we define:

$$A_k = \sum_{j=1}^{\tau} c_j \|y_{k+1-j} - y_{k-j}\|^2 \tag{8.2.9}$$

$$\text{for } c_i = \frac{6}{S}L^{1/2}\kappa^{3/2}\tau\sum_{j=i}^{\tau}\Big(1-\sigma^{1/2}S^{-1}\Big)^{i-j-1}\psi^{-1}. \tag{8.2.10}$$

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 $\mathbb{E}_k[X]$ 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[1], we assume that the minimizer $x_* = 0$, and that

[1]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: $\|v_k - x_*\|^2 + A_k + c(f(x_k) - f(x_*))$.

$f(x_*) = 0$ with no loss in generality. We define the **Lyapunov function**:

$$\rho_k = \|v_k\|^2 + A_k + cf(x_k) \qquad (8.2.11) \qquad \text{for } c = 2\sigma^{-1/2}S^{-1}\big(\beta\alpha^{-1}(1-\alpha)+1\big). \qquad (8.2.12)$$

This combines Nesterov's Lyapunov function for acceleration (Y. Nesterov 2012) and the asynchronicity error.

We now present this part's first main contribution.

Theorem 3. Let f be σ-strongly convex with a gradient ∇f that is L-Lipschitz with block Lipschitz constants $\{L_i\}_{i=1}^n$. Let ψ defined in (8.2.2) satisfy $\psi \leq \frac{3}{7}$ (i.e. $\tau \leq \frac{1}{21}S^{1/2}\underline{L}^{1/2}L^{-3/4}\kappa^{-1/4}$). Then for `A2BCD` we have:

$$\mathbb{E}_k[\rho_{k+1}] \leq \Big(1 - (1-\psi)\sigma^{1/2}S^{-1}\Big)\rho_k.$$

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

$$K_{\texttt{A2BCD}}(\epsilon) = \Big(\sigma^{-1/2}S + \mathcal{O}(1)\Big)\frac{\ln(1/\epsilon)}{1-\psi}, \qquad (8.2.13)$$

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

This result is proven in Chapter 9. A stronger result for $L_i \equiv L$ can be proven, but this adds to the complexity of the proof. In practice, asynchronous algorithms are far more resilient to delays than the theory predicts. τ 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, Qu, et al. 2016) (Theorem 5.1) shows a linear convergence rate of $1 - 2/\Big(1 + 2\sigma^{-1/2}S\Big)$ for `NU_ACDM`, which leads to the corresponding iteration complexity of $K_{\texttt{NU\_ACDM}}(\epsilon) = \Big(\sigma^{-1/2}S + \mathcal{O}(1)\Big)\ln(1/\epsilon)$. Hence, we have:

$$K_{\texttt{A2BCD}}(\epsilon) = \frac{1}{1-\psi}(1+o(1))K_{\texttt{NU\_ACDM}}(\epsilon)$$

When $0 \leq \psi \ll 1$, or equivalently, when $\tau \ll S^{1/2}\underline{L}^{1/2}L^{-3/4}\kappa^{-1/4}$, 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 τ required for good complexity.

Corollary 16. *Let the conditions of Theorem 3 hold. If all coordinate-wise Lipschitz constants L_i are equal (i.e. $L_i = L_1, \forall i$), then we have $K_{\texttt{A2BCD}}(\epsilon) \sim K_{\texttt{NU\_ACDM}}(\epsilon)$ 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 $K_{\texttt{A2BCD}}(\epsilon) \sim K_{\texttt{NU\_ACDM}}(\epsilon) = K_{\texttt{ACDM}}(\epsilon)$, when $\tau \ll n^{1/2}\kappa^{-1/4}$.*

Remark 2. Reduction to synchronous case. Notice that when $\tau = 0$, we have $\psi = 0$, $c_i \equiv 0$ and hence $A_k \equiv 0$. Thus `A2BCD` becomes equivalent to `NU_ACDM`, the Lyapunov function[2] ρ_k becomes equivalent to one found in (Allen-Zhu, Qu, et al. 2016)(pg. 9), and Theorem 3 yields the same complexity.

The maximum delay τ will be a function $\tau(p)$ of p, number of computing nodes. Clearly $\tau \geq p$, and experimentally it has been observed that $\tau = \mathcal{O}(p)$ (Leblond, Pedregosa, and Lacoste-Julien 2017). Let gradient complexity $K(\epsilon, \tau)$ be the number of gradients required for an asynchronous algorithm with maximum delay τ to attain suboptimality ϵ. $\tau(1) = 0$, 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 3, 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]Their Lyapunov function is in fact a generalization of the one found in (Y. Nesterov 2012).

8.3 Optimality

`NU_ACDM` and hence `A2BCD` are in fact optimal in some sense. That is, among a fairly wide class of coordinate descent algorithms $\mathcal{A}$, they have the best-possible worst-case complexity to highest order. We extend the work in (Lan and Y. Zhou 2017) to encompass algorithms are asynchronous and have unequal L_i. For a subset $S \in \mathbb{R}^d$, we let $\mathrm{IC}(S)$ (inconsistent read) denote the set of vectors v whose components are a combination of components of vectors in the set S. That is, $v = (v_{1,1}, v_{2,2}, \ldots, v_{d,d})$ for some vectors $v_1, v_2, \ldots, v_d \in S$. Here $v_{i,j}$ denotes the jth component of vector v_i.

Definition 17. Asynchronous Randomized Incremental Algorithms. Consider the unconstrained minimization problem (8.0.1) for function f satisfying the conditions stated previously in this section. We define the class $\mathcal{A}$ as algorithms G on this problem such that:

1. For each parameter set $(\sigma, L_1, \ldots, L_n, n)$, G has an associated IID random variable $i(k)$ with some fixed distribution $\mathbb{P}[i(k)] = p_i$ for $\sum_{i=1}^n p_i = 1$.

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

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

Theorem 4. *For any algorithm $G \in \mathcal{A}$ that solves eq. (8.0.1), and parameter set $(\sigma, L_1, \ldots, L_n, n)$, there is a dimension d, a corresponding function f on $\mathbb{R}^d$, and a starting point x_0, such that*

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

Hence $\mathcal{A}$ has a complexity lower bound: $K(\epsilon) \geq \frac{1}{4}(1 + o(1))\Big(\sum_{j=1}^n \sqrt{L_i/\sigma} + 2n\Big) \ln(1/2\epsilon)$

Our proof in Chapter 10 follows very similar lines to (Lan and Y. Zhou 2017; Yurii Nesterov 2013).

CHAPTER 9

Proofs for Asynchronous Acceleration

In this section, we prove Theorem 3. We find it convenient to define the norm:

$$\|s\|_* = \sqrt{\sum_{i=1}^{n} L_i^{-1/2} \|s_i\|^2} \tag{9.0.1}$$

9.1 Starting point

First notice that using the definition (8.2.8) of v_{k+1} we have:

$$\begin{aligned}
\|v_{k+1}\|^2 &= \|\beta v_k + (1-\beta) y_k\|^2 - 2\sigma^{-1/2} L_{i(k)}^{-1/2} \left\langle \beta v_k + (1-\beta) y_k, \nabla_{i(k)} f(\hat{y}_k) \right\rangle + \sigma^{-1} L_{i(k)}^{-1} \left\| \nabla_{i(k)} f(\hat{y}_k) \right\|^2 \\
\mathbb{E}_k \|v_{k+1}\|^2 &= \underbrace{\|\beta v_k + (1-\beta) y_k\|^2}_{A} - 2\sigma^{-1/2} S^{-1} \underbrace{\langle \beta v_k + (1-\beta) y_k, \nabla f(\hat{y}_k) \rangle}_{B} \\
&\quad + S^{-1} \sigma^{-1} \underbrace{\sum_{i=1}^{n} L_i^{-1/2} \|\nabla_i f(\hat{y}_k)\|^2}_{C}
\end{aligned} \tag{9.1.1}$$

We have the following general identity:

$$\|\beta x + (1-\beta) y\|^2 = \beta \|x\|^2 + (1-\beta) \|y\|^2 - \beta(1-\beta) \|x - y\|^2, \, \forall x, y \tag{9.1.2}$$

It can also easily be verified from (8.2.6) that we have:

$$v_k = y_k + \alpha^{-1}(1-\alpha)(y_k - x_k) \tag{9.1.3}$$

Using (9.1.2) on term A, (9.1.3) on term B, and recalling the definition (9.0.1) on term C, we have from (9.1.1):

$$\begin{aligned}
\mathbb{E}_k \|v_{k+1}\|^2 &= \beta \|v_k\|^2 + (1-\beta) \|y_k\|^2 - \beta(1-\beta) \|v_k - y_k\|^2 + S^{-1} \sigma^{-1/2} \|\nabla f(\hat{y}_k)\|_*^2 \\
&\quad - 2\sigma^{-1/2} S^{-1} \beta \alpha^{-1} (1-\alpha) \langle y_k - x_k, \nabla f(\hat{y}_k) \rangle - 2\sigma^{-1/2} S^{-1} \langle y_k, \nabla f(\hat{y}_k) \rangle
\end{aligned} \tag{9.1.4}$$

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

9.2 The Cross Term

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

Lemma 18. *Let $\chi, A > 0$. Let the delay be bounded by τ. 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 and Wotao Yin 2017a). □

Lemma 19. *We have:*

$$-\langle\nabla f(\hat{y}_k), y_k\rangle \leq -f(y_k) - \frac{1}{2}\sigma(1-\psi)\|y_k\|^2 + \boldsymbol{\frac{1}{2}L\kappa\psi^{-1}\tau\sum_{j=1}^{\tau}\|y_{k+1-j} - y_{k-j}\|^2} \quad (9.2.1)$$

$$\langle\nabla f(\hat{y}_k), x_k - y_k\rangle \leq f(x_k) - f(y_k) \quad (9.2.2)$$
$$+ \boldsymbol{\frac{1}{2}L\alpha(1-\alpha)^{-1}\left(\kappa^{-1}\psi\beta\|v_k - y_k\|^2 + \kappa\psi^{-1}\beta^{-1}\tau\sum_{j=1}^{\tau}\|y_{k+1-j} - y_{k-j}\|^2\right)}$$

The terms in bold in (9.2.1) and (9.2.2) are a result of the asynchronicity, and are identically 0 in its absence.

Proof. Our strategy is to separately analyze terms that appear in the traditional analysis of (Y. Nesterov 2012), and the terms that result from asynchronicity. We first prove (9.2.1):

$$\begin{aligned}-\langle\nabla f(\hat{y}_k), y_k\rangle &= -\langle\nabla f(y_k), y_k\rangle - \langle\nabla f(\hat{y}_k) - \nabla f(y_k), y_k\rangle \\ &\leq -f(y_k) - \frac{1}{2}\sigma\|y_k\|^2 + L\|\hat{y}_k - y_k\|\|y_k\| \quad (9.2.3)\end{aligned}$$

(9.2.3) follows from strong convexity ((3.1.1) with $x = y_k$ and $y = x_*$), and the fact that ∇f is L-Lipschitz. The term due to asynchronicity becomes:

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

using Lemma 18 with $\chi = \kappa\psi^{-1}, A = \|y_k\|$. Combining this with (9.2.3) completes the proof of (9.2.1).

We now prove (9.2.2):

$$\begin{aligned}\langle \nabla f(\hat{y}_k), x_k - y_k\rangle &= \langle \nabla f(y_k), x_k - y_k\rangle + \langle \nabla f(\hat{y}_k) - \nabla f(y_k), x_k - y_k\rangle \\ &\le f(x_k) - f(y_k) + L\|\hat{y}_k - y_k\|\|x_k - y_k\| \\ &\le f(x_k) - f(y_k) \\ &+ \frac{1}{2}L\left(\kappa^{-1}\psi\beta\alpha^{-1}(1-\alpha)\|x_k - y_k\|^2 + \kappa\psi^{-1}\beta^{-1}\alpha(1-\alpha)^{-1}\tau\sum_{j=1}^{\tau}\|y_{k+1-j} - y_{k-j}\|^2\right)\end{aligned}$$

Here the last line follows from Lemma 18 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 (8.2.6):

$$y_k - x_k = \alpha(1-\alpha)^{-1}(v_k - y_k) \qquad \square$$

9.3 Function-value term

Much like (Y. 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 20. *We have:*

$$\begin{aligned}\mathbb{E}_k f(x_{k+1}) \le f(y_k) - \frac{1}{2}h\left(2 - h\left(1 + \frac{\mathbf{1}}{\mathbf{2}}\boldsymbol{\sigma}^{\mathbf{1/2}}\underline{L}^{\mathbf{-1/2}}\boldsymbol{\psi}\right)\right)S^{-1}\|\nabla f(\hat{y}_k)\|_*^2 \\ + \boldsymbol{S^{-1}L\sigma^{1/2}\kappa\psi^{-1}\tau}\sum_{\boldsymbol{j=1}}^{\boldsymbol{\tau}}\|\boldsymbol{y_{k+1-j} - y_{k-j}}\|^{\mathbf{2}}\end{aligned}$$

Proof. From the definition (8.2.7) 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 (3.1.2) to obtain:

$$\begin{aligned}f(x_{k+1}) &\le f(y_k) + \langle \nabla f(y_k), x_{k+1} - y_k\rangle + \frac{1}{2}L_{i(k)}\|x_{k+1} - y_k\|^2 \\ \text{(from (8.2.7))} &= f(y_k) - hL_{i(k)}^{-1}\left\langle \nabla_{i(k)} f(y_k), \nabla_{i(k)} f(\hat{y}_k)\right\rangle + \frac{1}{2}h^2 L_{i(k)}^{-1}\left\|\nabla_{i(k)} f(\hat{y}_k)\right\|^2\end{aligned}$$

$$= f(y_k) - hL_{i(k)}^{-1}\left\langle \nabla_{i(k)}f(y_k) - \nabla_{i(k)}f(\hat{y}_k), \nabla_{i(k)}f(\hat{y}_k)\right\rangle - \frac{1}{2}h(2-h)L_{i(k)}^{-1}\left\|\nabla_{i(k)}f(\hat{y}_k)\right\|^2$$

$$\mathbb{E}_k f(x_{k+1}) \leq f(y_k) - hS^{-1}\sum_{i=1}^{n} L_i^{-1/2}\langle \nabla_i f(y_k) - \nabla_i f(\hat{y}_k), \nabla_i f(\hat{y}_k)\rangle - \frac{1}{2}h(2-h)S^{-1}\|\nabla f(\hat{y}_k)\|_*^2 \tag{9.3.1}$$

Here the last line followed from the definition (9.0.1) of the norm $\|\cdot\|_{*1/2}$. We now analyze the middle term:

$$\begin{aligned}
&-\sum_{i=1}^{n} L_i^{-1/2}\langle \nabla_i f(y_k) - \nabla_i f(\hat{y}_k), \nabla_i f(\hat{y}_k)\rangle \\
&= -\left\langle \sum_{i=1}^{n} L_i^{-1/4}(\nabla_i f(y_k) - \nabla_i f(\hat{y}_k)), \sum_{i=1}^{n} L_i^{-1/4}\nabla_i f(\hat{y}_k)\right\rangle \\
\text{(Cauchy Schwarz)} &\leq \left\|\sum_{i=1}^{n} L_i^{-1/4}(\nabla_i f(y_k) - \nabla_i f(\hat{y}_k))\right\|\left\|\sum_{i=1}^{n} L_i^{-1/4}\nabla_i f(\hat{y}_k)\right\| \\
&= \left(\sum_{i=1}^{n} L_i^{-1/2}\|\nabla_i f(y_k) - \nabla_i f(\hat{y}_k)\|^2\right)^{1/2}\left(\sum_{i=1}^{n} L_i^{-1/2}\|\nabla_i f(\hat{y}_k)\|^2\right)^{1/2} \\
(\underline{L} \leq L_i, \forall i \text{ and } (9.0.1)) &\leq \underline{L}^{-1/4}\|\nabla f(y_k) - \nabla f(\hat{y}_k)\|\|\nabla f(\hat{y}_k)\|_* \\
(\nabla f \text{ is } L\text{-Lipschitz}) &\leq \underline{L}^{-1/4}L\|y_k - \hat{y}_k\|\|\nabla f(\hat{y}_k)\|_*
\end{aligned}$$

We then apply Lemma 18 to this with $\chi = 2h^{-1}\sigma^{1/2}\underline{L}^{1/4}\kappa\psi^{-1}$, $A = \|\nabla f(\hat{y}_k)\|_*$ to yield:

$$\begin{aligned}
-\sum_{i=1}^{n} L_i^{-1/2}\langle \nabla_i f(y_k) - \nabla_i f(\hat{y}_k), \nabla_i f(\hat{y}_k)\rangle \leq h^{-1}L\sigma^{1/2}\kappa\psi^{-1}\tau\sum_{j=1}^{\tau}\|y_{k+1-j} - y_{k-j}\|^2 \\
+ \frac{1}{4}h\sigma^{1/2}\underline{L}^{-1/2}\psi\|\nabla f(\hat{y}_k)\|_*^2
\end{aligned} \tag{9.3.2}$$

Finally to complete the proof, we combine (9.3.1), with (9.3.2). □

9.4 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 21. *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:

$$\mathbb{E}_k[A_{k+1} - rA_k] = c_1 \mathbb{E}_k \|y_{k+1} - y_k\|^2 - \sum_{j=1}^{\infty} s_j \|y_{k+1-j} - y_{k-j}\|^2$$

Remark 3. 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 $c_1 \mathbb{E}_k \|y_{k+1} - y_k\|^2$, 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.

$$\begin{aligned}
\mathbb{E}_k[A_{k+1} - rA_k] &= \mathbb{E}_k \sum_{j=0}^{\infty} c_{j+1} \|y_{k+1-j} - y_{k-j}\|^2 - r\mathbb{E}_k \sum_{j=1}^{\infty} c_j \|y_{k+1-j} - y_{k-j}\|^2 \\
&= c_1 \mathbb{E}_k \|y_{k+1} - y_k\|^2 + \mathbb{E}_k \sum_{j=1}^{\infty} (c_{j+1} - rc_j) \|y_{k+1-j} - y_{k-j}\|^2
\end{aligned}$$

Noting the following completes the proof:

$$c_{i+1} - rc_i = \sum_{j=i+1}^{\infty} r^{i+1-j-1} s_j - r \sum_{j=i}^{\infty} r^{i-j-1} s_j = -s_i \qquad \square$$

Given that A_k allows us to negate difference terms, we now analyze the cost $c_1 \mathbb{E}_k \|y_{k+1} - y_k\|^2$ of this negation.

Lemma 22. *We have:*

$$\mathbb{E}_k \|y_{k+1} - y_k\|^2 \leq 2\alpha^2 \beta^2 \|v_k - y_k\|^2 + 2S^{-1} \underline{L}^{-1} \|\nabla f(\hat{y}_k)\|^2$$

Proof.

$$\begin{aligned}
y_{k+1} - y_k &= (\alpha v_{k+1} + (1-\alpha) x_{k+1}) - y_k \\
&= \alpha\Big(\beta v_k + (1-\beta) y_k - \sigma^{-1/2} L_{i(k)}^{-1/2} \nabla_{i(k)} f(\hat{y}_k)\Big) + (1-\alpha)\Big(y_k - h L_{i(k)}^{-1} \nabla_{i(k)} f(\hat{y}_k)\Big) - y_k \qquad (9.4.1) \\
&= \alpha\beta v_k + \alpha(1-\beta) y_k - \alpha\sigma^{-1/2} L_{i(k)}^{-1/2} \nabla_{i(k)} f(\hat{y}_k) - \alpha y_k - (1-\alpha) h L_{i(k)}^{-1} \nabla_{i(k)} f(\hat{y}_k)
\end{aligned}$$

$$= \alpha\beta(v_k - y_k) - \Big(\alpha\sigma^{-1/2}L_{i(k)}^{-1/2} + h(1-\alpha)L_{i(k)}^{-1}\Big)\nabla_{i(k)}f(\hat{y}_k)$$

$$\|y_{k+1} - y_k\|^2 \leq 2\alpha^2\beta^2\|v_k - y_k\|^2 + 2\Big(\alpha\sigma^{-1/2}L_{i(k)}^{-1/2} + h(1-\alpha)L_{i(k)}^{-1}\Big)^2\Big\|\nabla_{i(k)}f(\hat{y}_k)\Big\|^2 \tag{9.4.2}$$

Here (9.4.1) following from (8.2.8), the definition of v_{k+1}. (9.4.2) follows from the inequality $\|x+y\|^2 \leq 2\|x\|^2 + 2\|y\|^2$. The rest is simple algebraic manipulation.

$$\|y_{k+1} - y_k\|^2 \leq 2\alpha^2\beta^2\|v_k - y_k\|^2 + 2L_{i(k)}^{-1}\Big(\alpha\sigma^{-1/2} + h(1-\alpha)L_{i(k)}^{-1/2}\Big)^2\Big\|\nabla_{i(k)}f(\hat{y}_k)\Big\|^2$$

$$(\underline{\mathrm{L}} \leq L_i, \forall i) \leq 2\alpha^2\beta^2\|v_k - y_k\|^2 + 2L_{i(k)}^{-1}\Big(\alpha\sigma^{-1/2} + h(1-\alpha)\underline{\mathrm{L}}^{-1/2}\Big)^2\Big\|\nabla_{i(k)}f(\hat{y}_k)\Big\|^2$$

$$= 2\alpha^2\beta^2\|v_k - y_k\|^2 + 2L_{i(k)}^{-1}\underline{\mathrm{L}}^{-1}\Big(\underline{\mathrm{L}}^{1/2}\sigma^{-1/2}\alpha + h(1-\alpha)\Big)^2\Big\|\nabla_{i(k)}f(\hat{y}_k)\Big\|^2$$

$$\mathbb{E}\|y_{k+1} - y_k\|^2 \leq 2\alpha^2\beta^2\|v_k - y_k\|^2 + 2S^{-1}\underline{\mathrm{L}}^{-1}\Big(\underline{\mathrm{L}}^{1/2}\sigma^{-1/2}\alpha + h(1-\alpha)\Big)^2\|\nabla f(\hat{y}_k)\|_*^2$$

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

$$\underline{\mathrm{L}}^{1/2}\sigma^{-1/2}\alpha + h(1-\alpha) = h + \alpha\Big(\underline{\mathrm{L}}^{1/2}\sigma^{-1/2} - h\Big)$$

$$\text{(definitions of } h \text{ and } \alpha\text{: (8.2.3), and (8.2.5))} = 1 - \frac{1}{2}\sigma^{1/2}\underline{\mathrm{L}}^{-1/2}\psi + \sigma^{1/2}S^{-1}\Big(\underline{\mathrm{L}}^{1/2}\sigma^{-1/2}\Big)$$

$$\leq 1 - \sigma^{1/2}\underline{\mathrm{L}}^{-1/2}\Big(\frac{1}{2}\psi - \sigma^{-1/2}S^{-1}\underline{\mathrm{L}}^{1}\Big) \tag{9.4.3}$$

Rearranging the definition of ψ, we have:

$$S^{-1} = \frac{1}{9^2}\psi^2\underline{\mathrm{L}}^1 L^{-3/2}\kappa^{-1/2}\tau^{-2}$$

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

Using this on (9.4.3), we have:

$$\underline{\mathrm{L}}^{1/2}\alpha\sigma^{-1/2} + h(1-\alpha) \leq 1 - \sigma^{1/2}\underline{\mathrm{L}}^{-1/2}\Big(\frac{1}{2}\psi - \frac{1}{18^2}\underline{\mathrm{L}}^1 L^{-3/2}\kappa^{-1/2}\sigma^{-1/2}\underline{\mathrm{L}}^1\Big)$$

$$= 1 - \sigma^{1/2}\underline{\mathrm{L}}^{-1/2}\Big(\frac{1}{2}\psi - \frac{1}{18^2}(\underline{\mathrm{L}}/L)^2\Big)$$

$$(\psi \leq \frac{1}{2}) = 1 - \sigma^{1/2}\underline{\mathrm{L}}^{-1/2}\Big(\frac{1}{24} - \frac{1}{18^2}\Big) \leq 1.$$

This completes the proof. □

9.5 Master inequality

We are finally in a position to bring together all the previous results together into a master inequality for the Lyapunov function ρ_k (defined in eq. (8.2.11)). After this lemma is proven,

we will prove that the right hand size is negative, which will imply that ρ_k linearly converges to 0 with rate β.

Lemma 23. Master inequality. *We have:*

$$\begin{aligned}
&\mathbb{E}_k[\rho_{k+1} - \beta\rho_k] \\
\leq &+ \|y_k\|^2 && \times\Big(1 - \beta - \sigma^{-1/2}S^{-1}\sigma(1-\psi)\Big) \\
&+ \|v_k - y_k\|^2 && \times\beta\Big(2\alpha^2\beta c_1 + S^{-1}\beta L^{1/2}\kappa^{-1/2}\psi - (1-\beta)\Big) \\
&+ f(y_k) && \times\Big(c - 2\sigma^{-1/2}S^{-1}\Big(\beta\alpha^{-1}(1-\alpha) + 1\Big)\Big) \\
&+ f(x_k) && \times\beta\Big(2\sigma^{-1/2}S^{-1}\alpha^{-1}(1-\alpha) - c\Big) \\
&+ \sum_{j=1}^{\tau}\|y_{k+1-j} - y_{k-j}\|^2 && \times S^{-1}L\kappa\psi^{-1}\tau\sigma^{1/2}\Big(2\sigma^{-1} + c\Big) - s \\
&+ \|\nabla f(\hat{y}_k)\|_*^2 && \times S^{-1}\left(\sigma^{-1} + 2\underline{L}^{-1}c_1 - \frac{1}{2}ch\left(2 - h\left(1 + \frac{1}{2}\sigma^{1/2}\underline{L}^{-1/2}\psi\right)\right)\right)
\end{aligned} \tag{9.5.1}$$

Proof.

$$\begin{aligned}
&\mathbb{E}_k\|v_{k+1}\|^2 - \beta\|v_k\|^2 \\
(9.1.4) &= (1-\beta)\|y_k\|^2 - \beta(1-\beta)\|v_k - y_k\|^2 + S^{-1}\sigma^{-1}\|\nabla f(\hat{y}_k)\|_*^2 \\
&\quad - 2\sigma^{-1/2}S^{-1}\langle y_k, \nabla f(\hat{y}_k)\rangle \\
&\quad - 2\sigma^{-1/2}S^{-1}\beta\alpha^{-1}(1-\alpha)\langle y_k - x_k, \nabla f(\hat{y}_k)\rangle \\
&\quad \leq (1-\beta)\|y_k\|^2 - \beta(1-\beta)\|v_k - y_k\|^2 + S^{-1}\sigma^{-1}\|\nabla f(\hat{y}_k)\|_*^2 \qquad (9.5.2) \\
(9.2.1) &+ 2\sigma^{-1/2}S^{-1}\left(-f(y_k) - \frac{1}{2}\sigma(1-\psi)\|y_k\|^2 + \frac{1}{2}L\kappa\psi^{-1}\tau\sum_{j=1}^{\tau}\|y_{k+1-j} - y_{k-j}\|^2\right) \\
(9.2.2) &- 2\sigma^{-1/2}S^{-1}\beta\alpha^{-1}(1-\alpha)(f(x_k) - f(y_k)) \\
&\quad + \sigma^{-1/2}S^{-1}\beta L\left(\kappa^{-1}\psi\beta\|v_k - y_k\|^2 + \kappa\psi^{-1}\beta^{-1}\tau\sum_{j=1}^{\tau}\|y_{k+1-j} - y_{k-j}\|^2\right)
\end{aligned}$$

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

$$\begin{aligned}
\leq &+ \|y_k\|^2 && \times\Big(1 - \beta - \sigma^{-1/2}S^{-1}\sigma(1-\psi)\Big) \\
&+ \|v_k - y_k\|^2 && \times\beta\Big(\sigma^{-1/2}S^{-1}\beta L\kappa^{-1}\psi - (1-\beta)\Big) \\
&- f(y_k) && \times 2\sigma^{-1/2}S^{-1}\Big(\beta\alpha^{-1}(1-\alpha) + 1\Big)
\end{aligned}$$

$$
\begin{aligned}
&+ f(x_k) && \times 2\sigma^{-1/2} S^{-1} \beta \alpha^{-1} (1-\alpha) \\
&+ \sum_{j=1}^{\tau} \|y_{k+1-j} - y_{k-j}\|^2 && \times 2\sigma^{-1/2} S^{-1} L \kappa \psi^{-1} \tau \\
&+ \|\nabla f(\hat{y}_k)\|_*^2 && \times \sigma^{-1} S^{-1}
\end{aligned}
$$

Now finally, we add the function-value and asynchronicity terms to our analysis. We use Lemma 21 is with $r = 1 - \sigma^{1/2} S^{-1}$, and

$$
s_i = \begin{cases} s = 6 S^{-1} L^{1/2} \kappa^{3/2} \psi^{-1} \tau, & 1 \le i \le \tau \\ 0, & i > \tau \end{cases} \tag{9.5.3}
$$

Notice that this choice of s_i will recover the coefficient formula given in (8.2.9). Hence we have:

$$
\begin{aligned}
&\mathbb{E}_k[cf(x_{k+1}) + A_{k+1} - \beta(cf(x_k) + A_k)] \\
&\text{(Lemma 20)} \le cf(y_k) - \frac{1}{2} ch\Big(2 - h\Big(1 + \frac{1}{2}\sigma^{1/2} \underline{\mathrm{L}}^{-1/2} \psi\Big)\Big) S^{-1} \|\nabla f(\hat{y}_k)\|_*^2 - \beta c f(x_k) \qquad (9.5.4) \\
&\quad + S^{-1} L \sigma^{1/2} \kappa \psi^{-1} \tau \sum_{j=1}^{\tau} \|y_{k+1-j} - y_{k-j}\|^2 \\
&\text{(Lemmas 21 and 22)} + c_1 \Big(2\alpha^2 \beta^2 \|v_k - y_k\|^2 + 2 S^{-1} \underline{\mathrm{L}}^{-1} \|\nabla f(\hat{y}_k)\|^2\Big) \qquad (9.5.5) \\
&\quad - \sum_{j=1}^{\infty} s_j \|y_{k+1-j} - y_{k-j}\|^2 + A_k(r - \beta)
\end{aligned}
$$

Notice $A_k(r - \beta) \le 0$. Finally, combining (9.5.2) and (9.5.4) completes the proof. □

In the next section, we will prove that every coefficient on the right hand side of (9.5.1) is 0 or less, which will complete the proof of Theorem 3.

9.6 Proof of main theorem

Lemma 24. The coefficients of $\|y_k\|^2$, $f(y_k)$, and $\sum_{j=1}^{\tau} \|y_{k+1-j} - y_{k-j}\|^2$ in Lemma 23 are non-positive.

Proof. The coefficient $1-(1-\psi)\sigma^{1/2}S^{-1}-\beta$ of $\|y_k\|^2$ is identically 0 via the definition (8.2.4) of β. 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 (8.2.12) of c.

First notice from the definition (8.2.12) of c:

$$\begin{aligned} c &= 2\sigma^{-1/2}S^{-1}\Big(\beta\alpha^{-1}(1-\alpha)+1\Big) \\ \text{(definitions of } \alpha,\beta) &= 2\sigma^{-1/2}S^{-1}\Big(\Big(1-\sigma^{1/2}S^{-1}(1-\psi)\Big)(1+\psi)\sigma^{-1/2}S+1\Big) \\ &= 2\sigma^{-1/2}S^{-1}\Big((1+\psi)\sigma^{-1/2}S+\psi^2\Big) \\ &= 2\sigma^{-1}\Big((1+\psi)+\psi^2\sigma^{1/2}S^{-1}\Big) \qquad (9.6.1) \\ c &\le 4\sigma^{-1} \qquad (9.6.2) \end{aligned}$$

Here the last line followed since $\psi\le\frac{1}{2}$ and $\sigma^{1/2}S^{-1}\le 1$. We now analyze the coefficient of $\sum_{j=1}^{\tau}\|y_{k+1-j}-y_{k-j}\|^2$.

$$\begin{aligned} & S^{-1}L\kappa\psi^{-1}\tau\sigma^{1/2}\Big(2\sigma^{-1}+c\Big)-s \\ (9.6.2) &\le 6L^{1/2}\kappa^{3/2}\psi^{-1}\tau-s \\ \text{(definition (9.5.3) of } s) &\le 0 \end{aligned}$$

□

Lemma 25. The coefficient $\beta\Big(2\sigma^{-1/2}S^{-1}\alpha^{-1}(1-\alpha)-c\Big)$ of $f(x_k)$ in Lemma 23 is non-positive.

Proof.

$$\begin{aligned} & 2\sigma^{-1/2}S^{-1}\alpha^{-1}(1-\alpha)-c \\ (9.6.1) &= 2\sigma^{-1/2}S^{-1}(1+\psi)\sigma^{-1/2}S-2\sigma^{-1}\Big((1+\psi)+\psi^2\sigma^{1/2}S^{-1}\Big) \\ &= 2\sigma^{-1}\Big((1+\psi)-\Big((1+\psi)+\psi^2\sigma^{1/2}S^{-1}\Big)\Big) \\ &= -2\psi^2\sigma^{-1/2}S^{-1}\le 0 \end{aligned}$$

□

Lemma 26. The coefficient $S^{-1}\Big(\sigma^{-1}+2\underline{L}^{-1}c_1-\frac{1}{2}ch\Big(2-h\Big(1+\frac{1}{2}\sigma^{1/2}\underline{L}^{-1/2}\psi\Big)\Big)\Big)$ of $\|\nabla f(\hat{y}_k)\|_*^2$ in Lemma 23 is non-positive.

Proof. We first need to bound c_1.

$$\begin{aligned}
((9.5.3) \text{ and } (8.2.9)) \quad c_1 &= s\sum_{j=1}^{\tau}\left(1-\sigma^{1/2}S^{-1}\right)^{-j} \\
(9.5.3) \quad &\leq 6S^{-1}L^{1/2}\kappa^{3/2}\psi^{-1}\tau\sum_{j=1}^{\tau}\left(1-\sigma^{1/2}S^{-1}\right)^{-j} \\
&\leq 6S^{-1}L^{1/2}\kappa^{3/2}\psi^{-1}\tau^2\left(1-\sigma^{1/2}S^{-1}\right)^{-\tau}
\end{aligned}$$

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

$$\begin{aligned}
&\leq 6S^{-1}L^{1/2}\kappa^{3/2}\psi^{-1}\tau^2\exp\left(\tau\sigma^{1/2}S^{-1}\right) \\
(\text{since } \psi \leq 3/7 \text{ and hence } \tau\sigma^{1/2}S^{-1} \leq \frac{1}{7}) &\leq S^{-1}L^{1/2}\kappa^{3/2}\psi^{-1}\tau^2 \times 6\exp\left(\frac{1}{7}\right) \\
c_1 &\leq 7S^{-1}L^{1/2}\kappa^{3/2}\psi^{-1}\tau^2 \qquad (9.6.3)
\end{aligned}$$

We now analyze the coefficient of $\|\nabla f(\hat{y}_k)\|_*^2$

$$\begin{aligned}
&\sigma^{-1} + 2\underline{\mathrm{L}}^{-1}c_1 - \frac{1}{2}ch\left(2 - h\left(1+\frac{1}{2}\sigma^{1/2}\underline{\mathrm{L}}^{-1/2}\psi\right)\right) \\
(9.6.3 \text{ and } 8.2.5) &\leq \sigma^{-1} + 14S^{-1}\underline{\mathrm{L}}^{-1}L^{1/2}\kappa^{3/2}\psi^{-1}\tau^2 - \frac{1}{2}ch\left(1+\frac{1}{4}\sigma^{1}\underline{\mathrm{L}}^{-1}\psi^2\right) \\
&\leq \sigma^{-1} + 14S^{-1}\underline{\mathrm{L}}^{-1}L^{1/2}\kappa^{3/2}\psi^{-1}\tau^2 - \frac{1}{2}ch \\
(\text{Equation (8.2.2) of } \psi) &= \sigma^{-1} + \frac{14}{81}\sigma^{-1}\psi - \frac{1}{2}ch \\
(9.6.1, \text{ Equation (8.2.5) of } h) &= \sigma^{-1}\left(1+\frac{14}{81}\psi - \left((1+\psi)+\psi^2\sigma^{1/2}S^{-1}\right)\left(1-\frac{1}{2}\sigma^{1/2}\underline{\mathrm{L}}^{-1/2}\psi\right)\right) \\
(\sigma^{1/2}\underline{\mathrm{L}}^{-1/2} \leq 0 \text{ and } \sigma^{1/2}S^{-1} \leq 1) &\leq \sigma^{-1}\left(1+\frac{14}{81}\psi - (1+\psi)\left(1-\frac{1}{2}\psi\right)\right) \\
&= \sigma^{-1}\psi\left(\frac{14}{81}+\frac{1}{2}\psi-\frac{1}{2}\right) \\
(\psi \leq \frac{1}{2}) &\leq 0
\end{aligned}$$

□

Lemma 27. The coefficient $\beta\left(2\alpha^2\beta c_1 + S^{-1}\beta L^{1/2}\kappa^{-1/2}\psi - (1-\beta)\right)$ of $\|v_k - y_k\|^2$ in 23 is non-positive.

Proof.

$$2\alpha^2\beta c_1 + \sigma^{1/2}S^{-1}\beta\psi - (1-\psi)\sigma^{1/2}S^{-1}$$

$$
\begin{aligned}
(9.6.3) &\leq 14\alpha^2\beta S^{-1}L^{1/2}\kappa^{3/2}\psi^{-1}\tau^2 + \sigma^{1/2}S^{-1}\beta\psi - (1-\psi)\sigma^{1/2}S^{-1} \\
&\leq 14\sigma S^{-3}L^{1/2}\kappa^{3/2}\psi^{-1}\tau^2 + \sigma^{1/2}S^{-1}\psi - (1-\psi)\sigma^{1/2}S^{-1} \\
&= \sigma^{1/2}S^{-1}\Big(14S^{-2}L\kappa\tau^2\psi^{-1} + 2\psi - 1\Big)
\end{aligned}
$$

Here the last inequality follows since $\beta \leq 1$ and $\alpha \leq \sigma^{1/2}S^{-1}$. We now rearrange the definition of ψ to yield the identity:

$$
S^{-2}\kappa = \frac{1}{9^4}\underline{\mathrm{L}}^2 L^{-3}\tau^{-4}\psi^4
$$

Using this, we have:

$$
\begin{aligned}
&14S^{-2}L\kappa\tau^2\psi^{-1} + 2\psi - 1 \\
&= \frac{14}{9^4}\underline{\mathrm{L}}^2 L^{-2}\psi^3\tau^{-2} + 2\psi - 1 \\
&\leq \frac{14}{9^4}\Big(\frac{3}{7}\Big)^3 1^{-2} + \frac{6}{7} - 1 \leq 0
\end{aligned}
$$

Here the last line followed since $\underline{\mathrm{L}} \leq L$, $\psi \leq \frac{3}{7}$, and $\tau \geq 1$. Hence the proof is complete. □

Proof of Theorem 3. Using the master inequality 23 in combination with the previous Lemmas 24, 25, 26, and 27, we have:

$$
\mathbb{E}_k[\rho_{k+1}] \leq \beta\rho_k = \Big(1 - (1-\psi)\sigma^{1/2}S^{-1}\Big)\rho_k
$$

When we have:

$$
\Big(1 - (1-\psi)\sigma^{1/2}S^{-1}\Big)^k \leq \epsilon
$$

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

$$
\begin{aligned}
K(\epsilon)\ln\Big(1 - (1-\psi)\sigma^{1/2}S^{-1}\Big) &= \ln(\epsilon) \\
K(\epsilon) &= \frac{-1}{\ln(1 - (1-\psi)\sigma^{1/2}S^{-1})}\ln(1/\epsilon)
\end{aligned}
$$

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

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

$$\frac{-1}{\ln(1-x)} = \frac{1}{x} + \mathcal{O}(1)$$

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

$$K(\epsilon) = \frac{1}{1-\psi}\Big(\sigma^{-1/2}S + \mathcal{O}(1)\Big)\ln(1/\epsilon)$$

An expression for $K_{\texttt{NU\_ACDM}}(\epsilon)$, the complexity of `NU_ACDM` follows by similar reasoning.

$$K_{\texttt{NU\_ACDM}}(\epsilon) = \Big(\sigma^{-1/2}S + \mathcal{O}(1)\Big)\ln(1/\epsilon) \tag{9.6.4}$$

Finally we have:

$$\begin{aligned} K(\epsilon) &= \frac{1}{1-\psi}\left(\frac{\sigma^{-1/2}S + \mathcal{O}(1)}{\sigma^{-1/2}S + \mathcal{O}(1)}\right)K_{\texttt{NU\_ACDM}}(\epsilon) \\ &= \frac{1}{1-\psi}(1+o(1))K_{\texttt{NU\_ACDM}}(\epsilon) \end{aligned}$$

which completes the proof. □

CHAPTER 10

Optimality proof

In this section, we prove Theorem 4. 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:

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

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

$$f_i = \frac{L_i - \sigma}{4}\Big(\frac{1}{2}\langle x, A_i x\rangle - \langle e_1, x\rangle\Big) + \frac{\sigma}{2}\|x\|^2$$

which is clearly σ-strongly convex and L_i-Lipschitz on $\mathbb{R}^b$. From Lemma 8 of (Lan and Y. Zhou 2017), we know that this function has unique minimizer

$$x_{*,(i)} \triangleq \Big(q_i, q_i^2, \ldots, q_i^b\Big), \text{ for } q = \frac{\kappa_i^{1/2}-1}{\kappa_i^{1/2}+1}.$$

Finally, we define f via:

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

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

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

$$V(I_1, \ldots, I_n) = V_1(I_1) \oplus \ldots \oplus V_n(I_n).$$

$V(I_1, \ldots, I_n)$ 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 $\mathrm{IC}(V(I_1, \ldots, I_n)) = V(I_1, \ldots, I_n)$. Also, clearly, we have:

$$\nabla_i f(V(I_1, \ldots, I_n)) \subset V(0, \ldots, 0, \min\{I_i + 1, b\}, 0, \ldots, 0). \tag{10.0.1}$$

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

Let the initial point x_0 belong to $V\left(\bar{I}_1, \ldots, \bar{I}_n\right)$. 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 17 using (10.0.1), we have:

$$x_k \in V\left(\min\left\{\bar{I}_1 + I_{k,1}, b\right\}, \ldots, \min\left\{\bar{I}_n + I_{k,m}, b\right\}\right)$$

Hence if $x_{0,(i)} \in V_i(0)$ and $k \leq b$, then

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

Therefore for all i, we have:

$$\mathbb{E}\|x_k - x_*\|^2 \geq \mathbb{E}\left\|x_{k,(i)} - x_{*,(i)}\right\|^2 \geq \mathbb{E}\left[\left(q_i^{2I_{k,i}+2} - q_i^{2b+2}\right) / \left(1 - q_i^2\right)\right]$$

To evaluate this expectation, we note:

$$\begin{aligned}
\mathbb{E} q_i^{2I_{k,i}} &= \sum_{j=0}^{k} \binom{k}{j} p_i^j (1-p_i)^{k-j} q_i^{2j} \\
&= (1-p_i)^k \sum_{j=0}^{k} \binom{k}{j} \left(q_i^2 p_i (1-p_i)^{-1}\right)^j \\
&= (1-p_i)^k \left(1 + q_i^2 p_i (1-p_i)^{-1}\right)^k \\
&= \left(1 - \left(1 - q_i^2\right) p_i\right)^k
\end{aligned}$$

Hence

$$\mathbb{E}\|x_k - x_*\|^2 \geq \left(\left(1 - \left(1 - q_i^2\right)p_i\right)^k - q_i^{2b}\right)q_i^2/\left(1 - q_i^2\right).$$

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

$$x_{0,(j)} = (1 - \delta_{ij})x_{*,(j)}$$

For such a choice of x_0, we have

$$\|x_0 - x_*\|^2 = \left\|x_{*,(i)}\right\|^2 = q_i^2 + \ldots + q_i^{2b} = q_i^2\frac{1 - q_i^{2b}}{1 - q_i^2}$$

Hence for this choice of x_0:

$$\mathbb{E}\|x_k - x_*\|^2/\|x_0 - x_*\|^2 \geq \left(\left(1 - \left(1 - q_i^2\right)p_i\right)^k - q_i^{2b}\right)/\left(1 - q_i^{2b}\right)$$

Now notice:

$$\left(1 - \left(1 - q_i^2\right)p_i\right)^k = \left(q_i^{-2} - \left(q_i^{-2} - 1\right)p_i\right)^k q_i^{2k} \geq q_i^{2k}$$

Hence

$$\mathbb{E}\|x_k - x_*\|^2/\|x_0 - x_*\|^2 \geq \left(1 - \left(1 - q_i^2\right)p_i\right)^k\left(1 - q_i^{2b-2k}\right)/\left(1 - q_i^{2b}\right)$$

Now if we let $b = 2k$, then we have:

$$\begin{aligned}\mathbb{E}\|x_k - x_*\|^2/\|x_0 - x_*\|^2 &\geq \left(1 - \left(1 - q_i^2\right)p_i\right)^k\left(1 - q_i^{2k}\right)/\left(1 - q_i^{4k}\right)\\ &= \left(1 - \left(1 - q_i^2\right)p_i\right)^k/\left(1 + q_i^{2k}\right)\\ \mathbb{E}\|x_k - x_*\|^2/\|x_0 - x_*\|^2 &\geq \frac{1}{2}\max_i\left(1 - \left(1 - q_i^2\right)p_i\right)^k\end{aligned}$$

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:

$$p_i = \left(1 - q_i^2\right)^{-1}/\left(\sum_{j=1}^n\left(1 - q_j^2\right)^{-1}\right)$$

Hence

$$\mathbb{E}\|x_k - x_*\|^2/\|x_0 - x_*\|^2 \geq \frac{1}{2}\left(1 - \left(\sum_{j=1}^n\left(1 - q_j^2\right)^{-1}\right)^{-1}\right)^k$$

$$\sum_{j=1}^{n}\left(1-q_j^2\right)^{-1} = \frac{1}{4}\sum_{j=1}^{n}\left(\kappa_i^{1/2}+2+\kappa_i^{-1/2}\right)$$
$$\geq \frac{1}{4}\left(\sum_{j=1}^{n}\kappa_i^{1/2}+2n\right)$$
$$\mathbb{E}\|x_k-x_*\|^2/\|x_0-x_*\|^2 \geq \frac{1}{2}\left(1-\frac{4}{\sum_{j=1}^{n}\kappa_i^{1/2}+2n}\right)^k$$

Hence the complexity $I(\epsilon)$ satisfies:

$$\epsilon \geq \frac{1}{2}\left(1-\frac{4}{\sum_{j=1}^{n}\kappa_i^{1/2}+2n}\right)^{I(\epsilon)}$$
$$I(\epsilon) \geq -\left(\ln\left(1-\frac{4}{\sum_{j=1}^{n}\kappa_i^{1/2}+2n}\right)\right)^{-1}\ln(1/2\epsilon)$$
$$= \frac{1}{4}(1+o(1))\left(n+\sum_{j=1}^{n}\kappa_i^{1/2}\right)\ln(1/2\epsilon)$$

CHAPTER 11

ODE Analysis of Acceleration

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, Boyd, and Candes 2014). For simplicity, assume $L_i = L$, $\forall i$. We rescale (I.e. we replace $f(x)$ with $\frac{1}{\sigma}f$.) 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[1] (see Section (11.1)):

$$\ddot{Y} + 2n^{-1}\kappa^{-1/2}\dot{Y} + 2n^{-2}\kappa^{-1}\nabla f(Y) = 0 \tag{11.0.1}$$

We define the parameter $\eta \triangleq n\kappa^{1/2}$, and the energy: $E(t) = e^{n^{-1}\kappa^{-1/2}t}(f(Y) + \frac{1}{4}\left\|Y + \eta\dot{Y}\right\|^2)$. This is very similar to the Lyapunov function discussed in (8.2.11), with $\frac{1}{4}\left\|Y(t) + \eta\dot{Y}(t)\right\|^2$ 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 3, we can derive a linear convergence result with a similar rate. See Section 11.2.

Lemma 28. *If Y satisfies (11.0.1), the energy satisfies $E'(t) \leq 0$, $E(t) \leq E(0)$, and hence:*

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

We may also analyze an asynchronous version of (11.0.1) to motivate the proof of our main theorem. Here $\hat{Y}(t)$ is a delayed version of $Y(t)$ with the delay bounded by τ.

$$\ddot{Y} + 2n^{-1}\kappa^{-1/2}\dot{Y} + 2n^{-2}\kappa^{-1}\nabla f\left(\hat{Y}\right) = 0, \tag{11.0.2}$$

Unfortunately, this energy satisfies (see Section (11.4), (11.4.1)):

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

[1] For compactness, we have omitted the (t) from time-varying functions $Y(t)$, $\dot{Y}(t)$, $\nabla Y(t)$, etc.

Hence this energy $E(t)$ may not be decreasing in general. But, we may add a continuous-time **asynchronicity error** (see (T. Sun, Hannah, and Wotao Yin 2017)), much like in Definition 15, to create a decreasing energy. Let $c_0 \geq 0$ and $r > 0$ be arbitrary constants that will be set later. Define:

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

Lemma 29. *When $r\tau \leq \frac{1}{2}$, the asynchronicity error $A(t)$ satisfies:*

$$e^{-rt}\frac{d}{dt}\left(e^{rt}A(t)\right) \leq c_0\left\|\dot{Y}(t)\right\|^2 - \frac{1}{2}\tau^{-1}c_0 D(t).$$

See Section 11.3 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 3 (see Lemma 21). 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 $\left\|\dot{Y}(t)\right\|^2$. This restores monotonicity.

Theorem 5. *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:*

$$e^{-\eta^{-1}t}\frac{d}{dt}\left(E(t) + e^{\eta^{-1}t}A(t)\right) \leq 0. \tag{11.0.3}$$

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

Notice how this convergence condition is similar to Corollary 16, but a little looser. The convergence condition in Theorem 3 can actually be improved to approximately match this.

Proof.
$$\begin{aligned} e^{-\eta^{-1}t}\frac{d}{dt}\left(E(t) + e^{\eta^{-1}t}A(t)\right) &\leq \left(c_0 - \frac{1}{8}\eta\right)\left\|\dot{Y}\right\|^2 + \left(3\kappa^2\eta^{-1}\tau - \frac{1}{2}\tau^{-1}c_0\right)D(t) \\ &= 6\eta^{-1}\kappa^2\left(\tau^2 - \frac{1}{48}n^2\kappa^{-1}\right)\left\|\dot{Y}\right\|^2 \leq 0 \end{aligned}$$ □

The preceding should hopefully elucidate the logic and general strategy of the proof of Theorem 3.

11.1 Derivation of ODE for synchronous `A2BCD`

If we take expectations with respect to $\mathbb{E}_k$, then synchronous (no delay) `A2BCD` becomes:

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

$$\mathbb{E}_k x_{k+1} = y_k - n^{-1}\kappa^{-1}\nabla f(y_k)$$
$$\mathbb{E}_k v_{k+1} = \beta v_k + (1-\beta)y_k - n^{-1}\kappa^{-1/2}\nabla f(y_k)$$

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

$$y_k = \alpha v_k + (1-\alpha)x_k \tag{11.1.1}$$
$$x_{k+1} = y_k - s^{1/2}\kappa^{-1/2}\eta^{-1}\nabla f(y_k) \tag{11.1.2}$$
$$v_{k+1} = \beta v_k + (1-\beta)y_k - s^{1/2}\eta^{-1}\nabla f(y_k) \tag{11.1.3}$$

for coefficients:

$$\alpha = \left(1 + s^{-1/2}\eta\right)^{-1}$$
$$\beta = 1 - s^{1/2}\eta^{-1}$$

s is a discretization scale parameter that will be sent to 0 to obtain an ODE analogue of synchronous `A2BCD`. We first use (9.1.3) to eliminate v_k from from (11.1.3).

$$\begin{aligned}
0 &= -v_{k+1} + \beta v_k + (1-\beta)y_k - s^{1/2}\eta^{-1}\nabla f(y_k) \\
0 &= -\alpha^{-1}y_{k+1} + \alpha^{-1}(1-\alpha)x_{k+1} \\
&\quad + \beta\left(\alpha^{-1}y_k - \alpha^{-1}(1-\alpha)x_k\right) + (1-\beta)y_k - s^{1/2}\eta^{-1}\nabla f(y_k) \\
\text{(times by } \alpha) \quad 0 &= -y_{k+1} + (1-\alpha)x_{k+1} \\
&\quad + \beta(y_k - (1-\alpha)x_k) + \alpha(1-\beta)y_k - \alpha s^{1/2}\eta^{-1}\nabla f(y_k) \\
&= -y_{k+1} + y_k(\beta + \alpha(1-\beta)) \\
&\quad + (1-\alpha)x_{k+1} - x_k\beta(1-\alpha) - \alpha s^{1/2}\eta^{-1}\nabla f(y_k)
\end{aligned}$$

We now eliminate x_k using (11.1.1):

$$\begin{aligned}
0 &= -y_{k+1} + y_k(\beta + \alpha(1-\beta)) \\
&\quad + (1-\alpha)\left(y_k - s^{1/2}\eta^{-1}\kappa^{-1/2}\nabla f(y_k)\right) - \left(y_{k-1} - s^{1/2}\eta^{-1}\kappa^{-1/2}\nabla f(y_{k-1})\right)\beta(1-\alpha) \\
&\quad - \alpha s^{1/2}\eta^{-1}\nabla f(y_k)
\end{aligned}$$

$$
\begin{aligned}
&= -y_{k+1} + y_k(\beta + \alpha(1-\beta) + (1-\alpha)) - \beta(1-\alpha)y_{k-1} \\
&+ s^{1/2}\eta^{-1}\nabla f(y_{k-1})(\beta-1)(1-\alpha) \\
&- \alpha s^{1/2}\eta^{-1}\nabla f(y_k) \\
&= (y_k - y_{k+1}) + \beta(1-\alpha)(y_k - y_{k-1}) \\
&+ s^{1/2}\eta^{-1}(\nabla f(y_{k-1})(\beta-1)(1-\alpha) - \alpha\nabla f(y_k))
\end{aligned}
$$

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

$$
\begin{aligned}
0 &= (y_k - y_{k+1}) + \beta(1-\alpha)(y_k - y_{k-1}) \\
&\quad + s^{1/2}\eta^{-1}((\beta-1)(1-\alpha) - \alpha)\nabla f(y_k) + \mathcal{O}\left(s^{3/2}\right) \\
0 &= \left(-s^{1/2}\dot{Y} - \frac{1}{2}s\ddot{Y}\right) + \beta(1-\alpha)\left(s^{1/2}\dot{Y} - \frac{1}{2}s\ddot{Y}\right) \\
&\quad + s^{1/2}\eta^{-1}((\beta-1)(1-\alpha) - \alpha)\nabla f(y_k) + \mathcal{O}\left(s^{3/2}\right)
\end{aligned}
\tag{11.1.4}
$$

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

$$
\begin{aligned}
\beta(1-\alpha) &= \left(1 - s^{1/2}\eta^{-1}\right)\left(1 - \frac{1}{1 + s^{-1/2}\eta}\right) \\
&= \left(1 - s^{1/2}\eta^{-1}\right)\frac{s^{-1/2}\eta}{1 + s^{-1/2}\eta} \\
&= \frac{s^{-1/2}\eta - 1}{s^{-1/2}\eta + 1} \\
&= \frac{1 - s^{1/2}\eta^{-1}}{1 + s^{1/2}\eta^{-1}} \\
&= 1 - 2s^{1/2}\eta^{-1} + \mathcal{O}(s)
\end{aligned}
$$

We also have:

$$
\begin{aligned}
(\beta-1)(1-\alpha) - \alpha &= \beta(1-\alpha) - 1 \\
&= -2s^{1/2}\eta^{-1} + \mathcal{O}(s)
\end{aligned}
$$

Hence using these facts on (11.1.4), we have:

$$
0 = \left(-s^{1/2}\dot{Y} - \frac{1}{2}s\ddot{Y}\right) + \left(1 - 2s^{1/2}\eta^{-1} + \mathcal{O}(s)\right)\left(s^{1/2}\dot{Y} - \frac{1}{2}s\ddot{Y}\right)
$$

$$
\begin{aligned}
&+ s^{1/2}\eta^{-1}\Big(-2s^{1/2}\eta^{-1} + \mathcal{O}(s)\Big)\nabla f(y_k) + \mathcal{O}\Big(s^{3/2}\Big) \\
0 &= -s^{1/2}\dot{Y} - \frac{1}{2}s\ddot{Y} + \Big(s^{1/2}\dot{Y} - \frac{1}{2}s\ddot{Y} - 2s^1\eta^{-1}\dot{Y} + \mathcal{O}\Big(s^{3/2}\Big)\Big) \\
&\Big(-2s^1\eta^{-2} + \mathcal{O}\Big(s^{3/2}\Big)\Big)\nabla f(y_k) + \mathcal{O}\Big(s^{3/2}\Big) \\
0 &= -s\ddot{Y} - 2s\eta^{-1}\dot{Y} - 2s\eta^{-2}\nabla f(y_k) + \mathcal{O}\Big(s^{3/2}\Big) \\
0 &= -\ddot{Y} - 2\eta^{-1}\dot{Y} - 2\eta^{-2}\nabla f(y_k) + \mathcal{O}\Big(s^{1/2}\Big)
\end{aligned}
$$

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

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

11.2 Convergence proof for synchronous ODE

$$
\begin{aligned}
e^{-\eta^{-1}t}E'(t) &= \Big\langle \nabla f(Y(t)), \dot{Y}(t) \Big\rangle + \eta^{-1} f(Y(t)) \\
&\quad + \frac{1}{2}\Big\langle Y(t) + \eta\dot{Y}(t), \dot{Y}(t) + \eta\ddot{Y}(t) \Big\rangle + \eta^{-1}\frac{1}{4}\Big\|Y(t) + \eta\dot{Y}(t)\Big\|^2 \\
\text{(strong convexity (3.1.1))} &\leq \Big\langle \nabla f(Y), \dot{Y} \Big\rangle + \eta^{-1}\langle \nabla f(Y), Y \rangle - \frac{1}{2}\eta^{-1}\|Y\|^2 \\
&\quad + \frac{1}{2}\Big\langle Y + \eta\dot{Y}, -\dot{Y} - 2\eta^{-1}\nabla f(Y) \Big\rangle + \eta^{-1}\frac{1}{4}\Big\|Y(t) + \eta\dot{Y}(t)\Big\|^2 \\
&= -\frac{1}{4}\eta^{-1}\|Y\|^2 - \frac{1}{4}\eta\Big\|\dot{Y}\Big\|^2 \leq 0
\end{aligned}
$$

Hence we have $E'(t) \leq 0$. Therefore $E(t) \leq E(0)$. That is:

$$
E(t) = e^{n^{-1}\kappa^{-1/2}t}\Big(f(Y) + \frac{1}{4}\Big\|Y + \eta\dot{Y}\Big\|^2\Big) \leq E(0) = f(Y(0)) + \frac{1}{4}\Big\|Y(0) + \eta\dot{Y}(0)\Big\|^2 \tag{11.2.1}
$$

which implies:

$$
f(Y(t)) + \frac{1}{4}\Big\|Y(t) + \eta\dot{Y}(t)\Big\|^2 \leq e^{-n^{-1}\kappa^{-1/2}t}\Big(f(Y(0)) + \frac{1}{4}\Big\|Y(0) + \eta\dot{Y}(0)\Big\|^2\Big) \tag{11.2.2}
$$

11.3 Asynchronicity error lemma

This result is the continuous-time analogue of Lemma 21. First notice that $c(0) = c_0$ and $c(\tau) = 0$. We also have:

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

$$
\begin{aligned}
&= -r\left(e^{-rt} + e^{-rt}\frac{e^{-r\tau}}{1-e^{-r\tau}}\right) \\
&= -r\left(e^{-rt} + \left(e^{-rt}-1\right)\frac{e^{-r\tau}}{1-e^{-r\tau}} + \frac{e^{-r\tau}}{1-e^{-r\tau}}\right) \\
c'(t) &= -rc(t) - rc_0\frac{e^{-r\tau}}{1-e^{-r\tau}}
\end{aligned}
$$

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

$$
\begin{aligned}
A'(t) &= c_0\left\|\dot{Y}(t)\right\|^2 + \int_{t-\tau}^{t} c'(t-s)\left\|\dot{Y}(s)\right\|^2 ds \\
&= c_0\left\|\dot{Y}(t)\right\|^2 - rA(t) - rc_0\frac{e^{-r\tau}}{1-e^{-r\tau}}D(t)
\end{aligned}
$$

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

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

and the result easily follows.

11.4 Convergence analysis for the asynchronous ODE

We consider the same energy as in the synchronous case (that is, the ODE in (11.0.1)). Similar to before, we have:

$$
\begin{aligned}
e^{-\eta^{-1}t}E'(t) &\leq \left\langle \nabla f(Y), \dot{Y}\right\rangle + \eta^{-1}\langle \nabla f(Y), Y\rangle - \frac{1}{2}\eta^{-1}\|Y\|^2 \\
&+ \frac{1}{2}\left\langle Y + \eta\dot{Y}, -\dot{Y} - 2\eta^{-1}\nabla f\left(\hat{Y}\right)\right\rangle + \eta^{-1}\frac{1}{4}\left\|Y(t) + \eta\dot{Y}(t)\right\|^2 \\
&= \left\langle \nabla f(Y), \dot{Y}\right\rangle + \eta^{-1}\langle \nabla f(Y), Y\rangle - \frac{1}{2}\eta^{-1}\|Y\|^2 \\
&+ \frac{1}{2}\left\langle Y + \eta\dot{Y}, -\dot{Y} - 2\eta^{-1}\nabla f(Y)\right\rangle + \eta^{-1}\frac{1}{4}\left\|Y(t) + \eta\dot{Y}(t)\right\|^2 \\
&- \eta^{-1}\left\langle Y + \eta\dot{Y}, \nabla f\left(\hat{Y}\right) - \nabla f(Y)\right\rangle \\
&= -\frac{1}{4}\eta^{-1}\|Y\|^2 - \frac{1}{4}\eta\left\|\dot{Y}\right\|^2 - \eta^{-1}\left\langle Y + \eta\dot{Y}, \nabla f\left(\hat{Y}\right) - \nabla f(Y)\right\rangle
\end{aligned}
$$

where the final equality follows from the proof in Section 11.2. Hence

$$
e^{-\eta^{-1}t}E'(t) \leq -\frac{1}{4}\eta^{-1}\|Y\|^2 - \frac{1}{4}\eta\left\|\dot{Y}\right\|^2 + L\eta^{-1}\|Y\|\left\|\hat{Y} - Y\right\| + L\left\|\dot{Y}\right\|\left\|\hat{Y} - Y\right\| \tag{11.4.1}
$$

Now we present an inequality that is similar to Lemma 18.

Lemma 30. *Let $A, \chi > 0$. Then:*

$$\left\|Y(t) - \hat{Y}(t)\right\| A \leq \frac{1}{2}\chi\tau D(t) + \frac{1}{2}\chi^{-1}A^2$$

Proof. Since $\hat{Y}(t)$ is a delayed version of $Y(t)$, we have: $\hat{Y}(t) = Y(t - j(t))$ 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

$$\begin{aligned}
X(t) - \hat{X}(t) &= \int_{s=t-j(t)}^{t} X'(s)ds \\
\left\|X(t) - \hat{X}(t)\right\| A &= \left\|\int_{s=t-j(t)}^{t} X'(s)ds\right\| A \\
&\leq \frac{1}{2}\chi\left\|\int_{s=t-j(t)}^{t} X'(s)ds\right\|^2 + \frac{1}{2}\chi^{-1}A^2 \\
\text{(Holder's inequality)} &\leq \frac{1}{2}\chi\left(\int_{s=t-j(t)}^{t}\|X'(s)\|^2 ds\right)\left(\int_{s=t-j(t)}^{t} 1ds\right) + \frac{1}{2}\chi^{-1}A^2 \\
&\leq \frac{1}{2}\chi\tau\left(\int_{s=t-j(t)}^{t}\|X'(s)\|^2 ds\right) + \frac{1}{2}\chi^{-1}A^2
\end{aligned}$$

□

We use this lemma twice on $\|Y\|\left\|\hat{Y} - Y\right\|$ and $\left\|\dot{Y}\right\|\left\|\hat{Y} - Y\right\|$ in (11.4.1) with $\chi = 2L, A = \|Y\|$ and $\chi = 4L\eta^{-1}, A = \left\|\dot{Y}\right\|$ respectively, to yield:

$$\begin{aligned}
e^{-\eta^{-1}t}E'(t) &\leq -\frac{1}{4}\eta^{-1}\|Y\|^2 - \frac{1}{4}\eta\left\|\dot{Y}\right\|^2 \\
&\quad + L\eta^{-1}\left(L\tau D(t) + \frac{1}{4}L^{-1}\|Y\|^2\right) + L\left(2L\eta^{-1}\tau D(t) + \frac{1}{8}L^{-1}\eta\left\|\dot{Y}\right\|^2\right) \\
&= -\frac{1}{8}\eta\left\|\dot{Y}\right\|^2 + 3L^2\eta^{-1}\tau D(t)
\end{aligned}$$

The proof of convergence is completed in Chapter 11.

CHAPTER 12

Numerical Results on Acceleration

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

$$\min_{w\in\mathbb{R}^d} P(w) = \frac{1}{2n}\left\|A^T w - l\right\|^2 + \frac{\lambda}{2}\|w\|^2, \min_{\alpha\in\mathbb{R}^n} D(\alpha) = \frac{1}{2d^2\lambda}\|A\alpha\|^2 + \frac{1}{2d}\|\alpha + l\|^2 \tag{12.0.1}$$

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 (8.2.1), 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 data sets `w1a` (47272 samples, 300 features), `wxa` which combines the data from from `w1a` to `w8a` (293201 samples, 300 features), and `aloi` (108000 samples, 128 features) from LIBSVM (Chang and C.-J. 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 12.2 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 and Sidford 2013). 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 $A_i^T Ay$, 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 (8.2.6). Hence Ay must be recalculated from scratch.

However (Lee and Sidford 2013) 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. (8.2.6)) 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 12.3.

In Table 12.1, we plot the sub-optimality vs. time for decreasing values of λ, which corresponds to increasingly large condition numbers κ. When κ 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 κ has unnecessary overhead, which means async-RBCD can be quite competitive. When κ 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 $4-5\times$ 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.

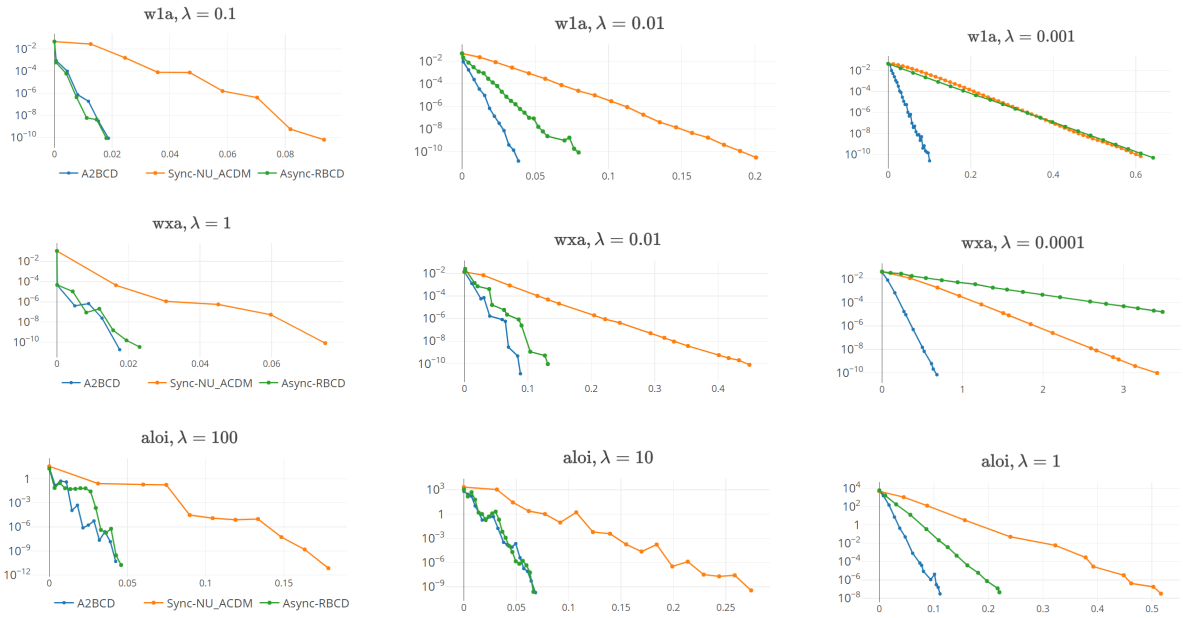

Table 12.1: Sub-optimality $f(y_k) - f(x_*)$ (y-axis) vs time in seconds (x-axis) for `A2BCD`, synchronous `NU_ACDM`, and asynchronous RBCD for data sets `w1a`, `wxa` and `aloi` for various values of λ.

12.1 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 $\nabla_{i(k)} f(\hat{y}_k)$, 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 $\nabla_{i(k)} f(\hat{y}_k)$, updating y_k and v_k is extremely fast (x_k can simply be eliminated). Hence it is natural to simply store y_k and v_k centrally, and update them when the delayed gradients $\nabla_{i(k)} f(\hat{y}_k)$. 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 $\nabla_{i(k)} f(\hat{y}_k)$ 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.

12.2 Parameter selection and tuning

When defining the coefficients, σ 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 $\nabla_{i(k)} f(\hat{y}_k)$ only needs to be calculated once per iteration. A larger (or overestimated) maximum delay τ will cause a larger asynchronicity parameter ψ, which leads to more conservative step sizes to compensate.

To estimate ψ, one can first performed a dry run with all coefficient set to 0 to estimate τ. All function parameters can be calculated exactly for this problem in terms of the data matrix and λ. We can then use these parameters and this tau to calculate ψ. ψ and τ 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 σ, the strong convexity modulus (Arjevani 2017). Ideally σ 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, Schmidt, and Bach 2012) Section 4 can be used.

12.3 Sparse update formulation

As mentioned in previously, authors in (Lee and Sidford 2013) 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 `A2BCD`,

and derive the equivalent iteration below:

$$\begin{pmatrix} y_{k+1} \\ v_{k+1} \end{pmatrix} = \begin{pmatrix} 1-\alpha\beta, & \alpha\beta \\ 1-\beta, & \beta \end{pmatrix} \begin{pmatrix} y_k \\ v_k \end{pmatrix} - \begin{pmatrix} \left(\alpha\sigma^{-1/2}L_{i_k}^{-1/2} + h(1-\alpha)L_{i_k}^{-1}\right)\nabla_{i(k)}f\left(\hat{y}^k\right) \\ \left(\sigma^{-1/2}L_{i(k)}^{-1/2}\right)\nabla_{i(k)}f\left(\hat{y}^k\right) \end{pmatrix}$$
$$\triangleq C\begin{pmatrix} y_k \\ v_k \end{pmatrix} - Q_k$$

where C and Q_k are defined in the obvious way. Hence we define auxiliary variables p_k, q_k defined via:

$$\begin{pmatrix} y_k \\ v_k \end{pmatrix} = C^k \begin{pmatrix} p_k \\ q_k \end{pmatrix} \tag{12.3.1}$$

These clearly follow the iteration:

$$\begin{pmatrix} p_{k+1} \\ q_{k+1} \end{pmatrix} = \begin{pmatrix} p_k \\ q_k \end{pmatrix} - C^{-(k+1)}Q_k \tag{12.3.2}$$

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 12.3.1.

The gradient of the dual function is given by:

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

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 (12.3.1), 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 12.1. 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 $\mathcal{O}(d)$).

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 $\hat{p}, \hat{q}, \hat{Ap}, \hat{Aq}$ respectively. We also find it convenient to define:

$$\begin{bmatrix} D_1^k \\ D_2^k \end{bmatrix} = \begin{bmatrix} \alpha\sigma^{-1/2}L_{i(k)}^{-1/2} + h(1-\alpha)L_{i(k)}^{-1} \\ \sigma^{-1/2}L_{i(k)}^{-1/2} \end{bmatrix} \tag{12.3.3}$$

Algorithm 1 Shared-memory implementation of A2BCD

Inputs: Function parameters A, λ, L, $\{L_i\}_{i=1}^n$, n, d. Delay τ (obtained in dry run). Starting vectors y, v.

Shared data: Solution vectors p, q; auxiliary vectors Ap, Aq; sparsifying matrix B

Node local data: Solution vectors $\hat{p}$, $\hat{q}$, auxiliary vectors $\hat{A}p$, $\hat{A}q$, sparsifying matrix $\hat{B}$.

Calculate parameters ψ, α, β, h via 14. Set $k = 0$.

Initializations: $p \leftarrow y$, $q \leftarrow v$, $Ap \leftarrow Ay$, $Aq \leftarrow Av$, $B \leftarrow I$.

while not converged, each computing node asynchronous **do**

  Randomly select block i via (8.2.1).

  Read shared data into local memory: $\hat{p} \leftarrow p$, $\hat{q} \leftarrow q$, $\hat{A}p \leftarrow Ap$, $\hat{A}q \leftarrow Aq$, $\hat{B} \leftarrow B$.

  Compute block gradient: $\nabla_i f(\hat{y}) = \frac{1}{n\lambda}\Big(\frac{1}{n}A_i^T\Big(\hat{B}_{1,1}\hat{A}p + \hat{B}_{1,2}\hat{A}q\Big) + \lambda\Big(\hat{B}_{1,1}\hat{p} + \hat{B}_{1,2}\hat{q}\Big)\Big)$

  Compute quantity $g_i = A_i^T \nabla_i f(\hat{y})$

  Shared memory updates:

  Update $B \leftarrow \begin{bmatrix} 1-\alpha\beta & \alpha\beta \\ 1-\beta & \beta \end{bmatrix} \times B$, calculate inverse $b \leftarrow B^{-1}$.

  $$\begin{bmatrix} p \\ q \end{bmatrix} \; -= b \begin{bmatrix} D_1^k \\ D_2^k \end{bmatrix} \otimes \nabla_i f(\hat{y}) \;, \qquad \begin{bmatrix} Ap \\ Aq \end{bmatrix} \; -= b \begin{bmatrix} D_1^k \\ D_2^k \end{bmatrix} \otimes g_i$$

  Increase iteration count: $k \leftarrow k + 1$

end while

Recover original iteration variables: $\begin{bmatrix} y \\ v \end{bmatrix} \leftarrow B \begin{bmatrix} p \\ q \end{bmatrix}$. Output y.

Part IV

Weak Convergence Under Unbounded Delay

In this part we present weak convergence results for ARock under unbounded delays from (Hannah and Wotao Yin 2017b).

CHAPTER 13

Proof of Convergence for Stochastic Unbounded Delays

13.1 Main Result

The first result in this part is the convergence of ARock under stochastic, potentially unbounded delays. First we precisely define the assumptions on the delay:

Definition 31. Evenly old delays. We say that delays are "evenly old" if there exists some constant B such that, with probability 1, we have $|j(k,i) - j(k,l)| \leq B$ for all $k \in \mathbb{N}$, $1 \leq i \leq m$, and $1 \leq l \leq m$.

Delays can be arbitrarily large, but the ages of the various block are similar if they are evenly old. Clearly, if we have **bounded delay** (that is, with probability 1 we have $j(k) \leq \tau$ for some τ), this implies the evenly old property with constant $B = \tau$.

Assumption 3. Stochastic unbounded delays. The sequence of delay vectors $\vec{j}(0), \vec{j}(1), \ldots$ are IID, and independent of the block sequence $i(0), i(1), \ldots$. In addition, they are evenly old.

Hence there exists a function $p : \mathbb{N}^m \to [0,1]$ such that, for all $k \in \mathbb{N}$, the probability that $\vec{j}(k)$ equals some vector $\vec{v}$ is given by

$$\mathbb{P}\left[\vec{j}(k) = \vec{v}\right] = p(\vec{v}). \tag{13.1.1}$$

Define

$$P_l = \mathbb{P}[j(k) \geq l]. \tag{13.1.2}$$

Theorem 6. Convergence under stochastic unbounded delays. *Assume that the block sequence $i(k)$ is a uniform IID block sequence (Assumption 1) and that the delays vectors $\vec{j}(k)$ are an evenly old, IID sequence that is independent of the block sequence (Assumption 3). Let the step size be $\eta^k = ch$ for an arbitrary fixed*[1] *$c \in (0,1)$, and h given below. Then the iterates of ARock converge weakly to a solution with probability 1 if either of the following holds:*

1. $\sum_{l=1}^{\infty}(lP_l)^{1/2} < \infty$, *and setting* $h = \left(1 + \frac{1}{\sqrt{m}}\sum_{l=1}^{\infty} P_l^{1/2}\left(l^{1/2} + l^{-1/2}\right)\right)^{-1}$.
2. $\sum_{l=1}^{\infty} P_l^{1/2} l < \infty$, *and setting* $h = \left(1 + \frac{2}{\sqrt{m}}\sum_{l=1}^{\infty} P_l^{1/2}\right)^{-1}$.

Convergence under unbounded delays in this setting has only been proven under very strong assumptions (See Chapter 2 for a discussion of existing results). Additionally, this result improves on the step size criterion of ARock and other similar algorithms if we are willing to assume stochastic delays (e.g. (Z. Peng et al. 2016; J. Liu and Wright 2015; Ji Liu et al. 2015)). So for instance, there may be a scenario where the maximum delay τ is very high, but delays near that size rarely occur. Theorem 6 implies that asynchronous algorithms will convergence under a much larger step size than prior work.

Table 13.1 gives some example distributions, and corresponding values P_l and step size h (we only used the step size h in Theorem 6 since it is easier to calculate). We give an upper bound for P_l and lower bound for h to simplify expressions. Let $I[A](x)$ denote the characteristic function (i.e. a function that equal 1 for $x \in D$ and 0 otherwise).

In addition to example distributions, we consider the step size in the following scenario. Let Y be a random variable representing the time between when a node starts reading the solution vector x, and when its update is applied. Say that we have $a \leq Y \leq b$, and that there are p computing nodes. In the worst-case scenario, a node takes b seconds, and $p \cdot \left(\frac{b}{a} + 1\right)$ updates have occurred during this time. Hence ignoring the specifics of the distribution, we have a delay bound $\tau = p \cdot \left(\frac{b}{a} + 1\right)$. It can be seen from Table 3.1, that in this scenario, if b/a doesn't grow, then $\sqrt{m} \gg p$ implies a step size of $c \in (0,1)$ will result in convergence.

Remark 4. Step size heuristic. Even if the assumption of independent IID delays does

[1] *By "arbitrary fixed" we mean that the constant c can be any number in $(0,1)$, so long as that number does not change. However it is possible to relax this.*

Table 13.1: Example delay distributions and step sizes

Distribution of $j(k)$	**P_l Upper bound**	**h Step size lower bound**
$j(k)$ arbitrary, with $\leq \tau$	$1 \cdot I[0 \leq l \leq \tau](l)$	$\left(1 + \frac{2\tau}{\sqrt{m}}\right)^{-1}$
$j(k)$ uniform on $\{0, 1, 2, \ldots, \tau\}$	$\left(1 - \frac{l}{\tau+1}\right) I[0 \leq l \leq \tau]$	$\left(1 + \frac{4\tau}{3\sqrt{m}}\right)^{-1}$
$j(k)$ exponentially decaying. I.e. $\mathbb{P}[j(k) = l] \leq Cr^l$ for $1 > r > 0$.	$\propto \frac{r^{l/2}}{1-r}$	$\left(1 + 2\sqrt{\frac{C}{m}} \frac{r^{1/2}}{\left(1-r^{1/2}\right)^{3/2}}\right)^{-1}$
Each of p agents has update time $Y \in [a, b]$	*	$\left(1 + \frac{2p \cdot \left(\frac{b}{a}+1\right)}{\sqrt{m}}\right)^{-1}$

not hold in practice, the preceding step size gives a useful heuristic to use given an empirical distribution of delays measured in a system. For example, when the number of blocks m satisfies $\sqrt{m} \gg \sum_{l=1}^{\infty} P_l^{1/2}$, the step size sequence should be $\eta^k \approx c$, where $c \in (0, 1)$ is an arbitrary fixed constant.

13.2 Preliminaries

This section proves Theorem 7 below, which is a more general version of Theorem 6 from the introduction. Theorem 7 involves a sequence of arbitrary parameters $\epsilon_1, \epsilon_2, \ldots$ that appear naturally in our analysis. The values of these parameters can be chosen situationally to obtain different result. In Section 13.7, we select (i) the values that give the weakest conditions on delays, and (ii) the values that give the largest allowable step size to obtain the two parts of Theorem 6 from the introduction.

Definition 32. Summable sequence. Let $a = (a_1, a_2, \ldots)$ $(a_i \in \mathbb{R}, \forall i)$ be a sequence. a is said to be **summable** or "in ℓ^1" if its ℓ^1 norm is finite, that is,

$$\|a\|_{\ell^1} = \sum_{i=1}^{\infty} |a_i| < \infty.$$

Theorem 7. Convergence under stochastic delays. *Consider ARock under the following conditions:*

1. *The block sequence $i(k)$ is a uniform IID block sequence (Assumption 1).*

2. The sequence of delay vectors $\vec{j}(k)$ is an evenly old, IID sequence that is independent of the sequence $i(k)$ (Assumption 3).

3. Let $\epsilon_1, \epsilon_2, \ldots \in (0, \infty)$ be an arbitrary sequence of parameters such that $\sum_{i=1}^{\infty} \frac{1}{\epsilon_i} < \infty$ and $\sum_{l=1}^{\infty} \epsilon_l P_l l < \infty$ for $P_l = \mathbb{P}[j(k) \geq l]$ (Assumption 4).

4. The step size is chosen as $\eta^k = ch$ for an arbitrary fixed $c \in (0, 1)$ and $h = \left(1 + \frac{1}{m} \sum_{l=1}^{\infty} \epsilon_l P_l + \left\| \frac{1}{\epsilon_i} \right\|_{\ell^1}\right)^{-1}$.

Then with probability 1, the sequence of ARock iterates converges weakly to a solution.

This theorem is proven in Section 13.6.3 after we build up a series of results throughout this section. This section is written in a way that attempts to explain the logic and intuition behind the approach taken. A general strategy for constructing Lyapunov functions is presented in Section 4.5. In Section 13.8, we discuss how to modify the proof for the simpler case of bounded delay.

13.3 Proof outline

Both convergence proofs rely on the following convergence criterion for fixed-point algorithms (see (Bauschke and P. L. Combettes 2011)):

Proposition 33. Convergence of nonexpansive fixed-point iterations. *Let T be a nonexpansive operator with at least one fixed point. If we have the following:*

(1) Norm convergence:[2] $\left\|x^k - x^*\right\|$ converges for every $x^* \in \mathrm{Fix}(T)$, and

(2) Fixed-point-residual (FPR) strong convergence:[3] $\|Tx^k - x^k\| \to 0$,

then x^k weakly converges to some $x^* \in \mathrm{Fix}(T)$ [4].

Proposition 33 is the basis of our convergence proofs in this part, as well the proof

[2]We call this property *norm convergence*. The *distance* of x^k to each fixed-point x^* is what is converging (in general to a nonzero value) and not x^k itself. This property does not appear to have been given a name in the literature, although it is an important property in convergence proofs.

[3]The *fixed-point residual* (FPR) at x is defined as $(T - I)(x)$

[4]Weak convergence is the same as regular convergence in $\mathbb{R}^N$, but differs in a general Hilbert space.

of convergence of KM iteration. Toward applying Proposition 33, we need martingale convergence theory. This allows us to prove *norm convergence* and *FPR strong convergence* using results on the above Lyapunov function, which will complete the proof. Martingale theory is what allowed the authors in (Z. Peng et al. 2016) to prove that x^k converges to a solution for minimization of a convex function with Lipschitz gradient, and not just that the function value converged to the optimal value.

13.4 Preliminary results

Recall that stochastic unbounded delays are analyzed under Assumptions 1 and 3. Recall from Chapter 7 that we defined $X^k = \left\{x^0, \ldots, x^k\right\}$ and $J^k = \left\{\vec{j}(0), \ldots, \vec{j}(k)\right\}$. Again let $x^* = 0$, and $f(x^*) = 0$. If η^k is $\sigma\left(X^k, J^k\right)$-measurable, then again, we have:

$$\mathbb{E}\left[\left\|x^{k+1}\right\|^2 | \sigma\left(X^k, J^k\right)\right] = \left\|x^k\right\|^2 \underbrace{-2\frac{\eta^k}{m}\left\langle x^k, S\hat{x}^k\right\rangle}_{\text{cross term}} + \frac{\left(\eta^k\right)^2}{m}\left\|S\hat{x}^k\right\|^2. \tag{13.4.1}$$

13.4.1 A fundamental inequality

We start with a fundamental inequality, which is the starting point for analyzing convergence.

Proposition 34. Fundamental inequality. *Under Assumptions 1 and 3, for $j(k)$ the current delay, and an arbitrary sequence $\epsilon_1, \epsilon_2, \ldots \in (0, \infty)$, the ARock iterates obey the following inequality:*

$$\begin{aligned}\mathbb{E}\left[\left\|x^{k+1} - x^*\right\|^2 \middle| \sigma\left(X^k, J^k\right)\right] \leq \left\|x^k - x^*\right\|^2 + \frac{1}{m}\sum_{i=1}^{j(k)} \epsilon_i \left\|x^{k+1-i} - x^{k-i}\right\|^2 \\ - \frac{\eta^k}{m}\left\|S\hat{x}^k\right\|^2 \left(1 - \eta^k\left(1 + \sum_{i=1}^{j(k)} \frac{1}{\epsilon_i}\right)\right).\end{aligned} \tag{13.4.2}$$

The ϵ_i sequence is an arbitrary subject to $\epsilon_i > 0, \forall i$. In (Z. Peng et al. 2016), they are set to a constant value. However we eventually set them so that $1/\epsilon_i$ is summable, which is fundamental to the convergence proof for unbounded delays.

Proof. Let us start with the cross term in (13.4.1). Since T is nonexpansive, $\frac{1}{2}S$ is firmly

nonexpansive (FNE)[5]. Hence,

$$\begin{aligned}
-\frac{2\eta^k}{m}\left\langle S\hat{x}^k, x^k\right\rangle &= -\frac{2\eta^k}{m}\left(\left\langle S\hat{x}^k, \hat{x}^k\right\rangle + \left\langle S\hat{x}^k, x^k - \hat{x}^k\right\rangle\right) \\
&= -\frac{2\eta^k}{m}\left(2\left\langle \frac{1}{2}S\hat{x}^k, \hat{x}^k\right\rangle + \left\langle S\hat{x}^k, x^k - \hat{x}^k\right\rangle\right) \\
\left(\frac{1}{2}S \text{ is FNE}\right) &\leq -\frac{2\eta^k}{m}\left(2\left\|\frac{1}{2}S\hat{x}^k\right\|^2 + \left\langle S\hat{x}^k, x^k - \hat{x}^k\right\rangle\right) \\
&= -\frac{\eta^k}{m}\left\|S\hat{x}^k\right\|^2 - \frac{2\eta^k}{m}\left\langle S\hat{x}^k, x^k - \hat{x}^k\right\rangle \\
\text{(break into coordinate blocks)} &= \sum_{l=1}^{m}\left(-\frac{\eta^k}{m}\left\|S_l\hat{x}^k\right\|^2 - \frac{2\eta^k}{m}\left\langle S_l\hat{x}^k, x_l^k - x_l^{k-j(k,l)}\right\rangle\right).
\end{aligned}$$

Take block l. We turn the inner product into a telescoping sum:

$$\begin{aligned}
&-\frac{\eta^k}{m}\left\|S_l\hat{x}^k\right\|^2 - \frac{2\eta^k}{m}\left\langle S_l\hat{x}^k, x_l^k - x_l^{k-j(k,l)}\right\rangle \\
&= -\frac{\eta^k}{m}\left\|S_l\hat{x}^k\right\|^2 - \frac{2\eta^k}{m}\left(\sum_{i=1}^{j(k,l)}\left\langle S_l\hat{x}^k, x_l^{k+1-i} - x_l^{k-i}\right\rangle\right) \\
\text{(Cauchy-Schwarz)} &\leq -\frac{\eta^k}{m}\left\|S_l\hat{x}^k\right\|^2 + \frac{2\eta^k}{m}\left(\sum_{i=1}^{j(k,l)}\frac{1}{2}\left(\left\|S_l\hat{x}^k\right\|^2\frac{\eta^k}{\epsilon_i} + \frac{\epsilon_i}{\eta^k}\left\|x_l^{k+1-i} - x_l^{k-i}\right\|^2\right)\right) \\
&\leq -\frac{\eta^k}{m}\left\|S_l\hat{x}^k\right\|^2 + \frac{\eta^k}{m}\left(\sum_{i=1}^{j(k)}\left(\left\|S_l\hat{x}^k\right\|^2\frac{\eta^k}{\epsilon_i} + \frac{\epsilon_i}{\eta^k}\left\|x_l^{k+1-i} - x_l^{k-i}\right\|^2\right)\right) \\
&= \frac{\eta^k}{m}\left\|S_l\hat{x}^k\right\|^2\left(\eta^k\left(\sum_{i=1}^{j(k)}\frac{1}{\epsilon_i}\right) - 1\right) + \frac{1}{m}\sum_{i=1}^{j(k)}\epsilon_i\left\|x_l^{k+1-i} - x_l^{k-i}\right\|^2.
\end{aligned}$$

Adding all the components back together, we have:

$$-2\frac{\eta^k}{m}\left\langle x^k, S\hat{x}^k\right\rangle + \frac{\left(\eta^k\right)^2}{m}\left\|S\hat{x}^k\right\|^2 \leq \frac{\eta^k}{m}\left\|S\hat{x}^k\right\|^2\left(\eta^k\left(1 + \left(\sum_{i=1}^{j(k)}\frac{1}{\epsilon_i}\right)\right) - 1\right) + \frac{1}{m}\sum_{i=1}^{j(k)}\epsilon_i\left\|x^{k+1-i} - x^{k-i}\right\|^2.$$

Hence the proposition follows by adding $\left\|x^k\right\|^2$ to each side, and using (13.4.1). □

[5]A firmly nonexpansive (FNE) operator $Q : \mathbb{H} \to \mathbb{H}$ is an operator that can be written as $Q = \frac{1}{2}I + \frac{1}{2}R$, where R is nonexpansive. Equivalently, FNE operators satisfy $\langle Qy - Qx, y - x\rangle \geq \|Qy - Qx\|^2,\ \forall x, y \in \mathbb{H}$.

13.5 Constructing Lyapunov function

Similar to Part II, in this section we demonstrate how to construct a Lyapunov function from (13.4.2) to prove convergence:

$$\underbrace{\xi^k}_{\text{Total error}} = \underbrace{\left\|x^k - x^*\right\|^2}_{\text{Classical error}} + \underbrace{\frac{1}{m}\sum_{i=1}^{\infty} c_i \left\|x^{k+1-i} - x^{k-i}\right\|^2}_{\text{Asynchronicity error}}$$

13.5.1 Analysis of the Lyapunov function

We now analyze the conditional expectation of the Lyapunov function defined in (4.4.1). Recall the definition of $x^{k-\vec{j}}$ and $x^{k-\vec{j}(k)}$ from Chapter 4.

Lemma 35. Branch point lemma. *Take arbitrary $\epsilon_1, \epsilon_2, \ldots \in (0, \infty)$. Under Assumptions 1 and 3, the ARock iterates and ξ^k defined in (4.4.1) satisfy the following inequality:*

$$\begin{aligned}\mathbb{E}\left[\xi^{k+1}\middle|\sigma\left(X^k, J^k\right)\right] \leq \left\|x^k\right\|^2 + \frac{1}{m}\left(\sum_{i=1}^{j(k)} \epsilon_i \left\|x^{k+1-i} - x^{k-i}\right\|^2 + \sum_{i=1}^{\infty} c_{i+1}\left\|x^{k+1-i} - x^{k-i}\right\|^2\right) \\ - \frac{\eta^k}{m}\left\|Sx^{k-\vec{j}(k)}\right\|^2\left(1 - \eta^k\left(1 + \frac{c_1}{m} + \sum_{i=1}^{j(k)}\frac{1}{\epsilon_i}\right)\right).\end{aligned} \tag{13.5.1}$$

Proof. Calculate the expectation:

$$\mathbb{E}\left[\xi^{k+1}\middle|\sigma\left(X^k, J^k\right)\right] = \mathbb{E}\left[\left\|x^{k+1}\right\|^2\middle|\sigma\left(X^k, J^k\right)\right] + \frac{c_1}{m}\mathbb{E}\left[\left\|x^{k+1} - x^k\right\|^2\middle|\sigma\left(X^k, J^k\right)\right] \tag{13.5.2}$$

$$+ \frac{1}{m}\sum_{i=1}^{\infty} c_{i+1}\left\|x^{k+1-i} - x^{k-i}\right\|^2. \tag{13.5.3}$$

The second term yields (by the definition of ARock iteration (4.1.1), and taking expectation over $i(k)$)

$$\mathbb{E}\left[\left\|x^{k+1} - x^k\right\|^2\middle|\sigma\left(X^k, J^k\right)\right] = \frac{\left(\eta^k\right)^2}{m}\left\|S(x^{k-j(k)})\right\|^2. \tag{13.5.4}$$

Then,

$$\mathbb{E}\left[\xi^{k+1}\middle|\sigma\left(X^k, J^k\right)\right] = \underbrace{\mathbb{E}\left[\left\|x^{k+1}\right\|^2\middle|\sigma\left(X^k, J^k\right)\right]}_{A} + \underbrace{\frac{c_1}{m}\mathbb{E}\left[\left\|x^{k+1} - x^k\right\|^2\middle|\sigma\left(X^k, J^k\right)\right]}_{B}$$

$$
\begin{aligned}
&+\underbrace{\frac{1}{m}\sum_{i=1}^{\infty} c_{i+1}\left\|x^{k+1-i}-x^{k-i}\right\|^2}_{C}\\
&\leq \underbrace{\left\|x^k\right\|^2+\frac{\eta^k}{m}\left\|Sx^{k-\vec{j}(k)}\right\|^2\left(\eta^k\left(1+\sum_{i=1}^{j(k)}\frac{1}{\epsilon_i}\right)-1\right)+\frac{1}{m}\sum_{i=1}^{j}\epsilon_i\left\|x^{k+1-i}-x^{k-i}\right\|^2}_{A} \text{ (by (13.4}\\
&\quad+\underbrace{\frac{c_1}{m}\left(\frac{\left(\eta^k\right)^2}{m}\left\|Sx^{k-\vec{j}(k)}\right\|^2\right)}_{B} \text{ (by (13.5.4))}\\
&\quad\underbrace{+\frac{1}{m}\sum_{i=1}^{\infty} c_{i+1}\left\|x^{k+1-i}-x^{k-i}\right\|^2}_{C}\\
&=\left\|x^k\right\|^2+\frac{1}{m}\left(\sum_{i=1}^{j(k)}\epsilon_i\left\|x^{k+1-i}-x^{k-i}\right\|^2+\sum_{i=1}^{\infty} c_{i+1}\left\|x^{k+1-i}-x^{k-i}\right\|^2\right)\\
&\quad-\frac{\eta^k}{m}\left\|Sx^{k-\vec{j}(k)}\right\|^2\left(1-\eta^k\left(1+\frac{c_1}{m}+\sum_{i=1}^{j(k)}\frac{1}{\epsilon_i}\right)\right).
\end{aligned}
$$

□

In the proposition below, we derive the natural choice of parameters of the Lyapunov function that allow a meaningful comparison between $\mathbb{E}\left[\xi^{k+1}\middle|\ \sigma\left(X^k\right)\right]$ and ξ^k. With this choice, we obtain

$$
\mathbb{E}\left[\xi^{k+1}\middle|\sigma\left(X^k\right)\right]\leq \xi^k-(\text{descent terms}),
$$

which strongly resembles norm convergence: one of the convergence conditions in Proposition 33.

We first make some assumptions on the parameters. The necessity of these assumptions will become clear in the proof of Lemma 36.

Assumption 4. Coefficient summability conditions. Let $\epsilon_1,\epsilon_2,\ldots\in(0,\infty)$ and let $c_i=\sum_{l=i}^{\infty}\epsilon_l P_l$. These sequences also satisfy the summability conditions:

$$
\sum_{i=1}^{\infty}\frac{1}{\epsilon_i}<\infty, \tag{13.5.5}
$$

$$
\sum_{i=1}^{\infty} c_i<\infty. \tag{13.5.6}
$$

Lemma 36. Descent lemma for stochastic delays. *Consider the Lyapunov function* ξ^k *defined in* (4.4.1). *Let Assumptions 1, 3 and 4 hold. Let* $h = \left(1 + \frac{c_1}{m} + \left\|\frac{1}{\epsilon_i}\right\|_{\ell^1}\right)^{-1}$. *Then, ARock yields the following inequality for step size* η^k*:*

$$\mathbb{E}\left[\xi^{k+1}\middle|\sigma\left(X^k\right)\right] \leq \xi^k - \left(1 - \eta^k/h\right)\frac{\eta^k}{m}\sum_{\vec{j}\in\mathbb{N}^m} p\left(\vec{j}\right)\left\|Sx^{k-\vec{j}}\right\|^2.$$

Proof. From Lemma 35 and (13.5.5), we have:

$$\begin{aligned}\mathbb{E}\left[\xi^{k+1}\middle|\sigma\left(X^k, J^k\right)\right] \leq \left\|x^k\right\|^2 + \frac{1}{m}\left(\sum_{i=1}^{j(k)}\epsilon_i\left\|x^{k+1-i} - x^{k-i}\right\|^2 + \sum_{i=1}^{\infty} c_{i+1}\left\|x^{k+1-i} - x^{k-i}\right\|^2\right) \\ - \frac{\eta^k}{m}\left\|Sx^{k-\vec{j}(k)}\right\|^2\left(1 - \eta^k\underbrace{\left(1 + \frac{c_1}{m} + \left\|\frac{1}{\epsilon_i}\right\|_{\ell^1}\right)}_{1/h}\right).\end{aligned} \tag{13.5.7}$$

Let $p_j = \mathbb{P}[j(k) = j]$. Now take expectations over delays (via taking expectation with respect to $\sigma\left(X^k\right)$ instead of $\sigma\left(X^k, J^k\right)$).

$$\begin{aligned}\mathbb{E}\left[\xi^{k+1}\middle|\sigma\left(X^k\right)\right] &\leq \left\|x^k\right\|^2 + \frac{1}{m}\left(\sum_{j=1}^{\infty} p_j\sum_{i=1}^{j}\epsilon_i\left\|x^{k+1-i} - x^{k-i}\right\|^2 + \sum_{i=1}^{\infty} c_{i+1}\left\|x^{k+1-i} - x^{k-i}\right\|^2\right) \\ &\quad - \left(1 - \eta^k/h\right)\frac{\eta^k}{m}\sum_{\vec{j}\in\mathbb{N}^m} p\left(\vec{j}\right)\left\|Sx^{k-\vec{j}}\right\|^2 \\ &= \left\|x^k\right\|^2 + \frac{1}{m}\left(\sum_{i=1}^{\infty}\left(\sum_{j=i}^{\infty} p_j\right)\epsilon_i\left\|x^{k+1-i} - x^{k-i}\right\|^2 + \sum_{i=1}^{\infty} c_{i+1}\left\|x^{k+1-i} - x^{k-i}\right\|^2\right) \\ &\quad - \left(1 - \eta^k/h\right)\frac{\eta^k}{m}\sum_{\vec{j}\in\mathbb{N}^m} p\left(\vec{j}\right)\left\|Sx^{k-\vec{j}}\right\|^2 \\ &= \left\|x^k\right\|^2 + \frac{1}{m}\left(\sum_{i=1}^{\infty}(\epsilon_i P_i + c_{i+1})\left\|x^{k+1-i} - x^{k-i}\right\|^2\right) - \left(1 - \eta^k/h\right)\frac{\eta^k}{m}\sum_{\vec{j}\in\mathbb{N}^m} p\left(\vec{j}\right)\left\|Sx^{k-\vec{j}}\right\|^2.\end{aligned}$$

Let $\eta^k \leq h$ to eliminate the last term. Ideally $\mathbb{E}\left[\xi^{k+1}\middle|\sigma\left(X^k\right)\right] \leq \xi^k$, which can be achieved with:

$$\left\|x^k\right\|^2 + \frac{1}{m}\left(\sum_{i=1}^{\infty}(\epsilon_i P_i + c_{i+1})\left\|x^{k+1-i} - x^{k-i}\right\|^2\right) \leq \left\|x^k\right\|^2 + \frac{1}{m}\sum_{i=1}^{\infty} c_i\left\|x^{k+1-i} - x^{k-i}\right\|^2.$$

The obvious choice of coefficients is then given by $c_{i+1} + P_i\epsilon_i = c_i$. However this doesn't uniquely determine the coefficients. We assume that $c_i \to 0$ as i goes to ∞ to ensure that

any bounded sequence has a corresponding Lyapunov function that is finite. Hence:

$$c_i = \sum_{l=i}^{\infty} \epsilon_i P_i.$$

This recovers the coefficient formula from Assumption 4. With this choice of coefficients, we have our result. □

13.6 Convergence proof

Now that we have built a Lyapunov function and obtained Lemma 36, we can prove convergence.

Lemma 37. *Let Assumptions 1, 3 and 4 hold. Use step size $\eta^k = ch$ for some arbitrary fixed $c \in (0,1)$, and h given in Lemma 36. Then with probability 1, ξ^k converges, and in addition,*

$$\sum_{k=0}^{\infty} \sum_{\vec{j} \in \mathbb{N}^m} p\left(\vec{j}\right) \left\| S x^{k-\vec{j}} \right\|^2 < \infty. \tag{13.6.1}$$

The proof of this lemma relies on the following:

Theorem 8. Supermartingale convergence theorem (P. Combettes and Pesquet 2015). *Let α^k, θ^k and γ^k be positive sequences adapted to sigma algebra $\mathcal{H}^k$, and let γ^k be summable with probably 1. If*

$$\mathbb{E}\left[\alpha^{k+1} | \mathcal{H}^k\right] + \theta^k \leq \alpha^k + \gamma^k,$$

then with probability 1, α^k converges to a $[0,\infty)$-valued random variable, and $\sum_{k=1}^{\infty} \theta^k < \infty$.

We now prove Lemma 37.

Proof. Apply Theorem 8 with $\alpha^k = \xi^k$, $\gamma^k = 0$, and $\theta^k = \left(1 - \eta^k/h\right) \frac{\eta^k}{m} \sum_{\vec{j} \in \mathbb{N}^m} p\left(\vec{j}\right) \left\| S x^{k-\vec{j}} \right\|^2$. We immediately obtain our result by noting that $\left(1 - \eta^k/h\right) \frac{\eta^k}{m}$ is a constant. □

13.6.1 Norm convergence

Now is the point where the "evenly old" assumption about the delays made in Assumption 3 becomes important, and it is hard to see a way to weaken it. First a lemma on convolutions is necessary.

Lemma 38. Convolution lemma ((Arendt et al. 2011), Proposition 1.3.2). *Define the convolution of sequences* $a = (\ldots, a_{-2}, a_{-1}, a_0, a_1, a_2, \ldots)$ *and* $b = (\ldots, b_{-2}, b_{-1}, b_0, b_1, b_2, \ldots)$ *as the sequence defined by the formula*[6]*:*

$$(a * b)(k) = \sum_{i=-\infty}^{\infty} a_i b_{k-i}. \tag{13.6.2}$$

Let a_i *be in* ℓ^1*, and let* b *be bounded with* $b_i \to 0$ *as* $i \to \infty$*. Then the convolution* $(a*b)(k) \to 0$ *as* $k \to \infty$*.*

Proposition 39. Norm convergence. *Let Assumptions 1, 3 and 4 hold. Then with probability* 1, $\left\|x^k - x^*\right\|$ *converges for all* $x^* \in \mathrm{Fix}(T)$*.*

Proof. We first prove that with probability 1, $\frac{1}{m}\sum_{i=1}^{\infty} c_i \left\|x^{k+1-i} - x^{k-i}\right\|^2 \to 0$.

1. P_l **is summable.** Since the sequence $\frac{1}{\epsilon_i}$ is summable, $\epsilon_i \to \infty$, and thus $\inf_{i\in\mathbb{N}} \epsilon_i > 0$. Hence

$$\sum_{l=1}^{\infty} P_l \leq \frac{1}{\inf_{i\in\mathbb{N}} \epsilon_i} \sum_{l=1}^{\infty} \epsilon_l P_l = \frac{1}{\inf_{i\in\mathbb{N}} \epsilon_i} c_1 < \infty$$

2. $k - j(k) \to \infty$**.** (That is, the components of iterate x^k are used only a finite number of times).

$$\mathbb{P}[k - j(k) \leq k_0] = P_{k-k_0}$$
$$\sum_{k=k_0}^{\infty} \mathbb{P}[k - j(k) \leq k_0] = \sum_{k=k_0}^{\infty} P_{k-k_0} < \infty$$

Therefore by the Borel-Cantelli lemma, $k - j(k) \leq k_0$ happens only a finite number of times with probability 1. Hence with probability 1, this is true for all $k_0 \in \mathbb{N}$, which implies that $k - j(k) \to \infty$.

[6] *The convolution is not always well-defined, because the sum may not be convergent for all* k*. However in this lemma, it is well-defined.*

3. $Sx^{k+\vec{t}} \to 0$ for all delay feasible "patterns" $\vec{t}$. We assume without loss in generality that none of the delay vectors attained $(\vec{j}(0), \vec{j}(1), \ldots)$ has probability 0 (since this occurs with probability 1). Let $\vec{t}(k) \triangleq j(k)(1, \ldots, 1) - \vec{j}(k)$. $j(k)$ is the age of the oldest block in $x^{k-\vec{j}(k)}$, whereas $\vec{t}(k) \in \{0, 1, \ldots, B\}^m$ represent the "pattern" of the rest of the delay. We call a vector $\vec{t} \in \{0, 1, \ldots, B\}^m$ **feasible** if it occurs with nonzero probability. Take (13.6.1), and group the sum into feasible patterns and we obtain:

$$\begin{aligned} &\sum_{k=0}^{\infty} \left\| Sx^{k+\vec{t}} \right\|^2 < \infty, \\ \implies &\left\| Sx^{k+\vec{t}} \right\| \to 0, \end{aligned} \tag{13.6.3}$$

for each feasible $\vec{t}$.

4. Delayed fixed-point residual $\left\| Sx^{k-\vec{j}(k)} \right\| \to 0$. Observe that

$$\left\| Sx^{k-\vec{j}(k)} \right\| = \left\| Sx^{(k-j(k))+\vec{t}(k)} \right\|.$$

Let $A\left(k, \vec{t}\right) = \left\| Sx^{(k-j(k))+\vec{t}} \right\|$ (this is a family of sequences indexed by $\vec{t}$). By equation (13.6.3), and the fact that $k - j(k) \to \infty$, we have $A(k, \vec{t}) \to 0$ for any *fixed* $\vec{t}$. Notice that $\left\| Sx^{k-\vec{j}(k)} \right\| = A(k, \vec{t}(k))$. At every step, $A(k, \vec{t}(k))$ selects one from a finite family of sequences, all of which converge to 0. Since there are only a finite number of these sequences, $A(k, \vec{t}(k)) \to 0$ and hence $\left\| Sx^{k-\vec{j}(k)} \right\| \to 0$[7].

5. Difference sum converges to 0.

$$\begin{aligned} &\frac{1}{m} \sum_{i=1}^{\infty} c_i \left\| x^{k+1-i} - x^{k-i} \right\|^2 \\ \leq &\frac{c^2 h^2}{m} \sum_{i=1}^{\infty} c_i \left\| Sx^{(k-i)-\vec{j}(k-i)} \right\|^2 \\ = &\frac{c^2 h^2}{m} \left((0, \ldots, 0, c_1, c_2, \ldots) * \left(\ldots, \left\| Sx^{(i-1)-\vec{j}(i-1)} \right\|^2, \left\| Sx^{(i)-\vec{j}(i)} \right\|^2, \left\| Sx^{(i+1)-\vec{j}(i+1)} \right\|^2, \ldots \right) \right)(k) \end{aligned}$$

This expression is the convolution of an ℓ^1 sequence (Assumption 4), and a bounded sequence that converges to 0 as $i \to \infty$ (by part 4 of this proof) respectively. Therefore by Lemma 38, $\frac{1}{m} \sum_{i=1}^{\infty} c_i \left\| x^{k+1-i} - x^{k-i} \right\|^2 \to 0$.

[7] If you select from an *infinite* number of sequences converging to 0, this may not be true. E.g. consider $B(k, i) = \delta_{k-i}$, where $\delta_0 = 1$ and $\delta_l = 0$ for all $l \neq 0$. For fixed i, $B(k, i) \to 0$. However $B(k, k) = 1$ for all k, and hence never converges to 0.

6. Norm convergence. Because ξ^k converges a.s. and $\frac{1}{m}\sum_{i=1}^{\infty} c_i \left\|x^{k+1-i} - x^{k-i}\right\|^2 \to 0$ a.s., we have that for any particular x^*, $\left\|x^k - x^*\right\|$ converges with probability 1. Because the space is *separable*, this implies that with probability 1, $\left\|x^k - x^*\right\|$ converges for **all** $x^* \in \mathrm{Fix}(T)$, which is subtly different (See (P. Combettes and Pesquet 2015), Proposition 2.3 (iii) for a proof of this fact.). □

13.6.2 Fixed-point-residual strong convergence

Proposition 40. FPR strong convergence. *Under the conditions of Proposition 39,* $\left\|Sx^k\right\| \to 0$ *with probability 1.*

Proof. From equation (13.6.1), we have that $\left\|Sx^{k+\vec{t}}\right\| \to 0$ for some feasible $\vec{t}$ (clearly there must be at least one feasible $\vec{t}$). Recall that m is the number of blocks, and B is the maximum difference in age between blocks. We have

$$\begin{aligned}
\left\|Sx^k\right\| &\leq \left\|Sx^{k+\vec{t}} - Sx^k\right\| + \left\|Sx^{k+\vec{t}}\right\| \\
&\leq 2\left\|x^{k+\vec{t}} - x^k\right\| + \left\|Sx^{k+\vec{t}}\right\| \\
\text{(triangle inequality)} &\leq 2\sum_{i=1}^{m}\left\|x_i^{k+t_i} - x_i^k\right\| + \left\|Sx^{k+\vec{t}}\right\| \\
&\leq 2\sum_{i=1}^{m}\sum_{l=1}^{t_i}\left\|x_i^{k+l} - x_i^{k-1+l}\right\| + \left\|Sx^{k+\vec{t}}\right\| \\
\text{(since } \vec{t} \in \{0,1,\ldots,B\}^m\text{)} &\leq 2m\sum_{l=1}^{B}\left\|x_i^{k+l} - x_i^{k-1+l}\right\| + \left\|Sx^{k+\vec{t}}\right\| \to 0,
\end{aligned}$$

since $\left\|x^{k+1} - x^k\right\| \to 0$ and $\left\|Sx^{k+\vec{t}}\right\| \to 0$ (from parts 5 and 3 of the proof of Proposition 39 respectively). □

13.6.3 Proof of Theorem 7

Proof. Norm convergence is proven in Proposition 39. The FPR strong convergence criterion is proven in Proposition 40. Having satisfied the conditions of Proposition 33, we conclude that the sequence of ARock iterates converges to a solution with probability 1. Hence we have proven Theorem 7. □

13.7 Parameter choice

Choosing different parameters $\epsilon_1, \epsilon_2, \ldots$ lead to different convergence results. We featured two possibilities in Theorem 6 (though there are obviously others). We need both $\frac{1}{\epsilon_i} \in \ell^1$ and $\sum_{l=1}^{\infty} c_l = \sum_{l=1}^{\infty} \epsilon_l P_l l < \infty$ for convergence under step size $\eta^k = ch = c\Big(1 + \frac{1}{m}\sum_{l=1}^{\infty} \epsilon_l P_l + \Big\|\frac{1}{\epsilon_i}\Big\|_{\ell^1}\Big)^{-1}$.

1. If we wish to have the weakest restriction on our distribution of delays, let $\epsilon_l = m^{-1/2} P_l^{-1/2} l^{-1/2}$. This leads to the convergence condition $\sum_{l=1}^{\infty} P_l^{1/2} l^{1/2} < \infty$ for step size $\eta^k = c\Big(1 + \frac{1}{\sqrt{m}}\sum_{l=1}^{\infty} P_l^{1/2}\Big(l^{1/2} + l^{-1/2}\Big)\Big)^{-1}$.

2. If we wish to have the largest allowable step size (at the expense of a strong condition on the delay distribution), let $\epsilon_l = m^{-1/2} P_l^{-1/2}$. This leads to the convergence condition $\sum_{l=1}^{\infty} P_l^{1/2} l^1 < \infty$ for step size $\eta^k = c\Big(1 + \frac{2}{\sqrt{m}}\sum_{l=1}^{\infty} P_l^{1/2}\Big)^{-1}$.

13.8 Bounded delay

Our main focus is on unbounded delay, because convergence under unbounded delay is a new result. It is easy, though, to modify this section's proof for the case of bounded delay, which results in a much simpler proof. Let $\epsilon_1, \ldots, \epsilon_\tau, \in (0, \infty)$ be a series of parameters, let $c \in (0, 1)$, let the step size be $\eta^k = c\Big(1 + \sum_{l=1}^{\tau}\Big(\frac{1}{m}\epsilon_l P_l + \frac{1}{\epsilon_l}\Big)\Big)^{-1}$. Then we have convergence with probability 1. The proof uses the following Lyapunov function instead of an infinite sum version:

$$\xi^k = \left\|x^k - x^*\right\|^2 + \frac{1}{m}\sum_{i=1}^{\tau} c_i \left\|x^{k+1-i} - x^{k-i}\right\|^2, \qquad \text{for } c_i = \sum_{l=i}^{\tau} \epsilon_l P_l.$$

CHAPTER 14

Proof of Convergence for Unbounded Deterministic Delays

The second result of this part proves convergence of ARock and related algorithms under deterministic unbounded delays. Analysis of deterministic unbounded delay allows our result to apply more generally, but the result itself is weaker. In order to achieve convergence, it is necessary to use a step size η^k that is a decreasing function of the current delay $j(k)$ (whereas in Theorem 6, a constant step size was sufficient). Also convergence is only on a family of subsequences.

Assumption 5. Deterministic unbounded delays. The sequence of delay vectors $\vec{j}(0), \vec{j}(1), \vec{j}(2), \ldots$ is an arbitrary sequence in $\mathbb{N}^m$, independent of $i(k)$, with $j(k) < \infty$.

Definition 41. Convergence on subsequences of bounded delay. Let $x^0, x^1, x^2, \ldots$ be a sequence of iterates and $\vec{j}(0), \vec{j}(1), \vec{j}(2), \ldots$ a corresponding sequence of delay vectors, with $\liminf j(k) < \infty$. Let Q_J be the subsequence of $x^0, x^1, x^2, \ldots$ where the iterates x^k with current delay $j(k) > J$ are removed[1]. We say that x^k converges to x^* on subsequences of bounded delay if x^k converges to x^* on every subsequence Q_J for $J \geq \liminf j(k)$[2].

Theorem 9. Convergence under deterministic unbounded delays. *Assume that the block sequence $i(k)$ is a sequence of uniform IID random variables (Assumption 1) and that the sequence of delay vectors $\vec{j}(0), \vec{j}(1), \vec{j}(2), \ldots$ is an arbitrary sequence in $\mathbb{N}^m$, independent of $i(k)$, with $\liminf j(k) < \infty$ (Assumption 5). Pick arbitrary, fixed $c \in (0, 1)$ and $\gamma > 0$. Let*

[1] Q_J represent subsequences of bounded delay.

[2] $J \geq \liminf j(k)$ ensures that Q_J is an infinite subsequence.

the step size be

$$\eta^k = c\left(1 + \frac{1}{\sqrt{m}}\left(1 + \frac{1}{\gamma} + \frac{1}{2+\gamma}(j(k)+1)^{2+\gamma}\right)\right)^{-1}. \tag{14.0.1}$$

Then with probability 1*, the iterates of ARock weakly converge to a solution* x^* *on all subsequences of bounded delay* Q_J *for* $J \geq \liminf j(k)$ *(Definition 41), where* x^* *does not depend on the bound* J*.*

This step size rule assumes a worst case scenario. In practice it can be used if it was necessary to be certain that the algorithm converges. Even if network conditions are very unfavorable, making delays large, the algorithm with the step size (14.0.1) makes some progress at every step. This result could also be used in the bounded delay regime when the bound τ is not known in advance. In previous results, τ is needed in advance to calculate the correct step size.

Theorem 9 also provides a rule adaptive to the current delay. If the step size were set according to τ (which is the case for the vast majority of recent papers), the step size may be exceedingly pessimistic if a delay of τ is very rare. However our result implies a much larger allowable step size when delays are smaller (even if they may becomes large at some point in the future). When the delays are bounded (but the bound is possibly unknown to us), Theorem 9 implies weak convergence of the full sequence with probability 1, not merely on subsequences of bounded delay.

The step size rule also gives the following useful heuristic: When the number of blocks m satisfies $\sqrt{m} \gg (j(k)+1)^{2+\gamma}$, the step size should be $\eta^k \approx c \in (0,1)$.

Proving convergence for deterministic delays leads to a slightly weaker convergence result. This is likely because deterministic unbounded delay is a very general condition. Below is our most general result:

Theorem 10. Convergence under deterministic delays. *Consider ARock under the following conditions:*

1. *The block sequence* $i(k)$ *is a sequence of uniform IID random variables (Assumption 1).*

2. The sequence of delay vectors $\vec{j}(0), \vec{j}(1), \vec{j}(2), \ldots$ *is an arbitrary sequence in* $\mathbb{N}^m$*, independent of* $i(k)$*, with* $\liminf j(k) < \infty$ *(Assumption 5).*

3. Let $\epsilon_1, \epsilon_2, \ldots \in (0, \infty)$ *be an arbitrary sequence of parameters such that* $\sum_{l=1}^{\infty} \epsilon_l < \infty$*.*

4. The step size is set to $\eta^k = ch_{j(k)}$ *for some arbitrary fixed* $c \in (0, 1)$ *and* $h_j = \left(1 + \frac{1}{m}\|\epsilon_i\|_{\ell^1} + \sum_{i=1}^{j} \frac{1}{\epsilon_i}\right)^{-1}$*.*

Then with probability 1*, the sequence of ARock iterates converges weakly to a solution on subsequences of bounded delay (Definition 41).*

This theorem is proven in Section 14.2.3. Similar to Theorem 7, there is a sequence of parameters $\epsilon_1, \epsilon_2, \ldots$. However in the case of deterministic delays, there is no "best" way to chose ϵ_i's unless stronger assumptions are made on the delays. It is impossible to optimize the parameters to uniformly ensure the maximum allowable step size, since optimizing for a current delay of $j = n$ can only come at the expense of decreasing the allowable step size for other values $m \neq n$. We set these parameters to a convenient, simple choice in Section 14.3 to obtain Theorem 9 presented in the introduction.

Remark 5. Bounded delay. We can obtain a bounded-delay version of Theorem 10 by truncating the metric to the first τ terms as in Section 13.8 and setting $\epsilon_{\tau+1}, \epsilon_{\tau+2}, \ldots = 0$. Using the step size $\eta^k = c\left(1 + \sum_{i=1}^{j}\left(\frac{1}{m}\epsilon_l + \frac{1}{\epsilon_i}\right)\right)^{-1}$ results in convergence with probability 1.

14.1 Building a Lyapunov function

We build a Lyapunov function in a similar way to before. Our starting point is the Branch Point Lemma 35. Recall that $\sigma\left(X^k, J^k\right) = \sigma\left(x^0, x^1, \ldots, x^k, \vec{j}(0), \vec{j}(1), \ldots, \vec{j}(k)\right)$, and let the Lyapunov function ξ^k be defined as before in equation (4.4.1). First though, it is necessary to make an assumption on the coefficients of the Lyapunov function. The necessity of this assumption will become clear in the proof of Lemma 42.

Assumption 6. Coefficient formula. Let $\epsilon_1, \epsilon_2, \ldots \in (0, \infty)$ be an arbitrary sequence of parameters such that $\sum_{l=1}^{\infty} \epsilon_l < \infty$. The coefficients of the Lyapunov function in equation (4.4.1) are given by $c_i = \sum_{l=i}^{\infty} \epsilon_l$.

14.1.1 Analysis of the Lyapunov function

Lemma 42. Descent lemma for deterministic delays. *Consider the Lyapunov function* ξ^k *defined in* (4.4.1). *Let Assumptions 1, 5 and 6 hold. Define*

$$H_j = \left(1 + \frac{c_1}{m} + \sum_{i=1}^{j} \frac{1}{\epsilon_i}\right)^{-1}. \tag{14.1.1}$$

Then ARock yields the following inequality for step size η^k*:*

$$\mathbb{E}\left[\xi^{k+1} \middle| \sigma\left(X^k, J^k\right)\right] \leq \xi^k - \frac{\eta^k}{m}\left\|Sx^{k-\vec{j}(k)}\right\|^2\left(1 - \left(\eta^k / h_{j(k)}\right)\right). \tag{14.1.2}$$

Proof. Start from the Branch Point Lemma (35):

$$\begin{aligned}
\mathbb{E}\left[\xi^{k+1} \middle| \sigma\left(X^k, J^k\right)\right] &\leq \left\|x^k\right\|^2 + \frac{1}{m}\left(\sum_{i=1}^{j(k)} \epsilon_i \left\|x^{k+1-i} - x^{k-i}\right\|^2 + \sum_{i=1}^{\infty} c_{i+1}\left\|x^{k+1-i} - x^{k-i}\right\|^2\right) \\
&\quad - \frac{\eta^k}{m}\left\|Sx^{k-\vec{j}(k)}\right\|^2\left(1 - \eta^k\left(1 + \frac{c_1}{m} + \sum_{i=1}^{j(k)} \frac{1}{\epsilon_i}\right)\right) \\
&\leq \left\|x^k\right\|^2 + \frac{1}{m}\left(\sum_{i=1}^{\infty} (\epsilon_i + c_{i+1})\left\|x^{k+1-i} - x^{k-i}\right\|^2\right) \quad - \frac{\eta^k}{m}\left\|Sx^{k-\vec{j}(k)}\right\|^2\left(1 - \left(\eta^k / h_{j(k)}\right)\right).
\end{aligned}$$

First assume $\eta^k / h_{j(k)} \leq 1$, to eliminate the last term. Ideally we have $\mathbb{E}\left[\xi^{k+1} \middle| \sigma\left(X^k, J^k\right)\right] \leq \xi^k$, which can be achieved with:

$$\left\|x^k\right\|^2 + \frac{1}{m}\left(\sum_{i=1}^{\infty} (c_{i+1} + \epsilon_i)\left\|x^{k+1-i} - x^{k-i}\right\|^2\right) \leq \left\|x^k\right\|^2 + \frac{1}{m}\sum_{i=1}^{\infty} c_i \left\|x^{k+1-i} - x^{k-i}\right\|^2.$$

Using a similar argument to the one used in the proof of Lemma 36, we obtain the coefficient formula:

$$c_i = \sum_{l=i}^{\infty} \epsilon_l.$$

With this choice of coefficients, Lemma 42 is proven. □

14.2 Convergence proof

Now that we have built the Lyapunov function, and obtained Lemma 42, it is possible to prove convergence.

Lemma 43. *Consider the Lyapunov function ξ^k defined in (4.4.1). Let Assumptions 1, 5 and 6 hold. Define h_j via equation (14.1.1). Let the step size $\eta^k = ch_{j(k)}$ for an arbitrary fixed $c \in (0,1)$. Then with probability 1, ξ^k converges, and we have:*

$$\sum_{k=1}^{\infty} h_{j(k)} \left\| Sx^{k-\vec{j}(k)} \right\|^2 < \infty, \tag{14.2.1}$$

$$\sum_{k=1}^{\infty} \left\| x^{k+1} - x^k \right\|^2 < \infty. \tag{14.2.2}$$

Hence $h_{j(k)} \left\| Sx^{k-\vec{j}(k)} \right\|^2 \to 0$ and $\left\| x^{k+1} - x^k \right\| \to 0$.

Proof. Now $\left\| x^{k+1} - x^k \right\| \leq ch_{j(k)} \left\| Sx^{k-\vec{j}(k)} \right\|$ (see Definition 4), and $h_{j(k)} \leq 1$. Hence:

$$\sum_{k=1}^{\infty} \left\| x^{k+1} - x^k \right\|^2 \leq \sum_{k=1}^{\infty} c^2 h_{j(k)}^2 \left\| Sx^{k-\vec{j}(k)} \right\|^2 \leq \sum_{k=1}^{\infty} h_{j(k)} \left\| Sx^{k-\vec{j}(k)} \right\|^2.$$

Clearly then, equation (14.2.1) will imply all parts of this lemma (since any summable sequence converges to 0).

Use the Supermartingale Convergence Theorem (Theorem 8) on Lemma 42 with $\alpha^k = \xi^k$, $\gamma^k = 0$, and $\theta^k = \frac{\eta^k}{m} \left\| Sx^{k-\vec{j}(k)} \right\|^2 \left(1 - \left(\eta^k / h_{j(k)}\right)\right)$. This implies that ξ^k converges with probability 1, and we have:

$$\sum_{k=1}^{\infty} \frac{ch_{j(k)}}{m} \left\| Sx^{k-\vec{j}(k)} \right\|^2 (1-c) < \infty,$$

$$\implies \sum_{k=1}^{\infty} h_{j(k)} \left\| Sx^{k-\vec{j}(k)} \right\|^2 < \infty.$$

This proves the lemma. □

14.2.1 Norm convergence

Lemma 44. *Assume the conditions of Lemma 43. Then with probability 1, $\left\| x^k - x^* \right\|$ converges for all $x^* \in \mathrm{Fix}(T)$.*

Proof. **1) Difference sum converges to 0:**

$$\frac{1}{m} \sum_{i=1}^{\infty} c_i \left\| x^{k+1-i} - x^{k-i} \right\|^2$$

$$= \left((0,\dots,0,c_1,c_2,\dots) * \left(\dots,\left\|x^{(i-1)+1}-x^{i-1}\right\|^2,\left\|x^{i+1}-x^{i}\right\|^2,\left\|x^{(i+1)+1}-x^{(i+1)}\right\|^2,\dots\right)\right)(k)$$

Hence the difference sum is the convolution of a bounded sequence that converges to 0 as $i \to \infty$ (by Assumption 6), and an ℓ^1 sequence (by Lemma 43), respectively. Notice the reversal of roles from Proposition 39. Therefore, by Lemma 38, the difference sum converges to 0 with probability 1.

2) Norm Convergence: Therefore for any particular $x^* \in \mathrm{Fix}(T)$, with probability 1, $\left\|x^k - x^*\right\|$ converges. As argued before in the proof of Proposition 39, because the space is *separable*, this implies that with probability 1, $\left\|x^k - x^*\right\|$ converges for **all** $x^* \in \mathrm{Fix}(T)$. □

14.2.2 Fixed-point-residual strong convergence on subsequences of bounded delay

Lemma 45. FPR strong convergence. *Let the conditions of Lemma 43 hold. Let $J \geq \liminf j(k)$. Let $Q_J \subset \mathbb{N}$ be the subsequence of indices, k, on which the current delay, $j(k)$, is less than or equal to J (see Definition 41). On this subsequence, we have $\left\|Sx^k\right\| \to 0$.*

Proof. **1) Delayed fixed-point residual $\left\|Sx^{k-\vec{j}(k)}\right\| \to 0$ on Q_J.** The starting point is (14.2.1) from Lemma 43:

$$\sum_{k=1}^{\infty} h_{j(k)}\left\|Sx^{k-\vec{j}(k)}\right\|^2 < \infty,$$

Consider the subsequence $Q_J \subset \mathbb{N}$. On this subsequence, the above becomes:

$$\infty > \sum_{k\in Q_J} h_{j(k)}\left\|Sx^{k-\vec{j}(k)}\right\|^2 \geq \sum_{k\in Q_J} h_T\left\|Sx^{k-\vec{j}(k)}\right\|^2 \qquad \text{(since } h_j \text{ is decreasing in } j\text{)}.$$

Hence $\infty > \sum_{k\in Q_J}\left\|Sx^{k-\vec{j}(k)}\right\|^2$. So $\left\|Sx^{k-\vec{j}(k)}\right\| \to 0$ on Q_J.

2) Fixed-point residual strong convergence.

$$\begin{aligned}\left\|Sx^k\right\| &\leq \left\|Sx^k - Sx^{k-\vec{j}(k)}\right\| + \left\|Sx^{k-\vec{j}(k)}\right\| \\ &\leq 2\left\|x^k - x^{k-\vec{j}(k)}\right\| + \left\|Sx^{k-\vec{j}(k)}\right\| \\ &\leq 2\sum_{l=1}^{m}\left\|x_l^k - x_l^{k-j(k,l)}\right\| + \left\|Sx^{k-\vec{j}(k)}\right\|\end{aligned}$$

$$\leq 2\sum_{l=1}^{m}\sum_{i=1}^{j(k,l)}\left\|x_l^{k+1-i}-x_l^{k-i}\right\|+\left\|Sx^{k-\vec{j}(k)}\right\|$$
$$\leq 2m\Big(\left\|x^k-x^{k-1}\right\|+\ldots+\left\|x^{k-(T+1)}-x^{k-T}\right\|\Big)+\left\|Sx^{k-j(k)}\right\|\to 0.$$

The last line converges to 0 because $\left\|x^k-x^{k-1}\right\|\to 0$ and $\left\|Sx^{k-j(k)}\right\|\to 0$. Hence $\left\|Sx^k\right\|\to 0$ on Q_J. □

14.2.3 Proof of Theorem 10

Proof. Norm convergence was proven in Lemma 44. FPR strong convergence on subsequences of bounded delay was proven in Lemma 45. Having satisfied the conditions of Proposition 33, we conclude that the sequence of ARock iterates converges to a solution with probability 1 on subsequence of bounded delay. □

14.3 Parameter choice

The parameters $\epsilon_1, \epsilon_2, \ldots$ are arbitrary. However, for the purposes of simplicity and demonstration, ϵ_l was set to $l^{1+\gamma}\sqrt{m}$ for $\gamma > 0$ to obtain Theorem 9 in the introduction, from the more general Theorem 10. Integration is used to simplify the summations involved in obtaining the step size formula.

CHAPTER 15

Bibliography

Bibliography

Agarwal, Alekh and John C Duchi. "Distributed Delayed Stochastic Optimization." In: *Advances in Neural Information Processing Systems 24*. 2011, pp. 873–881.

Allen-Zhu, Zeyuan. "Katyusha: The First Direct Acceleration of Stochastic Gradient Methods." In: *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*. STOC 2017. New York, NY, USA: ACM, 2017, pp. 1200–1205.

Allen-Zhu, Zeyuan and Elad Hazan. "Optimal Black-Box Reductions Between Optimization Objectives." In: *Proceedings of the 30th International Conference on Neural Information Processing Systems*. NIPS'16. USA: Curran Associates Inc., 2016, pp. 1614–1622.

Allen-Zhu, Zeyuan, Zheng Qu, et al. "Even Faster Accelerated Coordinate Descent Using Non-Uniform Sampling." In: *International Conference on Machine Learning*. International Conference on Machine Learning. June 11, 2016, pp. 1110–1119.

Arendt, Wolfgang et al. *Vector-Valued Laplace Transforms and Cauchy Problems: Second Edition*. Springer Science & Business Media, Apr. 5, 2011. 540 pp.

Arjevani, Yossi. "Limitations on Variance-Reduction and Acceleration Schemes for Finite Sums Optimization." In: *Advances in Neural Information Processing Systems 30*. Curran Associates, Inc., 2017, pp. 3540–3549.

Avron, H., A. Druinsky, and A. Gupta. "Revisiting Asynchronous Linear Solvers: Provable Convergence Rate through Randomization." In: *Parallel and Distributed Processing Symposium, 2014 IEEE 28th International*. Parallel and Distributed Processing Symposium, 2014 IEEE 28th International. May 2014, pp. 198–207.

Bauschke, Heinz H. and Patrick L. Combettes. *Convex Analysis and Monotone Operator Theory in Hilbert Spaces*. Springer Science & Business Media, Apr. 19, 2011. 470 pp.

Bertsekas, Dimitri P. "Distributed Asynchronous Computation of Fixed Points." In: *Mathematical Programming* 27.1 (1983), pp. 107–120.

Bertsekas, Dimitri P. and John N. Tsitsiklis. *Parallel and Distributed Computation: Numerical Methods*. Athena Scientific, 1997.

Cannelli, Loris et al. "Asynchronous Parallel Algorithms for Nonconvex Big-Data Optimization. Part II: Complexity and Numerical Results." In: (Jan. 17, 2017). arXiv: 1701.04900.

Chang, Chih-Chung and Chih-Jen Lin. "LIBSVM: A Library for Support Vector Machines." In: *ACM Trans. Intell. Syst. Technol.* 2.3 (May 2011), 27:1–27:27.

Chazan, D. and W. Miranker. "Chaotic Relaxation." In: *Linear Algebra and its Applications* 2.2 (Apr. 1, 1969), pp. 199–222.

Chow, Y., T. Wu, and W. Yin. "Cyclic Coordinate-Update Algorithms for Fixed-Point Problems: Analysis and Applications." In: *SIAM Journal on Scientific Computing* 39.4 (Jan. 1, 2017), A1280–A1300.

Combettes, P. and J. Pesquet. "Stochastic Quasi-Fejér Block-Coordinate Fixed Point Iterations with Random Sweeping." In: *SIAM Journal on Optimization* 25.2 (Jan. 1, 2015), pp. 1221–1248.

Davis, Damek. "SMART: The Stochastic Monotone Aggregated Root-Finding Algorithm." In: (Jan. 4, 2016). arXiv: 1601.00698.

Davis, Damek and Wotao Yin. "A Three-Operator Splitting Scheme and Its Optimization Applications." In: *Set-Valued and Variational Analysis* 25.4 (Dec. 1, 2017), pp. 829–858.

Duchi, John C., Sorathan Chaturapruek, and Christopher Ré. "Asynchronous Stochastic Convex Optimization." In: (Aug. 4, 2015). arXiv: 1508.00882.

Dutta, Sanghamitra et al. "Slow and Stale Gradients Can Win the Race: Error-Runtime Trade-Offs in Distributed SGD." In: (Mar. 3, 2018). arXiv: 1803.01113.

Eisenberg, Bennett. "On the Expectation of the Maximum of IID Geometric Random Variables." In: *Statistics & Probability Letters* 78.2 (Feb. 1, 2008), pp. 135–143.

Fang, Cong, Yameng Huang, and Zhouchen Lin. "Accelerating Asynchronous Algorithms for Convex Optimization by Momentum Compensation." In: (Feb. 27, 2018). arXiv: 1802.09747.

Hannah, Robert, Fei Feng, and Wotao Yin. "A2BCD: Asynchronous Acceleration with Optimal Complexity." In: International Conference on Learning Representations. Sept. 27, 2018.

Hannah, Robert, Yanli Liu, et al. "Breaking the Span Assumption Yields Fast Finite-Sum Minimization." In: *Advances in Neural Information Processing Systems 31*. Curran Associates, Inc., 2018, pp. 2312–2321.

Hannah, Robert and Wotao Yin. "More Iterations per Second, Same Quality – Why Asynchronous Algorithms May Drastically Outperform Traditional Ones." In: (Aug. 17, 2017). arXiv: 1708.05136.

— "On Unbounded Delays in Asynchronous Parallel Fixed-Point Algorithms." In: *Journal of Scientific Computing* (Dec. 12, 2017), pp. 1–28.

Johnstone, Patrick R. and Jonathan Eckstein. "Projective Splitting with Forward Steps: Asynchronous and Block-Iterative Operator Splitting." In: (Mar. 19, 2018). arXiv: 1803.07043.

Kella, Offer and Wolfgang Stadje. "Superposition of Renewal Processes and an Application to Multi-Server Queues." In: *Statistics & Probability Letters* 76.17 (Nov. 2006), pp. 1914–1924.

Lan, Guanghui and Yi Zhou. "An Optimal Randomized Incremental Gradient Method." In: *Mathematical Programming* (June 28, 2017), pp. 1–49.

Leblond, Rémi, Fabian Pedregosa, and Simon Lacoste-Julien. "ASAGA: Asynchronous Parallel SAGA." In: *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*. Apr. 10, 2017, pp. 46–54.

Lee, Y. T. and A. Sidford. "Efficient Accelerated Coordinate Descent Methods and Faster Algorithms for Solving Linear Systems." In: *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*. 2013 IEEE 54th Annual Symposium on Foundations of Computer Science. Oct. 2013, pp. 147–156.

Lian, Xiangru, Huan Zhang, et al. "A Comprehensive Linear Speedup Analysis for Asynchronous Stochastic Parallel Optimization from Zeroth-Order to First-Order." In: *Advances in Neural Information Processing Systems 29*. Curran Associates, Inc., 2016, pp. 3054–3062.

Lian, Xiangru, Wei Zhang, et al. "Asynchronous Decentralized Parallel Stochastic Gradient Descent." In: *International Conference on Machine Learning*. International Conference on Machine Learning. July 3, 2018, pp. 3043–3052.

Lin, Qihang, Zhaosong Lu, and Lin Xiao. "An Accelerated Proximal Coordinate Gradient Method." In: *Advances in Neural Information Processing Systems 27*. Curran Associates, Inc., 2014, pp. 3059–3067.

Liu, J. and S. Wright. "Asynchronous Stochastic Coordinate Descent: Parallelism and Convergence Properties." In: *SIAM Journal on Optimization* 25.1 (Jan. 1, 2015), pp. 351–376.

Liu, Ji et al. "An Asynchronous Parallel Stochastic Coordinate Descent Algorithm." In: *J. Mach. Learn. Res.* 16.1 (Jan. 2015), pp. 285–322.

Liu, Tianyi et al. "Towards Understanding Acceleration Tradeoff between Momentum and Asynchrony in Nonconvex Stochastic Optimization." In: *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., 2018.

Luo, Z. Q. and P. Tseng. "On the Convergence of the Coordinate Descent Method for Convex Differentiable Minimization." In: *Journal of Optimization Theory and Applications* 72.1 (Jan. 1992), pp. 7–35.

Luo, Zhi-Quan and Paul Tseng. "On the Convergence Rate of Dual Ascent Methods for Linearly Constrained Convex Minimization." In: *Mathematics of Operations Research* 18.4 (Nov. 1, 1993), pp. 846–867.

Mania, H. et al. "Perturbed Iterate Analysis for Asynchronous Stochastic Optimization." In: *SIAM Journal on Optimization* 27.4 (Jan. 1, 2017), pp. 2202–2229.

Meng, Qi et al. "Asynchronous Accelerated Stochastic Gradient Descent." In: *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence* (New York, New York, USA). IJCAI'16. AAAI Press, 2016, pp. 1853–1859.

Mitov, Kosto V. and Edward Omey. "Renewal Processes." In: *Renewal Processes*. SpringerBriefs in Statistics. Springer International Publishing, 2014, pp. 1–51.

Nesterov, Y. "Efficiency of Coordinate Descent Methods on Huge-Scale Optimization Problems." In: *SIAM Journal on Optimization* 22.2 (Jan. 1, 2012), pp. 341–362.

Nesterov, Yurii. "A Method of Solving a Convex Programming Problem with Convergence Rate O (1/K2)." In: *Soviet Mathematics Doklady*. Vol. 27. 1983, pp. 372–376.

Nesterov, Yurii. *Introductory Lectures on Convex Optimization: A Basic Course*. Springer Science & Business Media, Dec. 1, 2013. 253 pp.

Peng, Z. et al. "ARock: An Algorithmic Framework for Asynchronous Parallel Coordinate Updates." In: *SIAM Journal on Scientific Computing* 38.5 (Jan. 1, 2016), A2851–A2879.

Peng, Zhimin, Tianyu Wu, et al. "Coordinate Friendly Structures, Algorithms and Applications." In: *Annals of Mathematical Sciences and Applications* 1.1 (2016), pp. 57–119. arXiv: 1601.00863.

Peng, Zhimin, Yangyang Xu, et al. "On the Convergence of Asynchronous Parallel Iteration with Unbounded Delays." In: *Journal of the Operations Research Society of China* (Dec. 9, 2017), pp. 1–38.

Recht, Benjamin et al. "Hogwild!: A Lock-Free Approach to Parallelizing Stochastic Gradient Descent." In: *Advances in Neural Information Processing Systems 24*. 2011, pp. 693–701.

Roux, Nicolas L., Mark Schmidt, and Francis R. Bach. "A Stochastic Gradient Method with an Exponential Convergence _Rate for Finite Training Sets." In: *Advances in Neural Information Processing Systems 25*. Curran Associates, Inc., 2012, pp. 2663–2671.

Rumble, Stephen M. et al. "It's Time for Low Latency." In: *Proceedings of the 13th USENIX Workshop on Hot Topics in Operating Systems*. 2011, pp. 11–15.

Ryu, Ernest K. and Stephen Boyd. "Primer on Monotone Operator Methods." In: *Preprint, available at http://web. stanford.edu/∼eryu/papers/monotone_notes. pdf* (2015).

Ryu, Ernest K., Robert Hannah, and Wotao Yin. "Scaled Relative Graph: Nonexpansive Operators via 2D Euclidean Geometry." In: (Feb. 26, 2019). arXiv: 1902.09788 [math].

Serpedin, Erchin and Qasim M. Chaudhari. *Synchronization in Wireless Sensor Networks: Parameter Estimation, Peformance Benchmarks, and Protocols*. 1st. New York, NY, USA: Cambridge University Press, 2009.

Solomonik, Edgar and James Demmel. "Communication-Optimal Parallel 2.5D Matrix Multiplication and LU Factorization Algorithms." In: *Euro-Par 2011 Parallel Processing*. Lecture Notes in Computer Science 6853. Springer Berlin Heidelberg, Aug. 29, 2011, pp. 90–109.

Sra, Suvrit et al. "AdaDelay: Delay Adaptive Distributed Stochastic Optimization." In: *Artificial Intelligence and Statistics*. Artificial Intelligence and Statistics. May 2, 2016, pp. 957–965.

Su, Weijie, Stephen Boyd, and Emmanuel Candes. "A Differential Equation for Modeling Nesterov's Accelerated Gradient Method: Theory and Insights." In: *Advances in Neural Information Processing Systems 27*. 2014, pp. 2510–2518.

Sun, Ruoyu and Yinyu Ye. "Worst-Case Complexity of Cyclic Coordinate Descent: N2 Gap with Randomized Version." In: (Apr. 25, 2016). arXiv: 1604.07130.

Sun, Tao, Robert Hannah, and Wotao Yin. "Asynchronous Coordinate Descent under More Realistic Assumptions." In: *Advances in Neural Information Processing Systems 30*. 2017, pp. 6183–6191.

Sutter, Herb. "The Free Lunch Is over: A Fundamental Turn toward Concurrency in Software." In: *Dr. Dobb's journal* 30.3 (2005), pp. 202–210.

— *Welcome to the Jungle*. Nov. 8, 2011. URL: https://herbsutter.com/welcome-to-the-jungle/.

Taylor, Adrien B., Julien M. Hendrickx, and François Glineur. "Exact Worst-Case Convergence Rates of the Proximal Gradient Method for Composite Convex Minimization." In: *Journal of Optimization Theory and Applications* 178.2 (Aug. 1, 2018), pp. 455–476.

Tseng, P. "On the Rate of Convergence of a Partially Asynchronous Gradient Projection Algorithm." In: *SIAM Journal on Optimization* 1.4 (Nov. 1, 1991), pp. 603–619.

Tseng, P., D. Bertsekas, and J. Tsitsiklis. "Partially Asynchronous, Parallel Algorithms for Network Flow and Other Problems." In: *SIAM Journal on Control and Optimization* 28.3 (Mar. 1, 1990), pp. 678–710.

Zhou, Zhengyuan et al. "Distributed Asynchronous Optimization with Unbounded Delays: How Slow Can You Go?" In: *International Conference on Machine Learning*. International Conference on Machine Learning. July 3, 2018, pp. 5970–5979.