UNIVERSITY OF CALIFORNIA

Los Angeles

Kaczmarz Methods and Structured Matrix Decompositions

A dissertation submitted in partial satisfaction

of the requirements for the degree

Doctor of Philosophy in Mathematics

by

Yotam Yaniv

2024

ABSTRACT OF THE DISSERTATION

Kaczmarz Methods and Structured Matrix Decompositions

by

Yotam Yaniv
Doctor of Philosophy in Mathematics
University of California, Los Angeles, 2024
Professor Andrea Bertozzi, Co-Chair
Professor Deanna M. Hunter, Co-Chair

In this dissertation, we discuss two distinct topics, both of which leverage randomized algorithms in numerical linear algebra. First we study three variants of the Kaczmarz method, a stochastic iterative method for solving linear systems. We propose a variant of the Kaczmarz method that uses additional memory to save on computation. We provide theoretical analysis and experimental results of the method, highlighting a gap in the literature. Additionally, we propose a variant of the Kaczmarz method in the data streaming setting that has an additional heavy ball momentum term. We prove a convergence bound for this method and analyze its merits experimentally given coherent data. Furthermore, we develop a variant of the Kaczmarz method for solving a latent class regression problem. Next we shift gears and discuss structured matrix factorizations. The first matrix factorization that we propose is a stratified non-negative matrix factorization. The aim of this method is to provide unsupervised dimensionality reduction on non-negative data that may be distributed across different locations. We prove a convergence bound for this method and analyze its performance on synthetic text, image and tabular data. Finally, we propose a hierarchically semi-separable matrix factorization method that uses random matrix sketching.

The dissertation of Yotam Yaniv is approved.

Stanley J. Osher

Christopher R. Anderson

Deanna M. Hunter, Committee Co-Chair

Andrea Bertozzi, Committee Co-Chair

University of California, Los Angeles

2024

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

I would first like to thank my Advisors Andrea Bertozzi and Deanna Needell for their guidance throughout my PhD. I would also like to thank my mentor at Lawrence Berkeley National Lab Sherry Li.

My academic journey started many years ago with the support of my family, friends and collaborators. I would like to thank my mother and father for taking me to America and supporting me throughout my life both financially and emotionally. I would like to thank my siblings for being amazing brothers, it is weird to think about how we are all almost adults now. I would like to thank my partner Allison for supporting me even when I was stressed or frustrated.

I would also like to thank my friends from my primary and secondary education: Jacob A, Aaron, Stephen, Jacob H, Gil, Josh, Cormac, George, Matt, and Mark for our many years of fun. I would like to thank my favorite high school teachers Mr. Hanlon and Mr. Murray for inspiring me to think critically about programming, world history, world religions and my life.

I would also like to thank my friends from UMD: Nitin, Victor, Charlie, Josephine, Solomon, Kweku and Jon who taught me many life lessons and helped me gain a new perspective on a variety of topics ranging from algorithms to cheering for the Washington Capitals. Additionally, I appreciate the guidance from my undergrad mentors Tom Goldstein, and Antoine Mellet without your recommendations and advice I would not have made it here.

I am thankful to the graduate students in the department many of whom have become lifelong friends of mine. I would like to thank Adam Lott, Kevin Miller, Jacob Moorman, and Thomas Tu for their advice and mentorship. I would like to thank my office-mate Dominic Yang for his advice and our great conversations. I would like to thank Erin George and Wes Wise for being great roommates. I would like to thank my basic exam study group

consisting of Jason Brown, Grace Li, Cecilia Higgins, Andrew Sack, Jerry Luo and our wonderful teacher who went above and beyond for us Bon-Soon Lin. I would like to thank my friends James Chapman and Michael Johnson for all of our long runs, fun adventures, and beach days. I would not have been able to thrive at UCLA without the support of all of you. I am grateful that I had the opportunity to experience this with all of you.

Chapter 2 is a version of [YMS22] which is joint work with Jacob Moorman, William Swartworth, Thomas Tu, Daji Landis and Deanna Needell. Jacob Moorman and I proposed the initial idea and contributed the convergence analysis. William Swartworth and I contributed corollaries 11-15 about the Gramian selectable set. Daji Landis, Thomas Tu and I contributed the experimental results. All of which was done under the supervision of Deanna Needell.

Chapter 3 is a version of [JYN22] which is joint work with Ben Jarman and Deanna Needell. Ben Jarman proposed the initial project. Ben Jarman and I contributed the convergence analysis and experiments under the supervision of Deanna Needell.

Chapter 4 is a version of [GYN22] which is joint work with Erin George and Deanna Needell. Deanna Needell proposed the initial idea. I implemented the method and contributed the experiments and Erin contributed the convergence analysis. Together we wrote the paper as co-first authors under the supervision of Deanna Needell.

Chapter 5 is a version of [CYN23] which is joint work with James Chapman and Deanna Needell. James and I proposed the initial idea, contributed the experiments and convergence analysis under the supervision of Deanna Needell.

Chapter 6 is an adaptation of the background and appendices of a manuscript, [YMG23], which is joint work with Osman Asif Malik, Pieter Ghysels, and Xiaoye S. Li. Pieter Ghysels and Xiaoye S. Li proposed the project. Xiaoye S. Li, Pieter Ghysels and I wrote the background of our matrix compression method. Osman Asif Malik and I compiled the theoretical results for this work.

I was supported in part by the UCLA Division of Graduate Education (dissertation year

fellowship). The work in Chapter 2 was supported by NSF DMS 1737770, 2011140, 2108479, 2027277, NSF DGE 1829071. The work in Chapters 3 and 4 was supported by NSF DMS-2108479 and NSF DMS-2011140. The work in Chapter 6 was conducted thanks to the NSF MSGI summer internship program and the Exascale Computing Project (17-SC-20-SC). This research used resources of the National Energy Research Scientific Computing Center (NERSC), a U.S. Department of Energy Office of Science User Facility located at Lawrence Berkeley National Laboratory, operated under Contract No. DE-AC02-05CH11231 using NERSC award ASCR-ERCAP0017690.

VITA

2017	Undergraduate Research Intern, UCLA Applied Mathematics REU.
2018	Undergraduate Research Intern, UCLA Applied Mathematics REU.
2019	B.S. (Mathematics with high honors) and B.S. (Computer Science with honors) UMD, College Park.
2019–2024	Fellow and Research Assistant, Mathematics Department UCLA.
2020-2024	Teaching Assistant, Mathematics Department UCLA.
2021	M.A (Mathematics) UCLA.
2022	NSF MSGI Intern / Research Assistant, Lawrence Berkeley National Lab, Berkeley, California.
2023	NSF MSGI Intern, Lawrence Berkeley National Lab, Berkeley, California.

PUBLICATIONS

Y. Yaniv, O. A. Malik, P. Ghysels, X. S. Li. "Construction of Hierarchically Semi-Separable matrix Representation using Adaptive Johnson-Lindenstrauss Sketching." *arXiv preprint arXiv:2302.01977*, 2023.

J. Chapman, Y. Yaniv, D. Needell. "Stratified-NMF for Heterogeneous Data." *57th Asilomar Conference on Signals, Systems and Computers,* 2023.

E. George, Y. Yaniv, D. Needell. "Multi-Randomized Kaczmarz for Latent Class Regression." *56th Asilomar Conference on Signals, Systems and Computers,* 2022.

B. Jarman, Y. Yaniv, D. Needell. "Online Signal Recovery via Heavy Ball Kaczmarz." *56th Asilomar Conference on Signals, Systems and Computers,* 2022.

Y. Yaniv, J. D. Moorman, W. Swartworth, T. Tu, D. Landis, and D. Needell. "Selectable Set Randomized Kaczmarz." *Numerical Linear Algebra with Applications* (2022): e2458.

D. J. Arnold, D. Fernandez, R. Jia, C. Parkinson, D. Tonne, Y. Yaniv, A. L. Bertozzi, and S. J. Osher. "Modeling environmental crime in protected areas using the level set method." *SIAM Journal on Applied Mathematics* 79, no. 3 (2019): 802-821.

CHAPTER 1

Introduction

With the development of large storage devices and abundant data collection methods, from smartphones to medical devices, there has been a paradigm shift from the limited data regime to the big data regime. Indeed, the emphasis has transitioned from the extraction of maximum insight given limited data to extracting pertinent information from massive amounts of data. The aim of this dissertation is to address some of the problems in the big data regime by developing scalable linear algebraic algorithms that leverage sampling to speed up computation. Many of these scalable algorithms are not only essential for direct application but also serve as fundamental subroutines in modern machine learning models. Additionally, the analysis of these methods gives theoretical insight and foundation for more complex methods. Therefore our study of randomized linear algebra routines is essential for the advancement of modern machine learning and data analysis.

In this dissertation, we discuss two topics in randomized numerical linear algebra: stochastic iterative methods for solving linear systems and structured matrix factorizations. First, we discuss the Randomized Kaczmarz method, a specific stochastic iterative method for solving linear systems. The Kaczmarz method has recently grown in popularity due to its speed and low memory requirement.

In chapter 2, we develop a variant based on Randomized Kaczmarz (RK) called Selectable Set Randomized Kaczmarz (SSRK). SSRK is a variant of RK that leverages existing information about the Kaczmarz iterate to identify an adaptive selectable set and thus yields an improved convergence guarantee. In this chapter, we propose a general perspective for selectable set approaches and prove a convergence result for that framework. In addition,

we define two specific selectable set sampling strategies that have competitive convergence guarantees to those of other variants of RK. One selectable set sampling strategy leverages information about the previous iterate, while the other leverages the orthogonality structure of the problem via the Gramian matrix. We complement our theoretical results with numerical experiments that compare our proposed rules with those existing in the literature.

In chapter 3, we consider the problem of recovering a signal $x^* \in \mathbb{R}^n$ from a sequence of linear measurements. This problem arises in areas such as computerized tomography and wireless communications. In this chapter, we consider an online setting in which measurements are sampled one-by-one from some source distribution. We propose solving this problem with a variant of the Kaczmarz method with an additional heavy ball momentum term. Recent work has shown that the Kaczmarz method also enjoys linear convergence when applied to random measurement models, however convergence may be slowed when successive measurements are highly coherent. We demonstrate that the addition of heavy ball momentum may accelerate the convergence of the Kaczmarz method when data is coherent, and provide a theoretical analysis of the method culminating in a linear convergence guarantee for a wide class of source distributions.

In chapter 4, We propose an iterative algorithm based on the randomized Kaczmarz (RK) method to automatically identify subgroups in data and perform linear regression on these groups simultaneously. We prove almost sure convergence for this method, as well as linear convergence in expectation under certain conditions. The result is an interpretable collection of different weight vectors for the regressor variables that capture the different trends within data. Furthermore, we experimentally validate our convergence results by demonstrating the method can successfully identify two trends within simulated data.

Next, we shift gears and discuss structured matrix factorizations. In chapter 5, we propose a variant of non-negative matrix factorization. Non-negative matrix factorization (NMF) is an important technique for obtaining low dimensional representations of datasets. However, classical NMF does not take into account data that is collected at different times or

in different locations, which may exhibit heterogeneity. We resolve this problem by solving a modified NMF objective, Stratified-NMF, that simultaneously learns strata-dependent statistics and a shared topics matrix. We develop multiplicative update rules for this novel objective and prove convergence of the objective. Then, we experiment on synthetic data to demonstrate the efficiency and accuracy of the method. Lastly, we apply our method to three real world datasets and empirically investigate their learned features.

Finally, in chapter 6 we extend an adaptive partially matrix-free Hierarchically Semi-Separable (HSS) matrix construction algorithm by Gorman et al. which uses Gaussian sketching operators to a broader class of Johnson–Lindenstrauss (JL) sketching operators. This structured matrix factorization allows for more efficient matrix operations and factorizations once the matrix is stored in this format. We discuss the details of hierarchically semi-separable matrix construction and trace our proposed method.

CHAPTER 2

Selectable Set Randomized Kaczmarz

This chapter is an adaptation of [YMS22] which is joint work with Jacob Moorman, William Swartworth, Thomas Tu, Daji Landis and Deanna Needell. Jacob Moorman and I proposed the initial idea and contributed the convergence analysis. Additionally, I contributed to the experimental results, writing and revisions of this work.

We propose a new variant to randomized Kaczmarz (RK), a stochastic iterative method for solving linear systems with a low memory requirement. We use a small amount of additional memory to store a selectable set allowing us to save on computation when applying the method, therefore we call our variant selectable set randomized Kaczmarz (SSRK). We show that at the cost of storing additional data, we are able to save some time in computation. In the rest of this chapter we explain the proposed method, prove convergence guarantees and analyze SSRK experimentally. Finally, we highlight an existing gap in Kaczmarz theory where SSRK has strong theoretical guarantees but experimentally does not perform as well as other Kaczmarz variants.

2.1 Introduction

The Kaczmarz method [Kar37], also known as the algebraic reconstruction technique in computed tomography [GBH70], has become a popular method for solving large overdetermined systems of linear equations. The method has abundant applications ranging from digital signal and image processing to statistics and machine learning. We are primarily interested in the the regime of extremely large linear systems, where it may be too expensive to load

a large number of rows into memory. In this setting, the Kaczmarz method is particularly useful as it only requires loading a single row into memory at a time. We also consider sparse systems, which yield additional benefits for the Kaczmarz method; the time required for each iteration scales linearly with the number of nonzero entries in the selected row [Nat01].

To solve a system of equations $\mathbf{A}\mathbf{x} = \mathbf{b}$ with $\mathbf{A} \in \mathbb{R}^{m\times n}$, the Kaczmarz method operates iteratively, beginning with an initial vector $\mathbf{x}^0$ (often $\mathbf{x}^0 = \mathbf{0}$). On each iteration k, an equation $\mathbf{A}_{i_k}\mathbf{x} = b_{i_k}$, or equivalently row index i_k, is chosen and $\mathbf{x}^{k+1}$ is computed as the projection of $\mathbf{x}^k$ onto the set of solutions to that equation. Algebraically, the Kaczmarz update is given by

$$\mathbf{x}^{k+1} = \mathbf{x}^k - \frac{\mathbf{A}_{i_k}\mathbf{x}^k - b_{i_k}}{\|\mathbf{A}_{i_k}\|^2}\mathbf{A}_{i_k}^T, \tag{2.1}$$

where i_k is the index of the chosen equation, $\mathbf{A}_{i_k}$ is the corresponding row of the matrix $\mathbf{A}$, and $\|\cdot\|$ is the Euclidean norm. Algebraically projecting onto the solution of an equation $\mathbf{A}_{i_k}\mathbf{x} = b_{i_k}$ is equivalent to sampling a row index i_k and applying Equation (2.1). Thus we refer to sampling then projecting onto *equations* and sampling then applying Equation (2.1) to *row indices* interchangeably.

Like many iterative methods, the Kaczmarz method utilizes and depends on a sampling strategy to choose the equation for its update at each iteration. Different sampling strategies exhibit different convergence behavior. The first sampling strategy proven to result in linear convergence was a randomized strategy where equations are chosen at random with probabilities proportional to the corresponding squared row norms $\|\mathbf{A}_i\|^2$, which we deem the Randomized Kaczmarz method (RK) [SV09] . Subsequently, many variants of RK have been proposed and shown to converge linearly [BW18b, NSL16, Du19, GR15, HM21, MTM21].

In this chapter, we aim to develop Kaczmarz methods with linear convergence rates that mitigate inefficiencies in classical Kaczmarz methods by leveraging meta-information about the algorithm or problem. For instance, if the equation chosen on iteration k is already solved (i.e., $\mathbf{A}_{i_k}\mathbf{x}^k = b_{i_k}$), then Equation (2.1) reduces to $\mathbf{x}^{k+1} = \mathbf{x}^k$ and the iteration is wasted. Therefore, it is desirable avoid sampling equations that are already solved by the

current iterate $\mathbf{x}^k$. In general, checking whether an equation is solved is as expensive as the update itself. However, if the system of equations $\mathbf{Ax} = \mathbf{b}$ has some structure such as a known Gramian matrix $\mathbf{G} = \mathbf{AA}^T$, it can be possible to keep track of some equations that are known to be solved so that they can be avoided. The set of equations that are not known to be solved is referred to as the *selectable set* [NSL16]. In this work, we consider variants of RK that use a selectable set to avoid wasting iterations. Such variants are referred to as Selectable Set Randomized Kaczmarz methods (SSRK).

2.1.1 Related Work

Several works have considered using more general distributions in RK and have obtained similar convergence guarantees (see e.g. [GR15, NSW16, NSL16] and references therein). A different line of work has focused on sampling strategies that depend on the iterate $\mathbf{x}^k$ and thus change from iteration to iteration [BW18b, NSL16, Du19, HM21]. Most notable in the latter is the Max-Distance Kaczmarz method (MDK), also known as Motzkin's method, which chooses the equation that maximizes the normalized residual $\left|\mathbf{A}_{i_k}\mathbf{x}^k - b_{i_k}\right| / \|\mathbf{A}_{i_k}\|$ on each iteration. The term max-distance refers to the fact that MDK chooses the equation that leads to the largest update, since

$$\frac{\left|\mathbf{A}_{i_k}\mathbf{x}^k - b_{i_k}\right|}{\|\mathbf{A}_{i_k}\|} = \left\|\mathbf{x}^{k+1} - \mathbf{x}^k\right\|.$$

MDK yields a provably optimal per-iteration convergence guarantee at the expense of a high per-iteration computational cost [NSL16].

Several sampling strategies have been proposed that approximate MDK with a cheaper per-iteration cost. For example, the Sampling Kaczmarz Motzkin method (SKM) chooses a random subset of rows and selects the maximum-residual row from that subset [DHN17]. This results in much cheaper per-iteration costs than MDK, while still yielding a provably better convergence guarantee than RK [HM21]. Similarly to MDK, the Relaxed Greedy Randomized Kaczmarz method (RGRK) samples from the equations whose normalized residual $\left|\mathbf{A}_{i_k}\mathbf{x}^k - b_{i_k}\right| / \|\mathbf{A}_{i_k}\|$ exceeds some threshold [BW18a, BW18b]. RGRK has a faster con-

vergence guarantee than RK but a slower guarantee than MDK and is significantly more expensive per-iteration than MDK [BW18b, GMM21]. We compare the convergence guarantee of RGRK [BW18b] with that of SSRK in Section 2.3.2.

Nutini et al. consider several improvements to RK for sparse systems [NSL16]. In particular, this work introduces leveraging the *orthogonality graph*, which is formed from by considering the Gramian matrix $\mathbf{G} = \mathbf{A}\mathbf{A}^T$ as an adjacency matrix for an unweighted graph. In this graph the nodes are the rows of the matrix. Two nodes are joined by an edge if they are *not orthogonal*. The key observation is that a Kaczmarz update for row $\mathbf{A}_i$ only affects residual entries corresponding to adjacent nodes in this graph. This allows for tracking a so-called selectable set, which is the complement of the set of nodes for which the corresponding residual entry is known to be 0. Since sampling non-selectable rows yields no progress, one could hope to speed up RK by restricting the sampling to the selectable set. We consider the selectable set method in more detail, and specialize to several common types of sparse orthogonality graphs.

2.1.2 Contribution

In this chapter, we define a general framework for what we call *Selectable Set Randomized Kaczmarz methods (SSRK, Algorithm 1)*. This is a generalization of the orthogonality graph method proposed by Nutini et al. [NSL16]. We show that SSRK methods converge linearly with a speedup over RK related to the size of the selectable set. We define and analyze two specific SSRK methods, the Non-Repetitive Selectable Set Randomized Kaczmarz method (NSSRK, Algorithm 2) and the Gramian Selectable Set Randomized Kaczmarz method (GSSRK, Algorithm 3). These methods use different strategies to identify the selectable set. We show that NSSRK has a selectable set of size $m - 1$, while the size of the GSSRK selectable set is bounded from below by properties of the matrix $\mathbf{A}$. Finally, we note that the convergence guarantee of NSSRK is the same as that of Relaxed Greedy Randomized Kaczmarz method (RGRK) [BW18a, BW18b] despite converging much slower than RGRK

in practice. This suggests that the convergence guarantee for RGRK is not tight.

2.1.3 Organization

The rest of this chapter is structured as follows. The remainder of this section summarizes the notation that will be used throughout. In Section 2.2 we define SSRK methods and define two specific examples, NSSRK and GSSRK. Then, in Section 2.3, we prove a general convergence guarantee for SSRK methods, presented in Theorem 2, and use it to prove corollaries for specific methods and sampling strategies. Additionally, we discuss connections between the convergence analysis of Algorithm 2 and a popular Kaczmarz method proposed by Bai and Wu [BW18a, BW18b]. Next, in Section 2.4, we examine the improvement of applying Algorithm 3 to problems with structured systems. Then, we show some empirical results in Section 2.5 and finally, in Section 2.6, we summarize our work and provide a short discussion on Kaczmarz sampling strategies.

2.1.4 Notation and Assumption

We consider consistent systems of linear equations $\mathbf{A}\mathbf{x} = \mathbf{b}$ with $\mathbf{A} \in \mathbb{R}^{m\times n}$, $\mathbf{x} \in \mathbb{R}^n$, and right hand side vector (RHS) $\mathbf{b} \in \mathbb{R}^m$. We seek the least-norm solution to the system $\mathbf{x}^\star = \mathbf{A}^\dagger\mathbf{b}$. Throughout this chapter, m will globally represent the number of rows in the system and n will represent the number of columns of the matrix $\mathbf{A}$. Bold uppercase letters represent matrices, bold lowercase letters represent vectors, and standard letters represent scalars. $\mathbf{A}_i$ denotes the i^{th} row of the matrix $\mathbf{A}$, while $\mathbf{b}_i$ denotes the i^{th} element of the vector $\mathbf{b}$. We use $[m]$ as shorthand for the set $\{1, 2, \ldots, m\}$. The norm $\|\cdot\|$ is the Euclidean vector norm and $\|\cdot\|_F$ is the Fröbenius matrix norm. The smallest *nonzero* singular value of $\mathbf{A}$ is $\sigma_{\min}(\mathbf{A})$. The index of the equation chosen on iteration k is $i_k \in [m]$. The matrix $\mathbf{A}$ is presumed to have no rows of all zeros so that $\|\mathbf{A}_i\| > 0$ for all i and the Kaczmarz update (Equation (2.1)) is well defined for any i_k.

The initial iterate is denoted $\mathbf{x}^0$ and the iterate at iteration k is denoted $\mathbf{x}^k$. Likewise

for SSRK methods, the initial selectable set is $\mathcal{S}_0$, often chosen as $\mathcal{S}_0 = [m]$, and subsequent selectable sets are denoted $\mathcal{S}_k$. We define the complement of a selectable set as $\mathcal{S}^C = [m] \backslash \mathcal{S}$. In the analysis of the selectable set, we use the floor $\lfloor \ell \rfloor$ to denote the greatest integer less than or equal to ℓ and the ceiling $\lceil \ell \rceil$ to denote the smallest integer greater than or equal to ℓ. For ease of reference, in Table 2.1 we list the acronyms for the methods that we investigate and analyze in this chapter.

Method acronym	Method name	Reference(s)
RK	Randomized Kaczmarz	Strohmer and Vershynin 2009 [SV09]
MDK	Max-Distance Kaczmarz	Motzkin 1954 [MS54]
SSRK	Selectable Set Randomized Kaczmarz	Algorithm 1
NSSRK	Non-Repetitive Selectable Set Randomized Kaczmarz	Algorithm 2
GSSRK	Gramian Selectable Set Randomized Kaczmarz	Nutini et al. 2016 [NSL16] and Algorithm 3
GRK	Greedy Randomized Kaczmarz	Bai and Wu 2018 [BW18a]
RGRK	Relaxed Greedy Randomized Kaczmarz	Bai and Wu 2018 [BW18b]

Table 2.1: Acronyms of the methods discussed.

The scalars $p_1, p_2, \ldots, p_m$ represent probabilities associated with each equation of the system or equivalently each row of $\mathbf{A}$. We often refer to rows of $\mathbf{A}$ and equations $\mathbf{A}_i \mathbf{x} = \mathbf{b}_i$ interchangeably. We use $\text{Diag}(\mathbf{v})$ to denote the square matrix whose diagonal entries take the values from the vector $\mathbf{v}$ and whose remaining entries are all 0. In particular, we utilize the diagonal matrices of row norms $\mathbf{D} = \text{Diag}(\|\mathbf{A}_1\|, \|\mathbf{A}_2\|, \ldots, \|\mathbf{A}_m\|)$ and probabilities $\mathbf{P} = \text{Diag}(p_1, p_2, \ldots, p_m)$.

The Gramian matrix of $\mathbf{A}$ is $\mathbf{G} = \mathbf{A}\mathbf{A}^T$ where $\mathbf{G} \in \mathbb{R}^{m \times m}$. It has the property $G_{ij} = \langle \mathbf{A}_i, \mathbf{A}_j \rangle$ where $\langle \cdot, \cdot \rangle$ is the dot product between vectors. By the symmetry of the dot product, $\mathbf{G}$ is symmetric with $G_{ij} = G_{ji} = 0$ if and only if rows $\mathbf{A}_i$ and $\mathbf{A}_j$ are orthogonal. Thus, those entries with a nonzero value indicate that the corresponding rows of $\mathbf{A}$ are non-orthogonal. From the Gramian matrix $\mathbf{G}$, we derive a *non-orthogonality graph* where each node represents a row of $\mathbf{A}$ and a nonzero entry $\mathbf{G}_{ij}$ is interpreted as an edge between nodes i and j. We

always assume that our graphs do not contain self edges and allow $G_{ii} \neq 0$ from here and thereafter. Since each node in the non-orthogonality graph represents a row in the matrix $\mathbf{A}$, the number of nodes in the graph is m. For this graph, we let $\mathcal{M}$ denote the maximum independent set, the largest set in which no pair of nodes share an edge. We denote the cardinality and complement of a set as $|\mathcal{M}|$ and $\mathcal{M}^C = [m]\backslash\mathcal{M}$, respectively.

2.2 Selectable set method

Since the convergence behavior of the Kaczmarz method is highly dependent on the sampling strategy used to determine the order of projections, it is important to develop and analyze various sampling techniques. Here, we focus on the framework in which a *selectable set* of equations is identified in each iteration and then an equation is selected from that set, typically at random. Since the Kaczmarz update only improves the solution when selecting an equation that is not already solved, we aim to identify the selectable set that is precisely the set of equations not currently solved.

Definition 1. *A selectable set $\mathcal{S}_k \subset [m]$ for a Kaczmarz method, given a matrix $\mathbf{A}$, vector $\mathbf{b}$ and an iterate $\mathbf{x}^k$, is a set of indices that satisfies $i \notin \mathcal{S}_k \implies \mathbf{A}_i\mathbf{x}^k = \mathbf{b}_i$.*

Based on this definition, if an equation i is sampled from outside the selectable set, then $\mathbf{A}_i\mathbf{x}^k = \mathbf{b}_i$, implying that if row i were chosen for the Kaczmarz update, then $\mathbf{x}^{k+1} = \mathbf{x}^k$. Thus, sampling exclusively from the selectable set automatically guarantees faster convergence than that of a method that selects in the same random fashion from the entire set of equations Note that this is related to, but fundamentally different from, random sampling without replacement. Sampling without replacement indeed guarantees that the same equation is not selected in consecutive iterations, but an equation solved in iteration k need not be solved in even the next iteration. See Section 2.6.1 for more discussion.

In the Selectable Set Randomized Kaczmarz method (SSRK), the equation chosen at each iteration must be sampled from the current selectable set. We assume that a fixed

probability distribution on the equations is given. Then, instead of sampling i_k according to the probabilities $p_1, p_2, \ldots, p_m$ as in RK, SSRK samples i_k conditioned on $i_k \in \mathcal{S}_k$. This can be achieved by repeatedly sampling i_k according to the probabilities $p_1, p_2, \ldots, p_m$ until the condition $i_k \in \mathcal{S}_k$ is satisfied. This rejection sampling is mathematically equivalent to sampling from the explicit distribution $p_i / \sum_{j \in \mathcal{S}_k} p_j$ for $i \in \mathcal{S}_k$ and zero for $i \notin \mathcal{S}_k$ at each iteration. Explicitly updating the sampling distribution at each iteration is advantageous and yields computational improvement when the selectable set is small and explicitly known. Conversely, rejection sampling is advantageous if the selectable set contains a majority of the rows because it bypasses the computational overhead of recomputing the distribution.

Algorithm 1: Selectable Set Randomized Kaczmarz (SSRK)

Input Matrix $\mathbf{A}$, RHS $\mathbf{b}$, initial selectable set $\mathcal{S}_0$, initial iterate $\mathbf{x}^0 \in \text{row}(\mathbf{A})$, probabilities $p_1, p_2, \ldots, p_m > 0$

for $k = 0, 1, \ldots$ **do**

  Sample row i_k according to probabilities $p_1, p_2, \ldots, p_m$ with rejection until $i_k \in \mathcal{S}_k$

  Update $\mathbf{x}^{k+1} = \mathbf{x}^k - \frac{\mathbf{A}_{i_k}\mathbf{x}^k - b_{i_k}}{\|\mathbf{A}_{i_k}\|^2}\mathbf{A}_{i_k}^T$

  Update $\mathcal{S}_{k+1}$ so that $i \notin \mathcal{S}_{k+1} \implies \mathbf{A}_i\mathbf{x}^{k+1} = \mathbf{b}_i$ ▷ See Algorithms 2 and 3 for examples.

end

Output Approximate solution $\mathbf{x}^k$

2.2.1 Non-repetitive selectable set

A simple construction to update the selectable set $\mathcal{S}$ is to begin by including every index in the first selectable set $\mathcal{S}_0 = [m]$. Then, for each subsequent iteration, omit the most recently chosen index from the selectable set so that $\mathcal{S}_{k+1} = [m] \backslash \{i_k\}$. In this construction, $|\mathcal{S}_0| = m$ and $|\mathcal{S}_k| = m - 1$ for $k > 0$. In order to save memory, there is no need to explicitly construct $\mathcal{S}_k$. It is sufficient to keep track of the previously sampled row which corresponds

to $\mathcal{S}_k^C = \{i_{k-1}\}$. Sampling with rejection from this selectable set has a probability $(m-1)/m$ of succeeding on each attempt, since $|\mathcal{S}_k| = m-1$. The total number of attempts required to sample i_k is thus geometrically distributed with mean $m/(m-1)$. We refer to this method as the Non-Repetitive Selectable Set method (NSSRK) Algorithm 2.

Algorithm 2: Non-Repetitive Selectable Set Randomized Kaczmarz (NSSRK)

Input Matrix $\mathbf{A}$, RHS $\mathbf{b}$, initial iterate $\mathbf{x}^0 \in \text{row}(\mathbf{A})$, probabilities $p_1, p_2, \ldots, p_m > 0$

$\mathcal{S}_0 = [m]$

for $k = 0, 1, \ldots$ **do**

  Sample row i_k according to probabilities $p_1, p_2, \ldots, p_m$ with rejection until $i_k \in \mathcal{S}_k$

  Update $\mathbf{x}^{k+1} = \mathbf{x}^k - \frac{\mathbf{A}_{i_k}\mathbf{x}^k - b_{i_k}}{\|\mathbf{A}_{i_k}\|^2}\mathbf{A}_{i_k}^T$

  Set $\mathcal{S}_{k+1} = [m] \backslash \{i_k\}$

end

Output Approximate solution $\mathbf{x}^k$

2.2.2 Gramian-Based Selectable Set

A second method to update the selectable set, originally proposed in Nutini et al. is to leverage the Gramian $\mathbf{G} = \mathbf{A}\mathbf{A}^T$ of the matrix $\mathbf{A}$ [NSL16, Sep16]. In many structured problems we have access to both the matrix $\mathbf{A}$ and its Gramian $\mathbf{G}$. One example of such a problem is graph semi-supervised learning [BLS18]. The Gramian, by definition, has the property that $G_{ij} = \langle \mathbf{A}_i, \mathbf{A}_j \rangle$. That is, the ij^{th} entry of the Gramian is the inner product between the i^{th} and j^{th} rows of $\mathbf{A}$. So $G_{ij} = 0$ if and only if rows $\mathbf{A}_i$ and $\mathbf{A}_j$ are orthogonal. Based on Lemma 1, stated and proven below, we will develop an update to the selectable set based on the Gramian.

Lemma 1. *If an equation $\mathbf{A}_j\mathbf{x} = b_j$ solved by the iterate $\mathbf{x}^k$, and if $\mathbf{A}_{i_k}$ is orthogonal to $\mathbf{A}_j$ (i.e. $G_{i_k j} = 0$), then the equation is also solved by the next iterate $\mathbf{x}^{k+1}$.*

Proof. Let $\mathbf{x}^k$ and j satisfy $\mathbf{A}_j\mathbf{x}^k = b_j$, and suppose $\mathbf{A}_{i_k}$ is orthogonal to $\mathbf{A}_j$. Multiplying by $\mathbf{A}_j$ on the left of both sides of the Kaczmarz update (Equation (2.1)) results in

$$\begin{aligned}
\mathbf{A}_j\mathbf{x}^{k+1} &= \mathbf{A}_j\left(\mathbf{x}^k - \frac{\mathbf{A}_{i_k}x^k - b_{i_k}}{\|\mathbf{A}_{i_k}\|^2}\mathbf{A}_{i_k}^T\right) \\
&= \mathbf{A}_j\mathbf{x}^k - \mathbf{A}_j\frac{\mathbf{A}_{i_k}x^k - b_{i_k}}{\|\mathbf{A}_{i_k}\|^2}\mathbf{A}_{i_k}^T \\
&= \mathbf{A}_j\mathbf{x}^k - \frac{\mathbf{A}_{i_k}x^k - b_{i_k}}{\|\mathbf{A}_{i_k}\|^2}\mathbf{A}_j\mathbf{A}_{i_k}^T \\
&= \mathbf{A}_j\mathbf{x}^k - \frac{\mathbf{A}_{i_k}x^k - b_{i_k}}{\|\mathbf{A}_{i_k}\|^2}\langle\mathbf{A}_{i_k}, \mathbf{A}_j\rangle.
\end{aligned}$$

Using the assumption that $\mathbf{A}_j\mathbf{x}^k = b_j$,

$$\mathbf{A}_j\mathbf{x}^{k+1} = b_j - \frac{\mathbf{A}_{i_k}x^k - b_{i_k}}{\|\mathbf{A}_{i_k}\|^2}\langle\mathbf{A}_{i_k}, \mathbf{A}_j\rangle.$$

Finally, by the assumption that $\mathbf{A}_{i_k}$ is orthogonal to $\mathbf{A}_j$,

$$\begin{aligned}
\mathbf{A}_j\mathbf{x}^{k+1} &= b_j - \frac{\mathbf{A}_{i_k}x^k - b_{i_k}}{\|\mathbf{A}_{i_k}\|^2}0 \\
&= b_j.
\end{aligned}$$

□

Recall that any equation $\mathbf{A}_j\mathbf{x} = b_j$ that is not selectable must be solved by the iterate $\mathbf{x}^k$. Thus, from Lemma 1, we know that if $j \notin \mathcal{S}_k$ and if i_k satisfies $G_{i_kj} = 0$, then the equation $\mathbf{A}_j\mathbf{x} = b_j$ is still solved by the next iterate, i.e. $\mathbf{A}_j\mathbf{x}^{k+1} = b_j$. This suggests that any unselectable index $j \notin \mathcal{S}_k$ for which $G_{i_kj} = 0$ should remain unselectable on iteration $k+1$, since the corresponding equation is still solved. The Gramian Selectable Set Randomized Kaczmarz method (GSSRK), Algorithm 3, is based on this observation. In GSSRK, only those indexes j with $G_{i_kj} \neq 0$ are reintroduced to the selectable set at each iteration.

2.3 Convergence analysis

Now, we turn to proving convergence results for Algorithm 1. First, we prove a one-step convergence result for the general selectable set method with a fixed probability distribution.

Algorithm 3: Gramian Selectable Set Randomized Kaczmarz (GSSRK) [NSL16]

Input Matrix $\mathbf{A}$, RHS $\mathbf{b}$, Gramian $\mathbf{G} := \mathbf{A}\mathbf{A}^T$, initial iterate $\mathbf{x}^0 \in \text{row}(\mathbf{A})$, probabilities $p_1, p_2, \ldots, p_m > 0$

$\mathcal{S}_0 = [m]$

for $k = 0, 1, \ldots$ **do**

  Sample row i_k according to probabilities $p_1, p_2, \ldots, p_m$ with rejection until $i_k \in \mathcal{S}_k$

  Update $\mathbf{x}^{k+1} = \mathbf{x}^k - \frac{\mathbf{A}_{i_k}\mathbf{x}^k - b_{i_k}}{\left\|\mathbf{A}_{i_k}\right\|^2}\mathbf{A}_{i_k}^T$

  $\mathcal{S}_{k+1} = (\mathcal{S}_k \cup \{j : G_{i_k j} \neq 0\})\backslash\{i_k\}$

end

Output Approximate solution $\mathbf{x}^k$

We analyze a single iteration of the general case with an arbitrary sampling distribution, and a known selectable set S_k. We then focus on specific probability distributions common in the literature [SV09], and prove an improvement in the convergence constant which is inversely proportional to the size of the selectable set. This speedup is roughly what one should expect. For example if rows are sampled uniformly, then RK wastes an $1 - |S_k|/m$ fraction of iterations on updates which make no progress, whereas SSRK avoids this.

Theorem 2. *The iterates of Selectable Set Randomized Kaczmarz (Algorithm 1) satisfy*

$$\mathbb{E}_k \left\|\mathbf{x}^{k+1} - \mathbf{x}^\star\right\|^2 \leq \left(1 - \frac{\sigma_{\min}^2(\mathbf{P}^{\frac{1}{2}}\mathbf{D}^{-1}\mathbf{A})}{\sum_{j\in\mathcal{S}_k} p_j}\right)\left\|\mathbf{x}^k - \mathbf{x}^\star\right\|^2,$$

where $\mathbf{x}^\star$ *is the least-norm solution,* $\mathbf{P} = \text{Diag}(p_1, p_2, \ldots, p_m)$*, and* $\mathbf{D} = \text{Diag}(\|\mathbf{A}_1\|, \|\mathbf{A}_2\|, \ldots, \|\mathbf{A}_m\|)$ *when* $\sum_{j\in\mathcal{S}_k} p_j \neq 0$.

Proof. From the update formula Equation (2.1), we derive the usual update for the squared error

$$\left\|\mathbf{x}^{k+1} - \mathbf{x}^\star\right\|^2 = \left\|\mathbf{x}^k - \mathbf{x}^\star\right\|^2 - \frac{\left|\mathbf{A}_{i_k}\mathbf{x}^k - b_{i_k}\right|^2}{\left\|\mathbf{A}_{i_k}\right\|^2}.$$

Letting $\mathbb{E}_k$ denote the expectation conditioned on $i_0, i_1, \ldots, i_{k-1}$, we take this conditional expectation on both sides

$$\begin{aligned}\mathbb{E}_k\left[\left\|\mathbf{x}^{k+1}-\mathbf{x}^\star\right\|^2\right] &= \left\|\mathbf{x}^k-\mathbf{x}^\star\right\|^2 - \mathbb{E}_k\left[\frac{\left|\mathbf{A}_{i_k}\mathbf{x}^k - b_{i_k}\right|^2}{\left\|\mathbf{A}_{i_k}\right\|^2}\right] \\ &= \left\|\mathbf{x}^k-\mathbf{x}^\star\right\|^2 - \sum_{i\in\mathcal{S}_k}\frac{p_i}{\sum_{j\in\mathcal{S}_k}p_j}\frac{\left|\mathbf{A}_i\mathbf{x}^k-\mathbf{b}_i\right|^2}{\left\|\mathbf{A}_i\right\|^2}.\end{aligned}$$

Pulling the normalizing constant $1/\sum_{j\in\mathcal{S}_k}p_j$ out of the summation,

$$\mathbb{E}_k\left[\left\|\mathbf{x}^{k+1}-\mathbf{x}^\star\right\|^2\right] = \left\|\mathbf{x}^k-\mathbf{x}^\star\right\|^2 - \frac{1}{\sum_{j\in\mathcal{S}_k}p_j}\sum_{i\in\mathcal{S}_k}p_i\frac{\left|\mathbf{A}_i\mathbf{x}^k-\mathbf{b}_i\right|^2}{\left\|\mathbf{A}_i\right\|^2}.$$

By the definition of the selectable set, we know that $i\notin\mathcal{S}_k \implies \mathbf{A}_i\mathbf{x}^k-\mathbf{b}_i=0$ so we can extend our sum over all rows

$$\mathbb{E}_k\left[\left\|\mathbf{x}^{k+1}-\mathbf{x}^\star\right\|^2\right] = \left\|\mathbf{x}^k-\mathbf{x}^\star\right\|^2 - \frac{1}{\sum_{j\in\mathcal{S}_k}p_j}\sum_{i\in[m]}p_i\frac{\left|\mathbf{A}_i\mathbf{x}^k-\mathbf{b}_i\right|^2}{\left\|\mathbf{A}_i\right\|^2}.$$

By definition of $\mathbf{x}^\star$ we can rewrite $\mathbf{b}_i=\mathbf{A}_i\mathbf{x}^\star$

$$\begin{aligned}\mathbb{E}_k\left[\left\|\mathbf{x}^{k+1}-\mathbf{x}^\star\right\|^2\right] &= \left\|\mathbf{x}^k-\mathbf{x}^\star\right\|^2 - \frac{1}{\sum_{j\in\mathcal{S}_k}p_j}\sum_{i\in[m]}p_i\frac{\left|\mathbf{A}_i\mathbf{x}^k-\mathbf{A}_i\mathbf{x}^\star\right|^2}{\left\|\mathbf{A}_i\right\|^2} \\ &= \left\|\mathbf{x}^k-\mathbf{x}^\star\right\|^2 - \frac{1}{\sum_{j\in\mathcal{S}_k}p_j}\sum_{i\in[m]}p_i\frac{\left|\mathbf{A}_i(\mathbf{x}^k-\mathbf{x}^\star)\right|^2}{\left\|\mathbf{A}_i\right\|^2} \\ &= \left\|\mathbf{x}^k-\mathbf{x}^\star\right\|^2 - \frac{1}{\sum_{j\in\mathcal{S}_k}p_j}\sum_{i\in[m]}\left|\frac{p_i^{\frac{1}{2}}}{\left\|\mathbf{A}_i\right\|}\mathbf{A}_i(\mathbf{x}^k-\mathbf{x}^\star)\right|^2.\end{aligned}$$

We now rewrite the summation as a norm of a matrix-vector multiplication using the previously defined $\mathbf{P}$ and $\mathbf{D}$ matrices:

$$\mathbb{E}_k\left[\left\|\mathbf{x}^{k+1}-\mathbf{x}^\star\right\|^2\right]=\left\|\mathbf{x}^k-\mathbf{x}^\star\right\|^2-\frac{1}{\sum_{j\in\mathcal{S}_k}p_j}\left\|\mathbf{P}^{\frac{1}{2}}\mathbf{D}^{-1}\mathbf{A}(\mathbf{x}^k-\mathbf{x}^\star)\right\|^2.$$

Now, we establish the lower bound $\left\|\mathbf{P}^{\frac{1}{2}}\mathbf{D}^{-1}\mathbf{A}(\mathbf{x}^k-\mathbf{x}^\star)\right\|^2\geq\sigma_{\min}^2(\mathbf{P}^{\frac{1}{2}}\mathbf{D}^{-1}\mathbf{A})\left\|\mathbf{x}^k-\mathbf{x}^\star\right\|^2$, where $\sigma_{\min}$ is the smallest *nonzero* singular value based on the proof technique in Zouzias and Freris [ZF13]. By the assumption that $\mathbf{A}$ has no zero rows, $\mathbf{D}$ is symmetric positive definite (SPD). Likewise, since the probabilities $p_1,p_2,\ldots,p_m$ are positive, $\mathbf{P}$ is SPD. Since $\mathbf{D}$ and $\mathbf{P}$ are SPD, so are $\mathbf{P}^{\frac{1}{2}}$ and $\mathbf{D}^{-1}$. Thus, $\text{row}(\mathbf{P}^{\frac{1}{2}}\mathbf{D}^{-1}\mathbf{A})$ is equal to $\text{row}(\mathbf{A})$. The vector $\mathbf{x}^\star$ belongs to $\text{row}(\mathbf{A})$ because it is the least-norm solution to $\mathbf{Ax}=\mathbf{b}$. Additionally, the iterate $\mathbf{x}^k$ belongs to $\text{row}(\mathbf{A})$ because the initial iterate $\mathbf{x}^0$ belongs to $\text{row}(\mathbf{A})$ as does the direction of the Kaczmarz update at each iteration. Since the iterate $\mathbf{x}^k$ and the least-norm solution $\mathbf{x}^\star$ both belong to $\text{row}(\mathbf{A})$, so does their difference $\mathbf{x}^k-\mathbf{x}^\star$. Finally, recalling that $\text{row}(\mathbf{A})=\text{row}(\mathbf{P}^{\frac{1}{2}}\mathbf{D}^{-1}\mathbf{A})$, we see that $\mathbf{x}^k-\mathbf{x}^\star\in\text{row}(\mathbf{P}^{\frac{1}{2}}\mathbf{D}^{-1}\mathbf{A})$ and thus $\left\|\mathbf{P}^{\frac{1}{2}}\mathbf{D}^{-1}\mathbf{A}(\mathbf{x}^k-\mathbf{x}^\star)\right\|^2\geq\sigma_{\min}^2(\mathbf{P}^{\frac{1}{2}}\mathbf{D}^{-1}\mathbf{A})\left\|\mathbf{x}^k-\mathbf{x}^\star\right\|^2$. Applying this lower bound, we arrive at the desired result

$$\begin{aligned}\mathbb{E}_k\left[\left\|\mathbf{x}^{k+1}-\mathbf{x}^\star\right\|^2\right]&=\left\|\mathbf{x}^k-\mathbf{x}^\star\right\|^2-\frac{1}{\sum_{j\in\mathcal{S}_k}p_j}\left\|\mathbf{P}^{\frac{1}{2}}\mathbf{D}^{-1}\mathbf{A}(\mathbf{x}^k-\mathbf{x}^\star)\right\|^2\\&\leq\left\|\mathbf{x}^k-\mathbf{x}^\star\right\|^2-\frac{1}{\sum_{j\in\mathcal{S}_k}p_j}\sigma_{\min}^2(\mathbf{P}^{\frac{1}{2}}\mathbf{D}^{-1}\mathbf{A})\left\|\mathbf{x}^k-\mathbf{x}^\star\right\|^2\\&=\left(1-\frac{\sigma_{\min}^2(\mathbf{P}^{\frac{1}{2}}\mathbf{D}^{-1}\mathbf{A})}{\sum_{j\in\mathcal{S}_k}p_j}\right)\left\|\mathbf{x}^k-\mathbf{x}^\star\right\|^2.\end{aligned}$$

□

2.3.1 Corollaries

We now take a closer at how Theorem 2 applies to Algorithms 1 to 3 for two specific common choices of the probabilities $p_1,p_2,\ldots,p_m$. In particular, we analyze uniform probabilities $p_i=1/m$ in Corollary 3 and squared row norm probabilities $p_i=\|\mathbf{A}_i\|^2/\|\mathbf{A}\|_F^2$ in Corollary 5. For either choice of probabilities, the convergence guarantees vary depending on the selectable set $\mathcal{S}_k$ at each iteration.

Corollary 3. *When using uniform probabilities $p_i = 1/m$, the iterates of Algorithm 1 satisfy*

$$\mathbb{E}_k \left\|\mathbf{x}^{k+1} - \mathbf{x}^\star\right\|^2 \leq \left(1 - \frac{\sigma_{\min}^2(\mathbf{D}^{-1}\mathbf{A})}{|\mathcal{S}_k|}\right) \left\|\mathbf{x}^k - \mathbf{x}^\star\right\|^2,$$

where $\mathbf{D} = \mathrm{Diag}(\|A_1\|, \|A_2\|, \ldots, \|A_m\|)$.

Proof. Apply Theorem 2 with $p_i = \frac{1}{m}$. □

Corollary 3 shows that Algorithm 1 with uniform probabilities $p_i = 1/m$ achieves a convergence guarantee that depends on the number of selectable rows $|\mathcal{S}_k|$. The fewer the number of selectable rows, the faster the convergence guarantee. If nearly all rows are selectable at every iteration, Corollary 3 recovers the known convergence guarantee $(1 - \sigma_{\min}^2(\mathbf{D}^{-1}\mathbf{A})/m)$ for RK with uniform probabilities [NSL16]. In Algorithm 2, all but one row are selectable at each iteration, so the convergence guarantee is trivially slightly faster than that of RK as shown in Corollary 4.

Corollary 4. *When using uniform probabilities $p_i = \frac{1}{m}$, the iterates of Algorithm 2 satisfy*

$$\mathbb{E}_0 \left\|\mathbf{x}^1 - \mathbf{x}^\star\right\|^2 \leq \left(1 - \frac{\sigma_{\min}^2(\mathbf{D}^{-1}\mathbf{A})}{m}\right) \left\|\mathbf{x}^0 - \mathbf{x}^\star\right\|^2$$

$$\textit{and} \qquad \mathbb{E}_k \left\|\mathbf{x}^{k+1} - \mathbf{x}^\star\right\|^2 \leq \left(1 - \frac{\sigma_{\min}^2(\mathbf{D}^{-1}\mathbf{A})}{m-1}\right) \left\|\mathbf{x}^k - \mathbf{x}^\star\right\|^2 \qquad \textit{for} \qquad k \geq 1,$$

where $\mathbf{D} = \mathrm{Diag}(\|A_1\|, \|A_2\|, \ldots, \|A_m\|)$.

Proof. Substitute $|\mathcal{S}_0| = m$ and $|\mathcal{S}_k| = m - 1$ for $k \geq 1$ in Corollary 3. □

As discussed, using uniform probabilities results in a simple relationship between the number of selectable rows and the convergence guarantee at each iteration. In contrast, using squared row norm probabilities $p_i = \|\mathbf{A}_i\|^2 / \|\mathbf{A}\|_F^2$ results in a slightly more complicated relationship between the selectable set $\mathcal{S}_k$ and the convergence guarantee. This relationship is shown in Corollary 5.

Corollary 5. *When using probabilities proportional to the squared row norms* $p_i = \|\mathbf{A}_i\|^2 / \|\mathbf{A}\|_F^2$*, the iterates of Algorithm 1 satisfy*

$$\mathbb{E}_k \left\|\mathbf{x}^{k+1} - \mathbf{x}^\star\right\|^2 \leq \left(1 - \frac{\sigma_{\min}^2(\mathbf{A})}{\sum_{j\in\mathcal{S}_k} \|\mathbf{A}_j\|^2}\right) \left\|\mathbf{x}^k - \mathbf{x}^\star\right\|^2.$$

Proof. When $p_i = \|\mathbf{A}_i\|^2 / \|\mathbf{A}\|_F^2$, we have

$$\mathbf{P}^{\frac{1}{2}}\mathbf{D}^{-1} = \text{Diag}\left(\frac{\|\mathbf{A}_1\|}{\|\mathbf{A}\|_F}, \frac{\|\mathbf{A}_2\|}{\|\mathbf{A}\|_F}, \ldots, \frac{\|\mathbf{A}_m\|}{\|\mathbf{A}\|_F}\right) \text{Diag}\left(\frac{1}{\|\mathbf{A}_1\|}, \frac{1}{\|\mathbf{A}_2\|}, \ldots, \frac{1}{\|\mathbf{A}_m\|}\right) = \frac{1}{\|\mathbf{A}\|_F}\mathbf{I}.$$

Substituting this into Theorem 2 along with the probabilities $p_i = \|\mathbf{A}_i\|^2 / \|\mathbf{A}\|_F^2$, we find

$$\begin{aligned}
\mathbb{E}_k \left\|\mathbf{x}^{k+1} - \mathbf{x}^\star\right\|^2 &\leq \left(1 - \frac{\sigma_{\min}^2(\frac{1}{\|\mathbf{A}\|_F}\mathbf{A})}{\sum_{j\in\mathcal{S}_k} \frac{\|\mathbf{A}_j\|^2}{\|\mathbf{A}\|_F^2}}\right) \left\|\mathbf{x}^k - \mathbf{x}^\star\right\|^2 \\
&= \left(1 - \frac{\frac{1}{\|\mathbf{A}\|_F^2}\sigma_{\min}^2(\mathbf{A})}{\frac{1}{\|\mathbf{A}\|_F^2}\sum_{j\in\mathcal{S}_k} \|\mathbf{A}_j\|^2}\right) \left\|\mathbf{x}^k - \mathbf{x}^\star\right\|^2 \\
&= \left(1 - \frac{\sigma_{\min}^2(\mathbf{A})}{\sum_{j\in\mathcal{S}_k} \|\mathbf{A}_j\|^2}\right) \left\|\mathbf{x}^k - \mathbf{x}^\star\right\|^2.
\end{aligned}$$

□

Corollary 5 shows that Algorithm 1 with squared row norm probabilities $p_i = \|\mathbf{A}_i\|^2 / \|\mathbf{A}\|_F^2$ achieves a convergence guarantee that depends on the quantity $\sum_{j\in\mathcal{S}_k} \|\mathbf{A}_j\|^2$. This quantity is the squared Fröbenius norm of the row-submatrix of $\mathbf{A}$ composed of those rows that are selectable. When all of the rows of $\mathbf{A}$ have roughly the same norm, Corollary 5 suggests that Algorithm 1 converges faster when fewer rows are selectable. When the rows have very different norms, the relationship between the selectable set and the convergence guarantee is not so simple.

When nearly all rows are selectable at every iteration, Corollary 5 recovers the known convergence guarantee $\left(1 - \sigma_{\min}^2(\mathbf{A})/\|\mathbf{A}\|_F^2\right)$ for RK with squared row norm probabilities $p_i = \|\mathbf{A}_i\|^2 / \|\mathbf{A}\|_F^2$ [SV09]. In Algorithm 2, all but one row are selectable at each iteration, so the convergence guarantee is similar to that of RK as shown in Corollary 6.

Corollary 6. *When using probabilities proportional to the squared row norms* $p_i = \|\mathbf{A}_i\|^2 / \|\mathbf{A}\|_F^2$*, the iterates of Algorithm 2 satisfy*

$$\mathbb{E}_0 \left\|\mathbf{x}^1 - \mathbf{x}^\star\right\|^2 \leq \left(1 - \frac{\sigma_{\min}^2(\mathbf{A})}{\|\mathbf{A}\|_F^2}\right) \left\|\mathbf{x}^0 - \mathbf{x}^\star\right\|^2$$

$$\textit{and} \qquad \mathbb{E}_k \left\|\mathbf{x}^{k+1} - \mathbf{x}^\star\right\|^2 \leq \left(1 - \frac{\sigma_{\min}^2(\mathbf{A})}{\|\mathbf{A}\|_F^2 - \left\|\mathbf{A}_{i_{k-1}}\right\|^2}\right) \left\|\mathbf{x}^k - \mathbf{x}^\star\right\|^2 \qquad \textit{for} \qquad k \geq 1.$$

Proof. For iteration $k = 0$, substitute $\mathcal{S}_0 = [m]$ in Corollary 5. For iterations $k \geq 1$, substituting $\mathcal{S}_k = [m] \backslash \{i_{k-1}\}$ in Corollary 5 shows the desired result

$$\begin{aligned}
\mathbb{E}_k \left\|\mathbf{x}^{k+1} - \mathbf{x}^\star\right\|^2 &\leq \left(1 - \frac{\sigma_{\min}^2(\mathbf{A})}{\sum_{j \in [m] \backslash \{i_{k-1}\}} \|\mathbf{A}_j\|^2}\right) \left\|\mathbf{x}^k - \mathbf{x}^\star\right\|^2 \\
&= \left(1 - \frac{\sigma_{\min}^2(\mathbf{A})}{\sum_{j \in [m]} \|\mathbf{A}_j\|^2 - \left\|\mathbf{A}_{i_{k-1}}\right\|^2}\right) \left\|\mathbf{x}^k - \mathbf{x}^\star\right\|^2 \\
&= \left(1 - \frac{\sigma_{\min}^2(\mathbf{A})}{\|\mathbf{A}\|_F^2 - \left\|\mathbf{A}_{i_{k-1}}\right\|^2}\right) \left\|\mathbf{x}^k - \mathbf{x}^\star\right\|^2.
\end{aligned}$$

□

From Corollary 6, we see that NSSRK (Algorithm 2) is expected to converge faster on iterations when some row with large norm is not selectable. This is essentially an artifact of the sampling scheme. Rows with large norms are chosen disproportionately often; when such a row is not selectable, the sampling is more uniform, and convergence improves.

Next we compare the convergence analysis of NSSRK with convergence guarantee from the popular Kaczmarz method proposed in Bai and Wu [BW18b]. While the NSSRK method is relatively simple and does not yield large improvement over non-selectable set methods, the convergence guarantee corresponds to the same convergence guarantee as in [BW18b]. This leads us to believe that there may be a theory gap in Kaczmarz convergence analysis.

2.3.2 Comparison with Relaxed Greedy Randomized Kaczmarz (RGRK) theory

The Relaxed Greedy Randomized Kaczmarz method (RGRK)[BW18b] is similar to MDK in that both methods use sampling strategies that are biased toward rows with larger normalized residuals $\left|\mathbf{A}_i\mathbf{x}^k - \mathbf{b}_i\right| / \left\|\mathbf{A}_i\right\|$. In particular, RGRK considers only rows that satisfy

$$\frac{\left|\mathbf{A}_i\mathbf{x}^k - \mathbf{b}_i\right|^2}{\left\|\mathbf{A}_i\right\|^2} \geq \theta \max_{j\in[m]} \left(\frac{\left|\mathbf{A}_j\mathbf{x}^k - b_j\right|^2}{\left\|\mathbf{A}_j\right\|^2} \right) + (1-\theta)\, \frac{\left\|\mathbf{A}\mathbf{x}^k - \mathbf{b}\right\|^2}{\left\|\mathbf{A}\right\|_F^2} \qquad \text{for some } \theta \in [0,1] \quad (2.2)$$

and samples the row $\mathbf{A}_{i_k}$ with probability proportional to the squared residual $\left|\mathbf{A}_i\mathbf{x}^k - \mathbf{b}_i\right|^2$ from among such rows. RGRK satisfies the convergence result[BW18b]

$$\mathbb{E}_k \left\|\mathbf{x}^{k+1} - \mathbf{x}^\star\right\|^2 \leq \left(1 - \left(\theta \frac{\left\|\mathbf{A}\right\|_F^2}{\gamma} + (1-\theta)\right) \frac{\sigma_{\min}^2(\mathbf{A})}{\left\|\mathbf{A}\right\|_F^2}\right) \left\|\mathbf{x}^k - \mathbf{x}^\star\right\|^2 \qquad \text{for} \qquad k \geq 1, \quad (2.3)$$

where

$$\gamma = \max_{i\in[m]} \sum_{j=1, j\neq i}^{m} \left\|\mathbf{A}_j\right\|^2 = \left\|\mathbf{A}\right\|_F^2 - \min_{i\in[m]} \left\|\mathbf{A}_i\right\|^2. \quad (2.4)$$

This convergence result is optimized by the parameter $\theta = 1$ for which RGRK is equivalent to MDK. With $\theta = 1$, Equation (2.3) simplifies to

$$\mathbb{E}_k \left\|\mathbf{x}^{k+1} - \mathbf{x}^\star\right\|^2 \leq \left(1 - \frac{\sigma_{\min}^2(\mathbf{A})}{\gamma}\right) \left\|\mathbf{x}^k - \mathbf{x}^\star\right\|^2 \qquad \text{for} \qquad k \geq 1. \quad (2.5)$$

Coincidentally, NSSRK satisfies Equation (2.5) for squared row norm probabilities.

Corollary 7. *When using probabilities proportional to the squared row norms* $p_i = \left\|\mathbf{A}_i\right\|^2 / \left\|\mathbf{A}\right\|_F^2$*, the iterates of NSSRK (Algorithm 2) satisfy Equation* (2.5)*.*

Proof. By Corollary 6, we have the desired result

$$\begin{aligned}
\mathbb{E}_k \left\|\mathbf{x}^{k+1} - \mathbf{x}^\star\right\|^2 &\leq \left(1 - \frac{\sigma_{\min}^2(\mathbf{A})}{\left\|\mathbf{A}\right\|_F^2 - \left\|\mathbf{A}_{i_{k-1}}\right\|^2}\right) \left\|\mathbf{x}^k - \mathbf{x}^\star\right\|^2 \\
&\leq \left(1 - \frac{\sigma_{\min}^2(\mathbf{A})}{\left\|\mathbf{A}\right\|_F^2 - \min_{i\in[m]} \left\|\mathbf{A}_i\right\|^2}\right) \left\|\mathbf{x}^k - \mathbf{x}^\star\right\|^2 \\
&= \left(1 - \frac{\sigma_{\min}^2(\mathbf{A})}{\gamma}\right) \left\|\mathbf{x}^k - \mathbf{x}^\star\right\|^2.
\end{aligned}$$

□

Since NSSRK satisfies Equation (2.5) while RGRK only satisfies Equation (2.3), one might incorrectly assume that NSSRK will outperform RGRK. However, as we observe in Section 2.5, this is not the case. RGRK significantly outperforms NSSRK and GSSRK, neglecting CPU time. This discrepancy between convergence results and observed performance suggests that Equation (2.3) is not the tightest possible convergence result for RGRK. Indeed, RGRK was recently shown to satisfy the tighter convergence result [GMM21] based on the constant $\sigma_\infty^2(\mathbf{A})$, defined as

$$\sigma_\infty^2(\mathbf{A}) = \min_{\substack{\mathbf{x}\in \mathrm{row}(\mathbf{A})\setminus\{\mathbf{x}^\star\} \\ \exists j \text{ s.t. } \mathbf{A}_j(\mathbf{x}-\mathbf{x}^\star)=0}} \left(\max_i \frac{|\mathbf{A}_i(\mathbf{x}-\mathbf{x}^\star)|^2}{\|\mathbf{A}_i\|^2 \|\mathbf{x}-\mathbf{x}^\star\|^2} \right), \tag{2.6}$$

where $\mathbf{x}^\star \in \mathrm{row}(\mathbf{A})$. The improved convergence result is as follows:

$$\mathbb{E}_k \left\|\mathbf{x}^{k+1}-\mathbf{x}^\star\right\|^2 \leq \left(1-\theta\sigma_\infty^2(\mathbf{A})-(1-\theta)\frac{\sigma_{\min}^2(\mathbf{A})}{\|\mathbf{A}\|_F^2}\right)\left\|\mathbf{x}^k-\mathbf{x}^\star\right\|^2 \qquad \text{for} \qquad k>1.$$

This convergence result is optimized by the parameter $\theta = 1$, see Lemma 8, leading to the improved convergence bound of

$$\mathbb{E}_k \left\|\mathbf{x}^{k+1}-\mathbf{x}^\star\right\|^2 \leq \left(1-\sigma_\infty^2(\mathbf{A})\right)\left\|\mathbf{x}^k-\mathbf{x}^\star\right\|^2 \qquad \text{for} \qquad k>1.$$

The improvement of this new bound is explained by the relationship between $\sigma_\infty^2(\mathbf{A})$ and $\sigma_{\min}^2(\mathbf{A})$, which is matrix dependent and discussed in more detail in other works [NSL16, GMM21]. The following lemma, Lemma 8, attempts to explain this relationship (second $\leq$) and the fact that $\theta = 1$ is the optimal parameter (first $\leq$).

Lemma 8. *Let* $\mathbf{A}$ *be a matrix then*

$$\frac{\sigma_{\min}^2(\mathbf{A})}{\|\mathbf{A}\|_F^2} \leq \frac{\sigma_{\min}^2(\mathbf{A})}{\gamma} \leq \sigma_\infty^2(\mathbf{A}). \tag{2.7}$$

Where $\sigma_{\min}^2(\mathbf{A})$ *is the smallest* nonzero *singular value,* $\sigma_\infty^2(\mathbf{A})$ *is defined by Equation* (2.6) *and* γ *is defined by Equation* (2.4).

Proof. By definition $\gamma = \|\mathbf{A}\|_F^2 - \min_{i\in[m]} \|\mathbf{A}_i\|^2 \leq \|\mathbf{A}\|_F^2$ so

$$\frac{\sigma_{\min}^2(\mathbf{A})}{\|\mathbf{A}\|_F^2} \leq \frac{\sigma_{\min}^2(\mathbf{A})}{\gamma}.$$

Since $\sigma_{\min}^2(\mathbf{A})$ is the smallest nonzero singular value and $\mathbf{x}^\star \in \text{row}(\mathbf{A})$ we have the following:

$$\sigma_{\min}^2(\mathbf{A}) = \min_{\mathbf{x}\in\text{row}(\mathbf{A})\setminus\{\mathbf{x}^\star\}} \sum_i \frac{|\mathbf{A}_i(\mathbf{x}-\mathbf{x}^\star)|^2}{\|\mathbf{x}-\mathbf{x}^\star\|^2}.$$

Multiplying and dividing by $p_i = \frac{\|\mathbf{A}_i\|^2}{\|\mathbf{A}\|_F^2}$ then rearranging terms yields

$$\begin{aligned}\sigma_{\min}^2(\mathbf{A}) &= \min_{\mathbf{x}\in\text{row}(\mathbf{A})\setminus\{\mathbf{x}^\star\}} \sum_i \frac{\|\mathbf{A}_i\|^2}{\|\mathbf{A}\|_F^2} \frac{\|\mathbf{A}\|_F^2}{\|\mathbf{A}_i\|^2} \frac{|\mathbf{A}_i(\mathbf{x}-\mathbf{x}^\star)|^2}{\|\mathbf{x}-\mathbf{x}^\star\|^2} \\ &= \|\mathbf{A}\|_F^2 \min_{\mathbf{x}\in\text{row}(\mathbf{A})\setminus\{\mathbf{x}^\star\}} \sum_i p_i \frac{|\mathbf{A}_i(\mathbf{x}-\mathbf{x}^\star)|^2}{\|\mathbf{A}_i\|^2 \|\mathbf{x}-\mathbf{x}^\star\|^2}.\end{aligned}$$

Imposing the constraint that $\mathbf{x} \in \text{row}(\mathbf{A}) \setminus \{\mathbf{x}^\star\}$ *and* $\exists j$ s.t. $\mathbf{A}_j(\mathbf{x}-\mathbf{x}^\star) = 0$ then noting that $|\mathbf{A}_j(\mathbf{x}-\mathbf{x}^\star)|^2 = 0$ results in

$$\begin{aligned}\sigma_{\min}^2(\mathbf{A}) &= \|\mathbf{A}\|_F^2 \min_{\mathbf{x}\in\text{row}(\mathbf{A})\setminus\{\mathbf{x}^\star\}} \sum_i p_i \frac{|\mathbf{A}_i(\mathbf{x}-\mathbf{x}^\star)|^2}{\|\mathbf{A}_i\|^2 \|\mathbf{x}-\mathbf{x}^\star\|^2} \\ &\leq \|\mathbf{A}\|_F^2 \min_{\substack{\mathbf{x}\in\text{row}(\mathbf{A})\setminus\{\mathbf{x}^\star\} \\ \exists j \text{ s.t. } \mathbf{A}_j(\mathbf{x}-\mathbf{x}^\star)=0}} \sum_i p_i \frac{|\mathbf{A}_i(\mathbf{x}-\mathbf{x}^\star)|^2}{\|\mathbf{A}_i\|^2 \|\mathbf{x}-\mathbf{x}^\star\|^2} \\ &= \|\mathbf{A}\|_F^2 \min_{\substack{\mathbf{x}\in\text{row}(\mathbf{A})\setminus\{\mathbf{x}^\star\} \\ \exists j \text{ s.t. } \mathbf{A}_j(\mathbf{x}-\mathbf{x}^\star)=0}} \sum_{i\neq j} p_i \frac{|\mathbf{A}_i(\mathbf{x}-\mathbf{x}^\star)|^2}{\|\mathbf{A}_i\|^2 \|\mathbf{x}-\mathbf{x}^\star\|^2}.\end{aligned}$$

Taking a max over the rows i for the fraction of the summation

$$\begin{aligned}\sigma_{\min}^2(\mathbf{A}) &\leq \|\mathbf{A}\|_F^2 \min_{\substack{\mathbf{x}\in\text{row}(\mathbf{A})\setminus\{\mathbf{x}^\star\} \\ \exists j \text{ s.t. } \mathbf{A}_j(\mathbf{x}-\mathbf{x}^\star)=0}} \sum_{i\neq j} p_i \frac{|\mathbf{A}_i(\mathbf{x}-\mathbf{x}^\star)|^2}{\|\mathbf{A}_i\|^2 \|\mathbf{x}-\mathbf{x}^\star\|^2} \\ &\leq \|\mathbf{A}\|_F^2 \min_{\substack{\mathbf{x}\in\text{row}(\mathbf{A})\setminus\{\mathbf{x}^\star\} \\ \exists j \text{ s.t. } \mathbf{A}_j(\mathbf{x}-\mathbf{x}^\star)=0}} \max_\ell \frac{|\mathbf{A}_\ell(\mathbf{x}-\mathbf{x}^\star)|^2}{\|\mathbf{A}_\ell\|^2 \|\mathbf{x}-\mathbf{x}^\star\|^2} \sum_{i\neq j} p_i.\end{aligned}$$

Bounding $\sum_{i\neq j} p_i$ from above by $1-\min_i p_i$, simplifying and applying the definition of $\sigma_\infty^2(\mathbf{A})$

$$\begin{aligned}
\sigma^2_{\min}(\mathbf{A}) &\leq \|\mathbf{A}\|_F^2 \min_{\substack{\mathbf{x}\in \mathrm{row}(\mathbf{A})\setminus\{\mathbf{x}^\star\} \\ \exists j \text{ s.t. } \mathbf{A}_j(\mathbf{x}-\mathbf{x}^\star)=0}} \max_\ell \frac{|\mathbf{A}_\ell(\mathbf{x}-\mathbf{x}^\star)|^2}{\|\mathbf{A}_\ell\|^2 \|\mathbf{x}-\mathbf{x}^\star\|^2} \sum_{i\neq j} p_i \\
&\leq \|\mathbf{A}\|_F^2 \min_{\substack{\mathbf{x}\in \mathrm{row}(\mathbf{A})\setminus\{\mathbf{x}^\star\} \\ \exists j \text{ s.t. } \mathbf{A}_j(\mathbf{x}-\mathbf{x}^\star)=0}} \max_\ell \frac{|\mathbf{A}_\ell(\mathbf{x}-\mathbf{x}^\star)|^2}{\|\mathbf{A}_\ell\|^2 \|\mathbf{x}-\mathbf{x}^\star\|^2} (1-\min_i p_i) \\
&= \|\mathbf{A}\|_F^2 \sigma^2_\infty(\mathbf{A})(1-\min_i p_i)
\end{aligned}$$

Finally, distributing the $\|\mathbf{A}\|_F^2$ and substituting the definition of p_i and γ

$$\begin{aligned}
\sigma^2_{\min}(\mathbf{A}) &\leq \|\mathbf{A}\|_F^2 \sigma^2_\infty(\mathbf{A})(1-\min_i p_i) \\
&= \sigma^2_\infty(\mathbf{A})(\|\mathbf{A}\|_F^2 - \min_i \|\mathbf{A}_i\|^2) \\
&= \sigma^2_\infty(\mathbf{A})\gamma.
\end{aligned}$$

Thus, $\frac{\sigma^2_{\min}(\mathbf{A})}{\gamma} \leq \sigma^2_\infty(\mathbf{A})$ as desired. □

The improvement of this new bound is explained by the difference between $\frac{\sigma^2_{\min}(\mathbf{A})}{\gamma}$ and $\sigma^2_\infty(\mathbf{A})$, which we quantify by the above Lemma 8. It is matrix dependent and described in more detail in Gower et al. [GMM21] and Nutini et al. [NSL16]

2.4 Lower bounds on size of Gramian selectable set size

Investigating the size of the selectable set is essential for understanding the convergence of the GSSRK method, Algorithm 3 [NSL16, Sep16]. In Section 2.3, we proved that the convergence of the GSSRK method is dependent on the size of the selectable set, the smaller the selectable set the better the convergence guarantee is. In this section, we prove a lower bound on the size of the Gramian selectable set after $O(m)$ iterations where m is the number of rows in the matrix. For many structured problems the lower bound on the size of the selectable set is cm where $c \in (0,1)$. This means that the convergence guarantee only improves by a constant factor in comparison to Kaczmarz methods that do not use a selectable set.

In particular, here we prove a general method for calculating a lower bound on the Gramian based selectable set. Then we apply this lower bound to some structured problems. Our lower bound can also be used as a heuristic to assess in which cases applying the GSSRK method could lead to a large speedup. When the Gramian is sparse, meaning that there is a lot of orthogonality between rows of the matrix, the GSSRK method will likely yield improved convergence rates.

To develop our lower bound on the selectable set, we consider the Gramian of the matrix, $\mathbf{G} = \mathbf{A}\mathbf{A}^T$ as an adjacency matrix and examine the graph formed from this matrix, the *non-orthogonality graph*. This graph contains m nodes, each node representing a row of the original matrix $\mathbf{A}$, node i represents row $\mathbf{A}_i$. There is an edge between nodes i and j if $G_{ij} = \langle \mathbf{A}_i, \mathbf{A}_j \rangle \neq 0$, meaning that rows i and j of $\mathbf{A}$ are not orthogonal. So the graph encodes the orthogonality structure of the original matrix $\mathbf{A}$. Two rows are orthogonal if and only if they do not share an edge in the graph. Next we will use the non-orthogonality graph and a graph theoretic approach to develop a lower bound on the selectable set for Algorithm 3.

$$\mathbf{A} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 1 & 0 \\ 1 & 1 & 1 \end{bmatrix} \mathbf{G} = \mathbf{A}\mathbf{A}^T = \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 2 & 1 & 2 \\ 0 & 1 & 1 & 1 \\ 1 & 2 & 1 & 3 \end{bmatrix} \begin{matrix} \mathcal{M} = \{1,2\},\{1,3\} \\ |\mathcal{M}| = 2 \end{matrix}$$

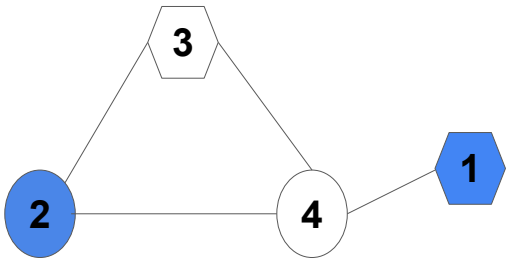


Figure 2.1: The non-orthogonality graph constructed from the Gramian $\mathbf{G}$ of the above matrix $\mathbf{A}$. This undirected graph is connected because each node is reachable from every other node along the edges of the graph. The size of the maximal independent set of the graph is $2 = |\mathcal{M}|$ because the largest set of nodes that that do not share an edge among them is 2. If we consider the subgraph formed from nodes 1 and 2 or nodes 1 and 3, these graph have no edges. No larger sets can be created because if we add any additional nodes to these sets, the induced subgraphs will contain edges.

Theorem 9. *Given a system of equations* $\mathbf{A}, \mathbf{b}$ *and an initial iterate* $\mathbf{x}^0$ *such that* $\mathbf{A}\mathbf{x}^0 \neq \mathbf{b}$. *Then the selectable set* $\mathcal{S}_k$ *from Algorithm 3 satisfies* $m - |\mathcal{M}| \leq |\mathcal{S}_k| \leq m - 1$ *after* $O(m)$ *iterations for all* k. *Where* $|\mathcal{M}|$ *is the size of the maximal independent set formed by considering the Gramian matrix* $\mathbf{G} = \mathbf{A}\mathbf{A}^T$ *as an adjacency matrix of a* connected graph.

Proof. Consider the graph constructed from the Gramian of our matrix. Since this graph is connected and we have an unsolved equation, after $O(m)$ iterations each equation will become unsolved at least once and therefore be selectable. We now consider the maximum size of the set of unselectable rows after $O(m)$ iterations. The only rows that are unselectable are rows that have been selected by GSSRK, meaning that all neighbors of this node must be selectable by the sampling process. Therefore if $\mathcal{S}_k^c$ is the set of unselectable rows. If $r_1, r_2 \in \mathcal{S}_k^c$, then

r_1 and r_2 cannot share an edge. Otherwise, the row more recently selected in the method, without loss of generality r_1 is necessarily unselectable then r_2 would necessarily become selectable because it is a neighbor of r_1. Thus, each row in $\mathcal{S}_k^c$ must not be a neighbor of any other row in $\mathcal{S}_k^c$. So the maximum size of $\mathcal{S}_k^c$ is at most the size of the maximum independent set of the graph, $\mathcal{M}$. Thus the size of the selectable set is at least: $m - |\mathcal{M}|$.
For $k \geq 1$, $\mathcal{S}_k$ does not include row i_{k-1} by the definition of the Kaczmarz update, at least one row is always solved, so $|\mathcal{S}_k| \leq m - 1$. □

An algebraic interpretation of $|\mathcal{M}|$ is that this number corresponds to the size of the largest set of rows of $\mathbf{A}$ such that all elements in this set are pairwise-orthogonal. Equivalently, this is the number of rows in the largest submatrix of the rows of $\mathbf{A}$ that can be constructed $\mathbf{Q} \in \mathbb{R}^{|\mathcal{M}| \times n}$, such that $\mathbf{Q}\mathbf{Q}^T = \mathbf{D}$ where $\mathbf{D}$ is a diagonal matrix. Since at most $\mathbf{A}$ has $|\mathcal{M}|$ pairwise orthogonal rows, by Lemma 1 all of the equations $\mathbf{A}_i\mathbf{x} = b_i$ where $\mathbf{A}_i$ is a row of $\mathbf{Q}$ can be simultaneously solved by a Kaczmarz method.

Next, we prove that the proposed lower bound of the the selectable set in Theorem 9 is tight and describe a row sampling scheme to achieve this lower bound.

Proposition 10. *There exists a sampling strategy for a Gramian based Selectable Set Kaczmarz method that achieves the lower bound of $m - |\mathcal{M}|$ from Theorem 9 on the size of the Gramian based selectable set.*

Proof. To achieve this lower bound on the selectable set, we construct a sequence of rows to be selected, sampled. After the method selects each row in this sequence once, the size of the selectable set is exactly $m - |\mathcal{M}|$. Let $\mathcal{M}$ denote the maximum independent set of Gramian graph G. Then the sampling strategy in which we select each row from $\mathcal{M}$ in succession results in each of the rows of $\mathcal{M}$ being not selectable. Since none of the rows in $\mathcal{M}$ are neighbors in the graph, meaning that all rows in $\mathcal{M}$ are pairwise orthogonal, selecting any row in $\mathcal{M}$ does not impact the selectable or unselectable status of any other row in $\mathcal{M}$. Thus, if we select each row in $\mathcal{M}$ once, there will be exactly $|\mathcal{M}|$ unselectable rows, meaning

that size of the selectable set will be exactly $m - |\mathcal{M}|$. □

If the maximum independent set formed from the Gramian is large, then the size of the selectable set will be small for Algorithm 3. Thus, based on our convergence analysis for selectable set methods, Theorem 2, we will have a convergence guarantee improvement over Kaczmarz methods that do not use a selectable set. We examine and provide examples of this convergence improvement experimentally in Section 2.5.

Next, we investigate how we can apply Theorem 9 to basic structured non-orthogonality graphs and examine the size of their selectable sets. We select the following graphs as they correspond to non-orthogonality graphs constructed from very sparse systems, and many serve as simple but motivating examples for such structures that arise in practice. We anticipate that by examining these specific structured graphs we may be able to understand when applying GSSRK to a system will yield significant convergence improvement.

First, we consider the path graph. This is a graph in which the adjacency matrix (the Gramian) has nonzero entries exclusively along the diagonal, the super diagonal and sub-diagonal, representative of sparse matrices from networks applications.

Corollary 11 (Path Gramian)**.** *For the case of a path graph Gramian with m vertices, the size of the selectable set is lower bounded by $\left\lfloor \frac{m}{2} \right\rfloor$.*

Proof. Taking every other vertex in the path, we obtain a maximum independent set of size $\left\lceil \frac{m}{2} \right\rceil$. We then apply Theorem 9 to obtain the bound $|\mathcal{S}_k| \geq m - \left\lceil \frac{m}{2} \right\rceil = \left\lfloor \frac{m}{2} \right\rfloor$. □

Next we consider a star graph whose adjacency matrix contains nonzeros on the diagonal, along a single row ℓ, and the column ℓ. This is representative of a matrix that is mostly sparse but may contain some nonsparse rows. An example of this structure arises in collaborative filtering where data may represent user prescribed ratings. Most users will have few ratings representing sparse rows, but some super-users will have many ratings, representing the nonsparse rows of the matrix.

Corollary 12 (Star Gramian)**.** *For the case of a star graph Gramian with m vertices, the size of the selectable set is lower bounded by* 1*.*

Proof. The $m-1$ leaves form a maximum independent set of size $m-1$. We then apply Theorem 9 to obtain the bound $|\mathcal{S}_k| \geq m-(m-1)=1$. □

We note that the high degree node of the star represents a nonsparse row which may cause minimal convergence improvement when applying selectable set methods. Next we consider a cycle graph in which the adjacency matrix has nonzeros along the diagonal, super diagonal, sub-diagonal, top right corner and the bottom left corner. This represents a sparse matrix similar to Corollary 11 with slightly less sparsity.

Corollary 13 (Cycle Gramian)**.** *For the case of a cycle graph Gramian with m vertices, the size of the selectable set is lower bounded by $\lceil \frac{m}{2} \rceil$.*

Proof. Taking every other vertex in the cycle except the last one if m is odd, we obtain a maximum independent set of size $\lfloor \frac{m}{2} \rfloor$. We then apply Theorem 9 to obtain the bound $|\mathcal{S}_k| \geq m - \lfloor \frac{m}{2} \rfloor = \lceil \frac{m}{2} \rceil$. □

Next we consider a banded graph in which the adjacency matrix contains nonzeros along the diagonal, ℓ super diagonals and ℓ sub-diagonals. This adjacency matrix occurs commonly in semi-supervised graph learning tasks in which only data for the k nearest neighbors of a node is kept.

Corollary 14 (Banded Gramian)**.** *For the case of a banded matrix Gramian with m vertices, bandwidth ℓ (upper and lower bandwidth ℓ), the size of the selectable set is lower bounded by $\lfloor \frac{\ell m}{\ell+1} \rfloor$.*

Proof. The maximal independent set can be constructed by taking the first node, node 1, which is adjacent to ℓ other nodes. Then taking node $1+\ell+1$ to avoid the ℓ neighbors of the first node. This node has at most 2ℓ neighbors but shares ℓ of them with the first node.

So now we can take node $1 + 2\ell + 2$. Repeating this process we deduce that we can select one node from every $\ell + 1$ nodes. Resulting in a maximal independent set of size $\left\lceil \frac{m}{\ell+1} \right\rceil$. We then apply Theorem 9 to obtain the bound $|\mathcal{S}_k| \geq m - \left\lceil \frac{m}{\ell+1} \right\rceil = \left\lfloor \frac{m\ell+m}{\ell+1} - \frac{m}{\ell+1} \right\rfloor = \left\lfloor \frac{\ell m}{\ell+1} \right\rfloor$. □

Note that a path graph is an example of a banded matrix with bandwidth one. Finally, we consider a symmetric ℓ regular graph. The adjacency matrix for this graph has ℓ nonzero entries on each row and corresponding column, excluding the diagonal entries thus $\ell < m$. Since this adjacency matrix corresponds to an undirected graph, it is symmetric. This adjacency matrix pattern is likely to occur when applying a generalized k-nearest neighbors algorithm to graph data.

Corollary 15 (ℓ-regular Gramian)**.** *For the case of an ℓ-regular graph with m vertices and degree $\ell < m$, the size of the selectable set is lower bounded by* $\max(\left\lceil \frac{m}{2} \right\rceil, \ell)$.

Proof. The size of the maximum independent set of an ℓ-regular graph [Ros64] is upper bounded by
$\min(\left\lfloor \frac{m}{2} \right\rfloor, m - \ell)$. Thus, we can apply Theorem 9 to obtain the bound $|\mathcal{S}_k| \geq \max(\left\lceil \frac{m}{2} \right\rceil, \ell)$.
□

Theorem 9 gives us a lower bound on the size of the selectable set in many cases, see Table 2.2. However, to apply this theorem directly, we need an upper bound on the size of the maximum independent set. For some classes of graphs, while a lower bound may be achievable, the upper bound is the trivial bound of $m - 1$.

Gramian graph	Maximum independent set size	Lower bound on size of selectable set $\lvert\mathcal{S}\rvert$
path graph	$\lceil \frac{m}{2} \rceil$	$\lfloor \frac{m}{2} \rfloor$
star graph	m-1	1
cycle graph	$\lfloor \frac{m}{2} \rfloor$	$\lceil \frac{m}{2} \rceil$
banded matrix graph, bandwidth ℓ	$\lceil \frac{m}{\ell+1} \rceil$	$\lfloor \frac{\ell m}{\ell+1} \rfloor$
ℓ-regular graph	$\min(\lfloor \frac{m}{2} \rfloor, m-\ell)$	$\max(\lceil \frac{m}{2} \rceil, \ell)$

Table 2.2: Lower bounds on size of the selectable set for structured Gramian problems.

2.5 Experiments

We evaluate RK, NSSRK, GSSRK, and GRK on a series of synthetic and real world matrices from the SuiteSparse matrix library (implementation of methods used available here: `https://github.com/jdmoorman/kaczmarz-algorithms/`). For each matrix, we plot the squared error norm versus iteration number for 20000 iterations. Results were averaged over 100 trials. Each line corresponds to the average error at the iteration over the 100 trials and the shading corresponds to one standard deviation above the mean and one standard deviation below the mean at each iteration. The vectors $\mathbf{x}^\star$ and $\mathbf{b}$ for all of the experiments are constructed by taking a standard random normal vector $\mathbf{v}$ (of mean 0 and standard deviation 1 entries), computing $\mathbf{x}^\star = \frac{\mathbf{A}^T\mathbf{v}}{\|\mathbf{A}^T\mathbf{v}\|}$, then applying $\mathbf{A}$ such that $\mathbf{b} = \mathbf{A}\mathbf{x}^\star$.

The two synthetic matrices that we consider are the circulant matrix and the 3-banded matrix. The circulant matrix is a 100×100 matrix with $\sigma_{\min} = 0.690$, see Figure 2.2a and Figure 2.3a. The matrix has non-zeros on the diagonal, sub-diagonal and top right corner. The non-zero entries of each row are the row number i divided by $\sqrt{2}$ so each row has non-zero entries $\frac{i}{\sqrt{2}}$. The Gramian of the circulant matrix corresponds to the cycle graph Gramian described in Corollary 13. The 3-banded matrix is a 100×100 matrix with standard

random normal entries along the diagonal and the three rows above and below the diagonal producing a matrix with bandwidth 3 and $\sigma_{\min} = 0.0180$, see Figure 2.2b and Figure 2.3b. The Gramian of this matrix corresponds to a banded matrix graph with bandwidth 6, in Corollary 14.

The two real world matrices we consider are Cities and the transpose of the N_pid from the SuiteSparse matrix library [DH11]. The Cities matrix is a dense matrix of size 55×46 with $\sigma_{\min} = 0.271$, see Figure 2.2c and Figure 2.3c. The transpose of the N_pid matrix is a sparse matrix of size 3923×3625 with 8054 nonzero entries that has $\sigma_{\min} = 0.0690$, see Figure 2.2d and Figure 2.3d.

The four algorithms that we consider are RK, NSSRK, GSSRK, and GRK, each of which has differing computational complexity. RK has a computational complexity of $\mathcal{O}(n)$ per iteration, where n is the number of columns in the matrix. Similarly, NSSRK also has a computational complexity of $\mathcal{O}(n)$, requiring $\mathcal{O}(1)$ to update the selectable set. Unlike NSSRK, GSSRK has the added overhead of updating the selectable set at each iteration requiring $\mathcal{O}(n+m)$ at each iteration, $\mathcal{O}(n)$ to compute the Kaczmarz update, Equation (2.1), and $\mathcal{O}(m)$, where m is the number of rows, to look at a row of the Gramian matrix and update the selectable set. Additionally, if it is necessary to pre-compute the Gramian with standard matrix multiplication, this requires $\mathcal{O}(nm^2)$. Finally, the most computationally costly method per iteration is GRK, which requires $\mathcal{O}(nm)$ at each iteration as a full residual computation, $\mathbf{A}\mathbf{x}_k - \mathbf{b}$, is required to construct the sampling set and distribution. Of course, these are all generic bounds, and different implementations may lead to improvements, such as parallelization, fast multiplies, and other specific uses of the system's structure.

In all four of the matrices pictured, as well as all the other matrices, we evaluated our methods on, we saw little to no difference between the performance of RK and NSSRK. GRK also outperformed RK, NSSRK, and GSSRK on every matrix. However, we did observe some differences in the performance of GSSRK relative to the other methods. In some examples, we see that GSSRK performs the same as RK and NSSRK, while in others, it performs

slightly better.

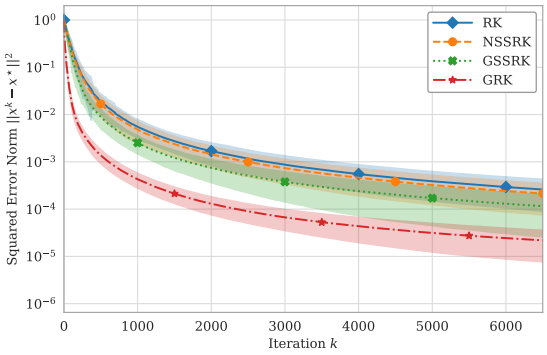
In Figure 2.2b and Figure 2.2a, our two synthetic matrices, we see that one of them (Figure 2.2b) displays very little difference in performance between GSSRK and RK, while the other (Figure 2.2a) has a much larger gap in performance between the two. This is due to the difference in size of their respective selectable sets. In the circulant matrix, each row is only non-orthogonal to two others, while each row in the banded matrix is non-orthogonal to six others. With a sparser non-orthogonality graph, we expect to see smaller selectable sets, which leads to improved performance.

We observe that in Figure 2.2c there is no performance gap between GSSRK and RK on the Cities matrix. Because the Cities matrix has no rows that are mutually orthogonal, the selectable set for this matrix includes every row, except for the one that was just picked, and thus the GSSRK method is equivalent to NSSRK. In Figure 2.2d, we see that GSSRK outperforms RK. Computing the Gramian of the N_pid matrix, we see that it is much sparser, with fewer than 0.1% of the entries being nonzero. Thus, the selectable set is much smaller, leading to improved performance, consistent with our theoretical results. We note that RK, NSSRK and GSSRK all require a sampling distribution that is commonly either the uniform or the row norm distribution [SV09] while GRK requires a choice of θ that was set to 0.5 as in [BW18a]. The results from both uniform and row norm distributions are shown and both display similar comparisons, where GRK performs best but is most costly, followed by GSSRK, then NSSRK, then RK.

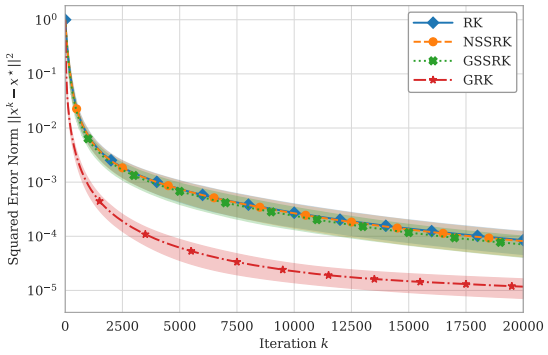
Next, we validate our theoretical results experimentally by varying $\sigma_{\min}$ and examining the convergence rates of GSSRK with *uniform probabilities*, Algorithm 3. Since the SSRK convergence result, Theorem 2, depends on $\sigma_{\min}$, the smallest nonzero singular value, we construct three matrices with varying $\sigma_{\min}$ and observe their convergence rates. We construct circulant matrices of sizes $50 \times 50, 100 \times 100$ and 150×150 with ones along the diagonal, sub-diagonal and top right corner and corresponding $\sigma_{\min} = 0.126, \sigma_{\min} = 0.063$ and $\sigma_{\min} = 0.042$. The results shown in Figure 2.4 support our theoretical guarantees. The results of the

experiment on the circulant matrices were averaged over 100 trials for each matrix. The shading corresponds to one standard deviation above the mean and one standard deviation below the mean.

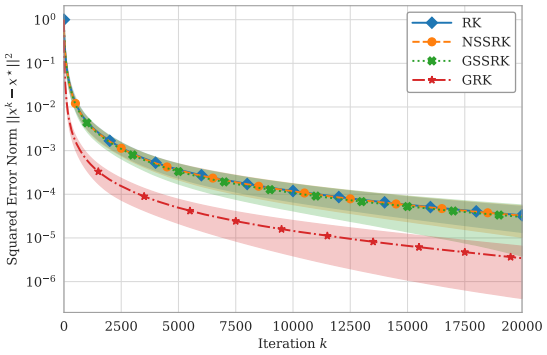
In conclusion, our experiments support our theoretical bounds and we observe that NSSRK and RK perform nearly identically. Since NSSRK has a minimal impact on the row sampling in comparison to RK methods, we do not expect improvement over RK specifically when m is large. Additionally we observe that GSSRK outperformed RK in some cases. These cases include matrices with many orthogonal rows which is likely to occur when solving sparse systems. Even though we have improved convergence guarantees for both GSSRK and NSSRK over RK, we observe that the empirical performance of the algorithms may not reflect this. Finally, we note that GRK always outperforms GSSRK and NSSRK even though both methods have identical or improved theoretical guarantees over GRK when considering a $\sigma_{\min}(\mathbf{A})$ bound. This supports the belief that there is a gap in the theoretical understanding of GRK and perhaps other Adaptive Kaczmarz methods [GMM21].



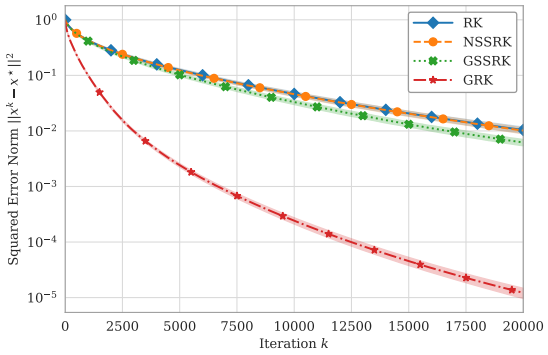
(a) Circulant Matrix Uniform Distribution $\sigma_{\min} = 0.690$



(b) Banded Matrix Uniform Distribution $\sigma_{\min} = 0.0180$

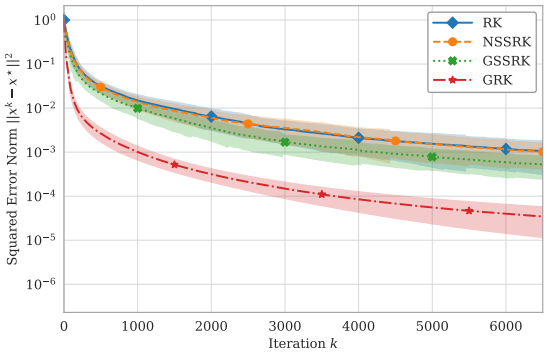


(c) Cities Matrix Uniform Distribution $\sigma_{\min} = 0.271$

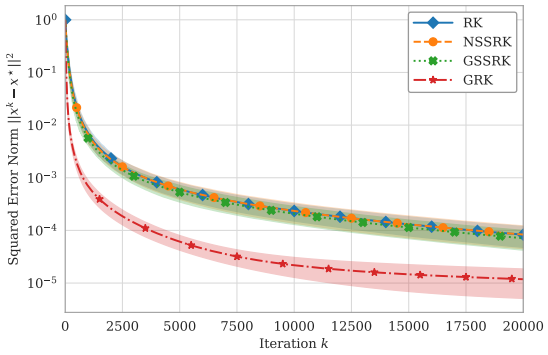


(d) N_Pid Matrix Uniform Distribution $\sigma_{\min} = 0.0690$

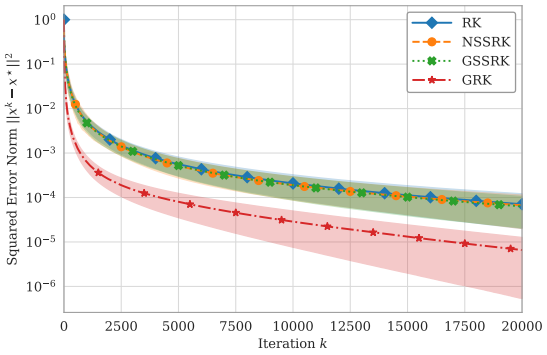
Figure 2.2: Squared error norm versus iteration for RK, NSSRK, GSSRK, and GRK (RGRK with $\theta = 1/2$) using *uniform row probabilities* for RK, NSSRK and GSSRK. Results were averaged over 100 trials. Figure 2.2b is a 3-banded matrix, Figure 2.2a is a circulant matrix, and Figures 2.2c and 2.2d are two real world matrices, Cities and N_pid all described in detail above. The shading denotes one standard deviation.



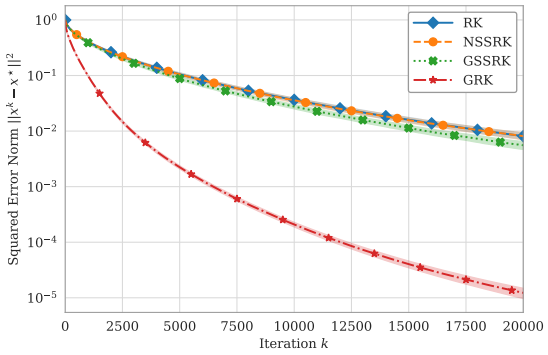
(a) Circulant Matrix Row Norm Distribution $\sigma_{\min} = 0.690$



(b) Banded Matrix Row Norm Distribution $\sigma_{\min} = 0.0180$



(c) Cities Matrix Row Norm Distribution $\sigma_{\min} = 0.271$



(d) N_PID Matrix Row Norm Distribution $\sigma_{\min} = 0.0690$

Figure 2.3: Squared error norm versus iteration for RK, NSSRK, GSSRK, and GRK (RGRK with $\theta = 1/2$) using *row norm probabilities* for RK, NSSRK and GSSRK. Results were averaged over 100 trials. Figure 2.3b is a 3-banded matrix, Figure 2.3a is a circulant matrix, and Figures 2.3c and 2.3d are two real world matrices, Cities and N_pid all described in detail above. The shading denotes one standard deviation.

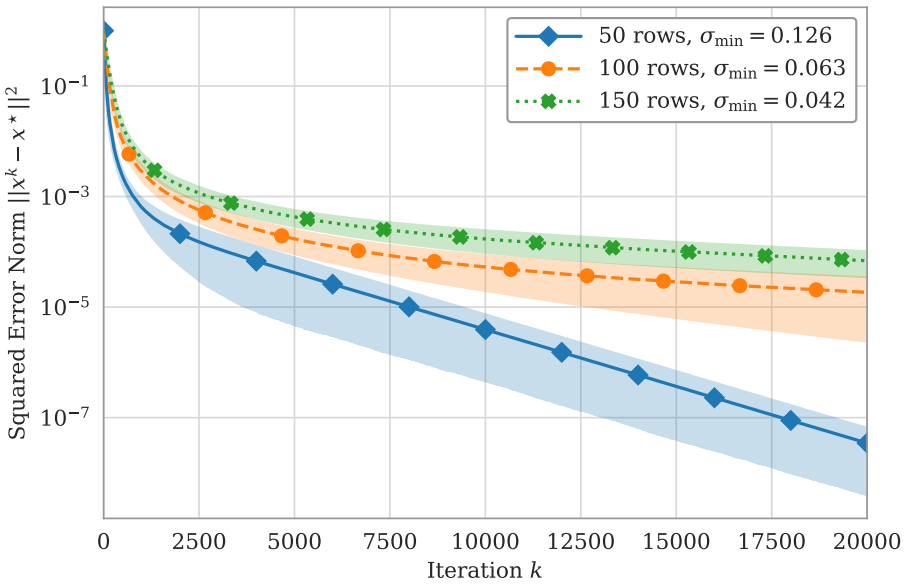


Figure 2.4: Squared error norm versus iteration for GSSRK with uniform row distribution on varying circulant matrix sizes, corresponding to varying $\sigma_{\min}$, to confirm that as $\sigma_{\min}$ increases the convergence rate improves. 100 trials on circulant matrices of sizes: $50 \times 50, 100 \times 100$ and 150×150. Each curve corresponds to the iteration average over the 100 trials and the shading corresponds to one standard deviation above and one standard deviation below the mean of norm-squared errors at each iteration.

2.6 Conclusion

In this chapter, we studied various selectable set approaches for the Kaczmarz method. These aimed to accelerate the standard approach of uniform random row sampling by attempting to move the iterates larger distances in each iteration, thereby reaching the desired solution in fewer steps. We proposed and analyzed a general selectable set randomized Kaczmarz method (Algorithm 1) in which rows are sampled from a selectable set $\mathcal{S}$. The selectable set, $\mathcal{S}$, is updated at each iteration of the method and satisfies the condition that given a row of the matrix j, $j \notin \mathcal{S} \implies \mathbf{A}_j \mathbf{x}^k = b_j$. We proved a general convergence result for this method, Theorem 9, in which we proved that there is an improvement inversely proportional to the size of the selectable set over Kaczmarz methods that sample from all rows.

After we defined the general selectable set framework we examined several strategies such as the simple non-repetitive strategy (Algorithm 2) and the Gramian based strategy (Algorithm 3). Although the non-repetitive strategy was quite simple and perhaps naive, it led to interesting theoretical results that are comparable to those in state of the art methods such as the methods proposed in [BW18a, BW18b], see Section 2.3.2. This led us to believe that there is a theory gap in existing Kaczmarz literature and that a tighter analysis could be possible for some Kaczmarz methods.

The Gramian based strategy leveraged the orthogonality structure of the matrix which is encoded in the Gramian, $\mathbf{G} = \mathbf{A}\mathbf{A}^T$. There are many structured problems in which both the matrix $\mathbf{A}$ and the Gramian $\mathbf{G}$ are known such as graph-based semi-supervised learning [BLS18]. Leveraging the Gramian to update the selectable set yielded tighter convergence bounds than using the non-repetitive strategy, but with the added overhead of updating the selectable set using the Gramian. Bounds on GSSRK were originally proposed by Nutini et al. [NSL16]; we generalized their results by constructing a bound in terms of the singular values of the original matrix, instead of singular values of submatrices, which may be less natural to compute in applications. We discussed the lower bound on the size of the selectable set for the Gramian update when dealing with structured problems, and noted that the size of the

selectable set is usually on the order of number of rows $O(m)$. This relatively large size meant that there is a constant improvement of the convergence guarantee over corresponding non-selectable set methods. Finally, we provided some numerical experiments on both synthetic and real-world matrices from the SuiteSparse package [DH11] that demonstrated the benefits of using selectable sets and showcase how the variations compare.

2.6.1 Future directions

We addressed the problem of projecting onto a row of an equation that has already been solved through sampling from a selectable set $\mathcal{S}$ of unsolved equations. In our methods, we consider the p_i values as a fixed distribution that we renormalize based on the selectable set. Based on our one step convergence result, Theorem 2, one could try to optimize the probabilities p_i themselves each iteration instead of drawing from this fixed distribution. This would require additional computational cost but may have the benefit of faster convergence rates.

A different simple yet surprisingly effective modification of the probability distribution at each iteration is *sampling without replacement* [PJM21, RR12]. In these methods a row can only be sampled again after all of the other rows of the matrix have been sampled. This contrasts the standard method of sampling *with replacement* in which there is a fixed probability for a row to be selected at each iteration throughout the duration of the algorithm. One possible explanation for why sampling without replacement shows empirical improvement over sampling with replacement could stem from the problem that selectable set methods aim to solve. During some iterations of a Kaczmarz method, there is no improvement to the iterate because the method is projecting onto an already solved equation yielding no change to the iterate. However, empirically and theoretically, the selectable set method does not account for the improvement seen when sampling without replacement. This suggests that there is more going on in the interaction between projections than simply being (near) orthogonal or sharing a solution space.

The Recht-Ré conjecture [RR12] gave a theoretical explanation of why sampling without replacement outperforms sampling with replacement. Recent works [LL20, IKW16] have proved that the Recht-Ré conjecture does hold for smaller dimensions, giving a theoretical explanation of why sampling without replacement outperforms sampling with replacement. However, recently Lai et al. [LL20] were able to prove that for dimensions five or larger this conjecture is false. Thus the theory gap of proving why sampling without replacement outperforms sampling with replacement in Kaczmarz methods is currently an open problem, that while seemingly related to selectable sets, is still not explained.

CHAPTER 3

Online Signal Recovery via Heavy Ball Kaczmarz

This chapter is an adaptation of [JYN22] which is joint work with Ben Jarman and Professor Needell. Ben Jarman initially proposed the idea. Ben Jarman and I contributed the theory, experiments and writing under the supervision of Professor Needell.

We propose a new variant of the Kaczmarz method to online signal recovery setting. Our proposed method, leverages an additional heavy ball momentum term. This term allows for faster convergence when the source distribution which we sample from is highly coherent, inner product between different linear measurements is large. We prove its convergence and experimentally analyze its performance on both synthetic and real world datasets.

3.1 Introduction

3.1.1 The Kaczmarz Method

Recovering a signal $x^* \in \mathbb{R}^n$ from a collection of linear measurements is an important problem in computerized tomography [Nat01], sensor networks [SHS01], compressive sensing [EK12, FR13], machine learning subroutines [Bot10], and beyond. When the collection of linear measurements is finite, say of size m, and accessible at any time, the problem is equivalent to solving a system of linear equations $Ax = b$ with $A \in \mathbb{R}^{m\times n}$ and $b \in \mathbb{R}^m$, which has been well-studied. A popular method for solving this classical problem is the Kaczmarz method [Kar37]: beginning with an initial iterate x_0, at each iteration a row of the system is sampled and the previous iterate is projected onto the hyperplane defined by the solution space given

by that row. More precisely, if the row $a_i^\top x = b_i$ is sampled at iteration k, the update has the form

$$x_k = x_{k-1} - \frac{\langle a_i, x_{k-1}\rangle - b_i}{\|a_i\|^2} a_i.$$

The original method proposed cycling through rows in order, such that $i = k \bmod m$. In [HM93] it was observed empirically that randomized row selection accelerates convergence, and in the landmark work [SV09] it was proven that selecting rows at random with probability proportional to their Euclidean norm yields linear convergence in expectation.

In this work, we consider an online model in which at each discrete time $t = 1, 2, \ldots$ a linear measurement $(\varphi_t, y_t) \in \mathbb{R}^n \times \mathbb{R}$ is received. We assume that each measurement is noiseless, i.e. $\langle \varphi_t, x^* \rangle = y_t$ for all t, and that measurements are *streamed* through memory and are not stored. Note that the linear system setting described above is a special case of this model, but we now allow for measurements to be sampled from a more general source. The Kaczmarz method is well-suited to this setting as it requires access to only a single measurement at each iteration. See, for example, [CP12], where measurement data is viewed as being sampled i.i.d. from some distribution $\mathcal{D}$ on $\mathbb{R}^n$. We assume the noiseless, i.i.d. setting throughout this chapter. A Kaczmarz update in this setting has the following form, when initialized with some arbitrary x_0: at discrete times $t = 1, 2, \ldots,$ a measurement $(\varphi_t, y_t) \in \mathbb{R}^n \times \mathbb{R}$ is received, where $\varphi_t \sim \mathcal{D}$, and a Kaczmarz iteration is computed

$$x_t = x_{t-1} - \frac{\langle \varphi_t, x_{t-1}\rangle - y_t}{\|\varphi_t\|^2} \varphi_t.$$

In [CP12] it was shown that under certain conditions on $\mathcal{D}$, the method enjoys linear convergence in expectation. Further related works have placed online Kaczmarz in the context of learning theory [LZ15], and have analyzed sparse online variants [LZ18, LWS14]. Random vector models have also appeared in analyses of Kaczmarz methods for phase retrieval [TV18] and for sparsely corrupted data [HNR22].

3.1.2 Heavy Ball Momentum

Heavy ball momentum is a popular addition to gradient descent methods, in which an additional step is taken in the direction of the previous iteration's movement. Proposed initially in [Pol64], it has proven very popular in machine learning [SMD13, KSH17, GLZ19], with a guarantee of linear convergence for stochastic gradient methods with heavy ball momentum proven in [LR20] (improving on earlier sublinear guarantees in [YLL16, GPS18]). A gradient descent method itself [NSW16], the Kaczmarz method may be modified with heavy ball momentum to give updates of the following form:

$$x_{t+1} = x_t - \frac{\langle \varphi_t, x_t \rangle - y_t}{\|\varphi_t\|^2} \varphi_t + \beta(x_t - x_{t-1}),$$

where $\beta \geq 0$ is a momentum parameter. In [LR20] it was shown that when applied to a linear system (i.e., when each φ_t is sampled from the rows of a matrix A), the Kaczmarz method with heavy ball momentum converges linearly in expectation. Experimental results indicate accelerated convergence compared to the standard Kaczmarz method on a range of datasets, while the momentum term does not affect the order of the computational cost.

In this work, we propose an *online* variant of the Kaczmarz method with heavy ball momentum. We prove that our method converges linearly in expectation for a wide range of distributions $\mathcal{D}$, and offer particular examples. This theory is supported by numerical experiments on both synthetic and real-world data, which in particular demonstrate the benefits of adding momentum when measurements are highly coherent.

3.2 Proposed Method & Empirical Results

We propose an online variant of the Kaczmarz method, modified to include a heavy ball momentum term $\beta \in (0, 1)$, which we call OHBK(β) (see Algorithm 4). We note that our method is a generalization of the momentum Kaczmarz method for systems of linear equations introduced in [LR20]. The method requires only a single measurement to be held in storage at a time, while leveraging information about previous measurements through the

momentum term.

Algorithm 4: Online Heavy Ball Kaczmarz β, OHBK(β)

Input initial iterate x_0, measurements $\{(\varphi_t, y_t)\}_{t=1}^{\infty}$, momentum parameter β

$x_1 = x_0$

for $t = 1, 2, \ldots$ **do**

  Update $x_{t+1} = x_t - \frac{\langle \varphi_t, x_t \rangle - y_t}{\|\varphi_t\|^2} \varphi_t + \beta(x_t - x_{t-1})$

end

We test our method on synthetic and real-world data. For each data source, we compare our method OHBK(β) for a variety of β to an online Kaczmarz method without momentum, which we denote by OK (equivalently, OHBK(0)).

We first experiment on synthetic data. We sample $x^* \in \mathbb{R}^{50}$ with standard Gaussian entries, and take $\{\varphi_t\}_{t=1}^{\infty}$ to be vectors of length 50 with $U[0, 1]$ entries. We note that this process produces particularly coherent data, that is, the vectors $\{\varphi_t\}_{t=1}^{\infty}$ have small pairwise inner products. Each y_t is then computed as $y_t = \langle \varphi_t, x^* \rangle$ to ensure measurements are noiseless. In Figure 3.2 we perform a parameter search over 100 trials for β and plot the median error after 100 iterations versus β with shading for the 25th through 75th percentiles. Introducing some amount of momentum provides an acceleration, however, taking β to be too large places too much weight on previous information and is less effective. In Figure 3.1 we show convergence down to machine epsilon of OHBK(β) versus online randomized Kaczmarz (i.e. OHBK(0)) for a selection of β (averaging over 10 trials), and the acceleration provided by momentum is clear.

In Figure 3.3, we investigate the effect of momentum on highly coherent systems further. We perform 4000 iterations of OHBK(β) on $U[\varepsilon, 1]$ signals of length 50, for $\varepsilon \in [0, 1]$, for a range of momentum parameters β (again averaged over 10 trials). We see that momentum provides a significant speedup in convergence even for highly coherent systems (i.e. for large ϵ). However, as $\epsilon \to 1$, recovering the signal becomes intractable.

We compare the effect of the signal length n on the optimal momentum parameter β in Figure 3.4. We perform parameter searches for signals of length $n \in \{50, 100, 500, 1000\}$ and mark the optimal values of β. The optimal choice of β does not appear to vary significantly with n.

In Figure 3.5 we use a system generated from the Wisconsin Diagnostic Breast Cancer (WDBC) dataset, where each measurement is computed from a digitized image of a fine needle aspirate of a breast mass and describes characteristics of the cell nuclei present [WSM95]. We stream through each measurement of the 699-row, 10-feature dataset once to replicate the online model, and again see that the addition of momentum provides a noteworthy acceleration to convergence.

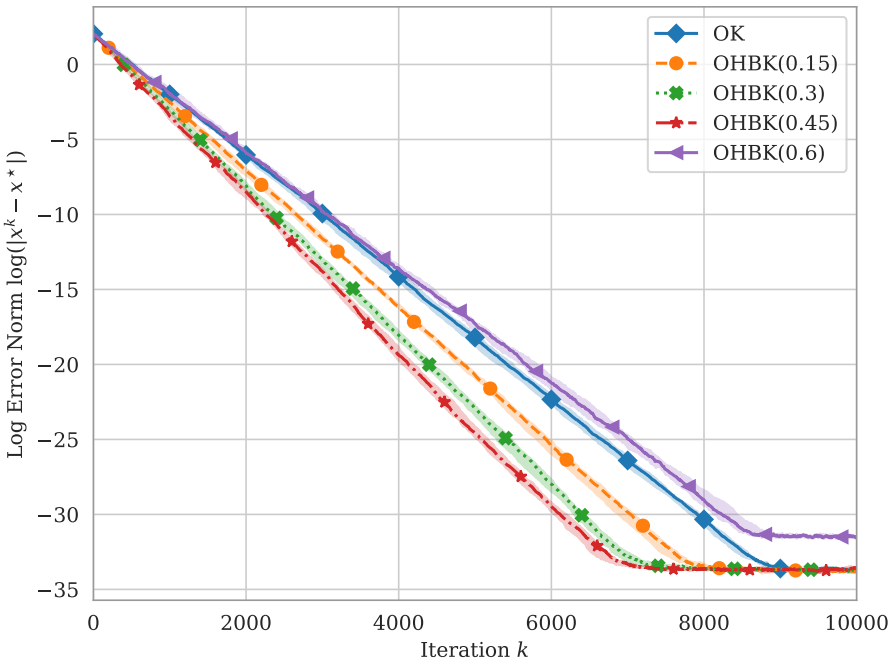


Figure 3.1: Error versus iteration for OHBK(β) applied to $U[0, 1]$ signals of length 50.

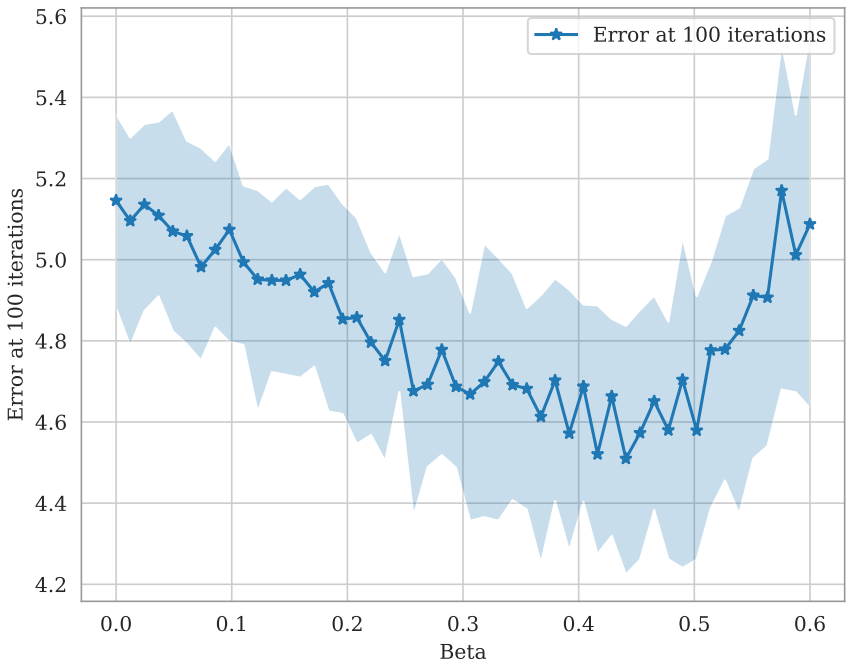


Figure 3.2: $\|x_{100} - x^*\|$ versus β for a range of $\beta \in [0, 0.6]$, for $U[0, 1]$ signals of length 50.

3.3 Theoretical Results

Throughout our theory, we assume that $\{\varphi_t\}_{t=1}^{\infty}$ is a sequence of independent samples from some distribution $\mathcal{D}$. We provide a general linear convergence (in expectation) result with a rate depending on the matrix $W := \mathbb{E}_{\mathcal{D}}\left[\frac{\varphi\varphi^\top}{\|\varphi\|^2}\right]$, in particular on its smallest and largest singular values $\sigma_{\min}(W)$ and $\sigma_{\max}(W)$.

Theorem 16 (Convergence in Expectation of OHBRK)**.** *Suppose that measurement vectors $\{\varphi_t\}_{t=1}^{\infty}$ are sampled independently from $\mathcal{D}$, and $W = \mathbb{E}_{\mathcal{D}}\left[\frac{\varphi\varphi^\top}{\|\varphi\|^2}\right]$. Then if β is small enough such that*

$$4\beta + 4\beta^2 - (1+\beta)\sigma_{\min}(W) + \beta\sigma_{\max}(W) < 0,$$

the iterates produced by OHBK(β) satisfy the following guarantee: for some $\delta > 0$, $q \in (0, 1)$, we have

$$\mathbb{E}[\|x_t - x^*\|^2] \leq q^t(1+\delta)\|x_0 - x^*\|^2.$$

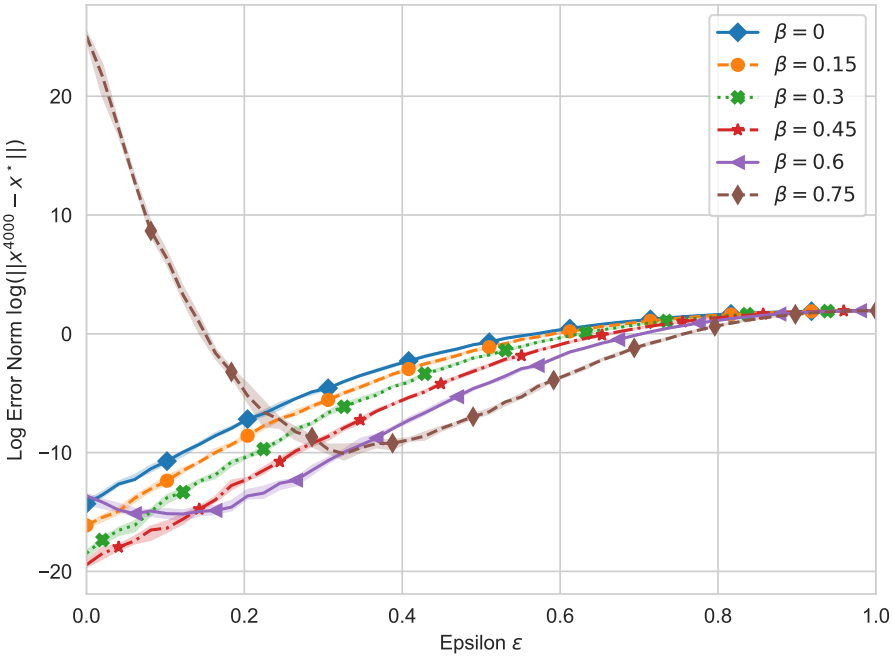


Figure 3.3: $\log \|x_{4000} - x^*\|$ versus ε for OHBK(β) applied to $U[\varepsilon, 1]$ signals of length 50.

More interpretable conditions on β may be obtained for particular classes of distribution $\mathcal{D}$. In particular, if $\varphi / \|\varphi\|$ is distributed uniformly on the unit sphere (which is the case if $\mathcal{D}$ itself is the uniform distribution on the unit sphere, or if $\mathcal{D}$ is the standard n-dimensional Gaussian), then $W = \frac{1}{n} I$ and we require

$$\beta + \beta^2 < \frac{1}{4n}$$

to guarantee linear convergence in expectation.

3.4 Proof of Main Result

In this section we prove Theorem 16 by following the steps of ([LR20], Theorem 1), making modifications for the online case and simplifications to some of the constants for our special case. First we present a lemma from [LR20] which we will use in our convergence proof.

Lemma 17 ([LR20], Lemma 9)**.** *Let $\{F_t\}_{t \geq 0}$ be a sequence of non-negative real numbers*

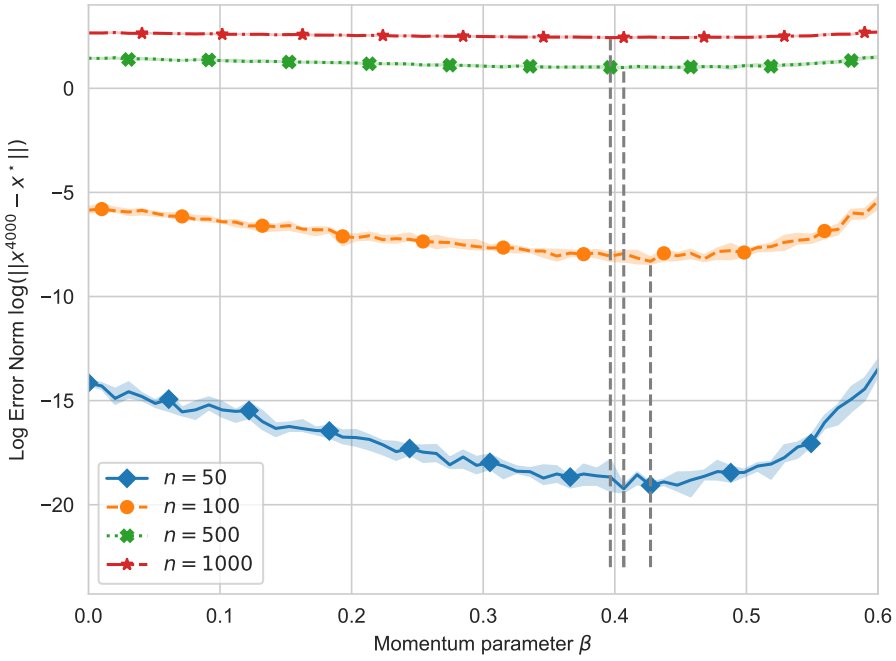


Figure 3.4: $\log \|x_{4000} - x^*\|$ versus β for OHBK(β) applied to $U[0,1]$ signals of length n. The gray verticals show the value of β yielding the minimum error.

with $F_0 = F_1$ *that satisfies the relation* $F_{t+1} \leq a_1 F_t + a_2 F_{t-1}$ *for all* $t \geq 1$, *with* $a_2 > 0$ *and* $a_1 + a_2 < 1$. *Then the following inequality hold for all* $t \geq 1$

$$F_{t+1} \leq q^t (1+\delta) F_0,$$

where $q = \frac{a_1 + \sqrt{a_1^2 + 4a_2}}{2} < 1$, $\delta = q - a_1$ *and* $q \leq a_1 + a_2$.

A proof of this lemma can be found in [LR20].

We begin our convergence analysis by writing the squared $L2$ error at the $(t+1)^{\text{th}}$ iteration and substituting the OHBK(β) update into it,

$$\|x_{t+1} - x^*\|^2 = \left\| x_t - \frac{\langle \varphi_t, x_t \rangle - y_t}{\|\varphi_t\|^2} \varphi_t + \beta(x_t - x_{t-1}) - x^* \right\|^2.$$

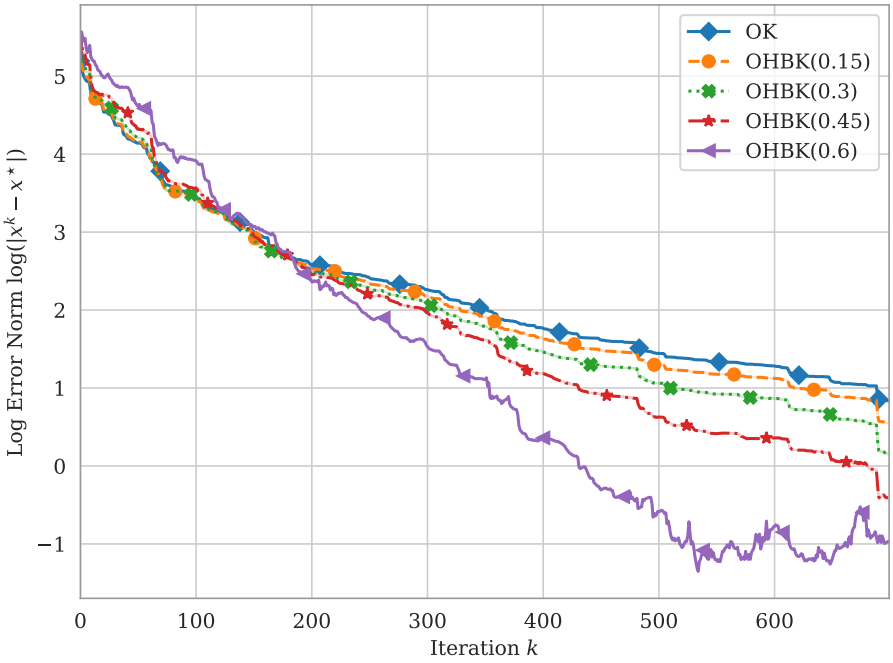


Figure 3.5: Error versus iteration for OHBK(β) applied to the WDBC dataset.

Next, we group our equation into three terms:

$$\begin{aligned}\|x_{t+1} - x^*\|^2 = &\left\|x_t - x^* - \frac{\langle \varphi_t, x_t\rangle - y_t}{\|\varphi_t\|^2}\varphi_t\right\|^2 \\ &+\beta^2 \|x_t - x_{t-1}\|^2 \\ &+2\beta\langle x_t - x^* - \frac{\langle \varphi_t, x_t\rangle - y_t}{\|\varphi_t\|^2}\varphi_t, x_t - x_{t-1}\rangle\end{aligned} \tag{3.1}$$

We bound the first term of Equation (3.1) by following a standard Kaczmarz convergence argument and the fact that $y_t = \langle \varphi_t, x^*\rangle$. We have that

$$\begin{aligned}&\left\|x_t - x^* - \frac{\langle \varphi_t, x_t\rangle - y_t}{\|\varphi_t\|^2}\varphi_t\right\|^2 \\ &= \|x_t - x^*\|^2 + \left\|\frac{\langle \varphi_t, x_t\rangle - y_t}{\|\varphi_t\|^2}\varphi_t\right\|^2 - \\ &\quad 2\left\langle \frac{\langle \varphi_t, x_t\rangle - y_t}{\|\varphi_t\|^2}\varphi_t, x_t - x^*\right\rangle\end{aligned}$$

$$= \|x_t - x^*\|^2 + \frac{(\langle \varphi_t, x_t \rangle - y_t)^2}{\|\varphi_t\|^2} - 2\frac{(\langle \varphi_t, x_t \rangle - y_t)^2}{\|\varphi_t\|^2}$$
$$= \|x_t - x^*\|^2 - \frac{(\langle \varphi_t, x_t \rangle - y_t)^2}{\|\varphi_t\|^2}.$$

We bound the second term of Equation (3.1) by first adding and subtracting x^*

$$\beta^2 \|x_t - x_{t-1}\|^2 = \beta^2 \|(x_t - x^*) + (x^* - x_{t-1})\|^2 .$$

Then by applying the fact that $\|a+b\|^2 \leq 2\|a\|^2 + 2\|b\|^2$ we have that

$$\beta^2 \|(x_t - x^*) + (x^* - x_{t-1})\|^2$$
$$\leq 2\beta^2 \|x_t - x^*\|^2 + 2\beta^2 \|x_{t-1} - x^*\| .$$

Thus we have that

$$\beta^2 \|x_t - x_{t-1}\|^2 \leq 2\beta^2 \|x_t - x^*\|^2 + 2\beta^2 \|x_{t-1} - x^*\| .$$

Finally we bound the third term of Equation (3.1) as

$$2\beta \left\langle x_t - x^* - \frac{\langle \varphi_t, x_t \rangle - y_t}{\|\varphi_t\|^2}\varphi_t, x_t - x_{t-1} \right\rangle =$$
$$2\beta \langle x_t - x^*, x_t - x_{t-1} \rangle +$$
$$2\beta \left\langle \frac{\langle \varphi_t, x_t \rangle - y_t}{\|\varphi_t\|^2}\varphi_t, x_{t-1} - x_t \right\rangle$$

$$= 2\beta \|x_t - x^*\|^2 + 2\beta \langle x_t - x^*, x^* - x_{t-1} \rangle +$$
$$2\beta \left\langle \frac{\langle \varphi_t, x_t \rangle - y_t}{\|\varphi_t\|^2}\varphi_t, x_{t-1} - x_t \right\rangle$$
$$= \beta \|x_t - x^*\|^2 + \beta \|x_t - x_{t-1}\|^2 - \beta \|x_{t-1} - x^*\|^2 +$$
$$2\beta \left\langle \frac{\langle \varphi_t, x_t \rangle - y_t}{\|\varphi_t\|^2}\varphi_t, x_{t-1} - x_t \right\rangle$$
$$\leq \beta \|x_t - x^*\|^2 + \beta \|x_t - x_{t-1}\|^2 - \beta \|x_{t-1} - x^*\|^2 -$$
$$\beta \langle \frac{\langle \varphi_t, x_t \rangle - y_t}{\|\varphi_t\|^2}\varphi_t, x_t - x^* \rangle +$$
$$\beta \langle \frac{\langle \varphi_t, x_{t-1} \rangle - y_t}{\|\varphi_t\|^2}\varphi_t, x_{t-1} - x^* \rangle .$$

Combining the three bounds, we have

$$\begin{aligned}\|x_{t+1}-x^*\|^2 \leq \|x_t-x^*\|^2 - \frac{(\langle\varphi_t,x_t\rangle - y_t)^2}{\|\varphi_t\|^2}\\ +2\beta^2\|x_t-x^*\|^2 + 2\beta^2\|x_{t-1}-x^*\|^2\\ +\beta\|x_t-x^*\|^2 + \beta\|x_t-x_{t-1}\|^2 - \beta\|x_{t-1}-x^*\|^2 -\\ \beta\langle\frac{\langle\varphi_t,x_t\rangle - y_t}{\|\varphi_t\|^2}\varphi_t, x_t-x^*\rangle +\\ \beta\langle\frac{\langle\varphi_t,x_{t-1}\rangle - y_t}{\|\varphi_t\|^2}\varphi_t, x_{t-1}-x^*\rangle.\end{aligned}$$

Simplifying and grouping like terms we have

$$\begin{aligned}\|x_{t+1}-x^*\|^2 \leq (1+2\beta^2+\beta)\|x_t-x^*\|^2 +\\ (2\beta^2-\beta)\|x_{t-1}-x^*\|^2 -\\ \frac{(\langle\varphi_t,x_t\rangle - y_t)^2}{\|\varphi_t\|^2} + \beta\|x_t-x_{t-1}\|^2 -\\ \beta\langle\frac{\langle\varphi_t,x_t\rangle - y_t}{\|\varphi_t\|^2}\varphi_t, x_t-x^*\rangle +\\ \beta\langle\frac{\langle\varphi_t,x_{t-1}\rangle - y_t}{\|\varphi_t\|^2}\varphi_t, x_{t-1}-x^*\rangle.\end{aligned}$$

Applying the simplification for the second term of Equation (3.1) and simplifying the inner products, we have

$$\begin{aligned}\|x_{t+1}-x^*\|^2 \leq (1+2\beta^2+3\beta)\|x_t-x^*\|^2 +\\ (2\beta^2+\beta)\|x_{t-1}-x^*\|^2 -\\ (\beta+1)\frac{\langle\varphi_t,x_t-x^*\rangle^2}{\|\varphi_t\|^2} +\\ \beta\frac{\langle\varphi_t,x_{t-1}-x^*\rangle^2}{\|\varphi_t\|^2}.\end{aligned}$$

Taking an expectation over our signal of our simplified equation

$$
\begin{aligned}
\mathbb{E}[\|x_{t+1} - x^*\|^2] \leq (1 + 2\beta^2 + 3\beta) \|x_t - x^*\|^2 + \\
(2\beta^2 + \beta) \|x_{t-1} - x^*\|^2 - \\
(\beta + 1)\mathbb{E}[\frac{(\langle \varphi_t, x_t - x^* \rangle)^2}{\|\varphi_t\|^2}] + \\
\beta\mathbb{E}[\frac{(\langle \varphi_t, x_{t-1} - x^* \rangle)^2}{\|\varphi_t\|^2}] \\
= (1 + 2\beta^2 + 3\beta) \|x_t - x^*\|^2 + \\
(2\beta^2 + \beta) \|x_{t-1} - x^*\|^2 - \\
(1 + \beta)(x_t - x^*)^T \mathbb{E}\left[\frac{\varphi_t \varphi_t^T}{\|\varphi_t\|^2}\right] (x_t - x^*) + \\
\beta(x_{t-1} - x^*)^T \mathbb{E}\left[\frac{\varphi_t \varphi_t^T}{\|\varphi_t\|^2}\right] (x_{t-1} - x^*).
\end{aligned}
$$

Let $W := \mathbb{E}\left[\frac{\varphi_t \varphi_t^T}{\|\varphi_t\|^2}\right]$. We can then bound the above in terms of the largest and smallest singular values of W:

$$
\begin{aligned}
\mathbb{E}[\|x_{t+1} - x^*\|^2] \leq (1 + 2\beta^2 + 3\beta) \|x_t - x^*\|^2 + \\
(2\beta^2 + \beta) \|x_{t-1} - x^*\|^2 - \\
(1 + \beta)\sigma_{\min}(W) \|x_t - x^*\|^2 + \\
\beta\sigma_{\max}(W) \|x_{t-1} - x^*\|^2 \\
= (1 + 2\beta^2 + 3\beta - (1 + \beta)\sigma_{\min}(W)) \|x_t - x^*\|^2 + \\
(2\beta^2 + \beta + \beta\sigma_{\max}(W)) \|x_{t-1} - x^*\|^2 .
\end{aligned}
$$

Finally, we apply Lemma 17, wherein the two coefficients are given by $a_1 = 1+2\beta^2+3\beta-(1+\beta)\sigma_{\min}(W)$ and $a_2 = 2\beta^2 + \beta + \beta\sigma_{\max}(W)$. Since we assumed that $a_1 + a_2 = 1 + 4\beta^2 + 4\beta + (1 + \beta)\sigma_{\min}(W) + \beta\sigma_{\max}(W) < 1$ and since $\beta > 0$ then $a_2 = 2\beta^2 + \beta + \beta\sigma_{\max}(W) > 0$ thus the assumptions for Lemma 17 hold, so we have that

$$
\mathbb{E}[\|x_t - x^*\|^2] \leq q^t(1 + \delta) \|x_0 - x^*\|^2
$$

where $q = \frac{a_1+\sqrt{a_1^2+4a_1}}{2}$, $\delta = q - a_1$ and $a_1 + a_2 \leq q < 1$. Since $q \in (0, 1)$ we have shown that the norm squared error of the iterates produced by OHBK(β) converges linearly in expectation.

3.5 Conclusion and Future Directions

In this work we discuss using a Kaczmarz method variant with momentum to solve an online signal recovery problem. We leverage a heavy ball momentum term, a classical acceleration method, to improve the convergence rate. We prove a theoretical convergence rate for OHBK(β), and verify this convergence empirically on both synthetic and real-world data. We demonstrate empirically that for coherent measurements, the addition of momentum indeed accelerates convergence, and provided some initial exploration into the dependence of the convergence rate on the signal length n and momentum strength β.

It is notable that in our convergence analysis, we did not recover a theoretically optimal value for β. Doing so, and comparing this value to empirically best values, would be an interesting future direction. Furthermore, we would like to obtain theoretical parameter relationships: for example, how the optimal momentum strength depends on the signal length and coherency of the measurements. It may in fact be optimal to adaptively adjust the momentum parameter across iterations based on the current iterate and properties of incoming measurements. Additionally, we would like to leverage other accelerated gradient methods such as ADAM [KB14]. Finally, we would like to consider solving the online signal recovery problem in the case where each measurement is no longer exact, but instead contains some amount of noise [Nee10]. This could be achieved, for example, using relaxation.

CHAPTER 4

Multi-Randomized Kaczmarz for Latent Class Regression

This chapter is a version of [GYN22] which is joint work with Erin George and Deanna Needell. Erin and I are joint first authors of this work. The aim of this work is to solve a latent class linear regression problem which we pose as solving multiple linear systems at the same time. We assume that the systems have been scrambled together and we do not know how many and which rows belong to which system.

The goal of posing this setting and proposing a Kaczmarz method based solution was to model a scenario in which multiple underlying groups exist within our data and each group may require a different regressor, solution vector. If we solve the joint system for a single solution, this solution is sub-optimal for some of the groups whereas if we assume multiple solutions, they yield a better results for each group. An example application of this could be dosage rates for medicine depending age. If we regress on all of the data, we may have sub-optimal doses for everyone but if we divide the data into older populations and younger populations we may see that older people require larger doses due to building up a resistance throughout their lives while younger people require smaller doses for more effective treatments.

Since Kaczmarz allows us to iteratively solve linear systems one row at a time we are able to develop an algorithm to solve the latent class regression problem. We sample a single row, decide which solution that row belongs by explicitly biasing to the closer solution then project onto it.

In the remainder of this chapter we propose our novel Kaczmarz based method for solving the latent class linear regression problem. We prove its convergence under mild assumptions and experimentally demonstrate its performance on both synthetic and real world datasets.

4.1 Introduction

Often, one needs to perform regression tasks on extremely large-scale data. Methods such as the randomized Kaczmarz method (RK) [Kar37, SV09] have gained recent attention for their ability to solve such systems with needing to only access a single row at a time rather than the full system in memory. However, in many settings, two or more population subgroups may be present in the data requiring multiple regressors. Often times, computing a single regressor will result in a minority group having far worse predictive power than the majority. Additionally, the minority group is not known a priori requiring that we both discover and regress on these subgroups on the fly. Here, we present a variant of RK that addresses this problem via multiple regressors.

Formally, given multiple consistent systems of equations $M^{(i)}x_*^{(i)} = b^{(i)}$, $i \in \{0, 1, \ldots, n\}$ we consider the combined matrices

$$M = \begin{bmatrix} M^{(0)} \\ M^{(1)} \\ \vdots \\ M^{(n)} \end{bmatrix} \quad b = \begin{bmatrix} b^{(0)} \\ b^{(1)} \\ \vdots \\ b^{(n)} \end{bmatrix}$$

with the goal of recovering $x_*^{(0)}, x_*^{(1)}, \ldots, x_*^{(n)}$ where the rows of these matrices may be shuffled. Next we define the class of a set of rows and right hand side entries.

Definition 2 (Class)**.** *Given a regressor $x_*^{(i)}$ a set of rows resulting in matrix $M^{(j)}$ and right hand sides $b^{(j)}$ are in class i if $M^{(j)}x_*^{(i)} = b^{(j)}$.*

We assume that the class of each row is not known beforehand. This task corresponds with uncovering multiple systems and their solutions. In the statistics literature, this problem

can be framed as latent class linear regression where each class represents an overdetermined system of equations [WD94, MV04]. Classically, this problem can be solved by using an expectation-maximization (EM) algorithm to iteratively fit the regressor coefficients and then classify the rows [Moo96, KYB19]. The EM algorithm has been extensively studied in a statistics framework with convergence properties discussed in [Wu83] and [DLR77]. More recently the EM framework has been used to learn class data representations in unsupervised machine learning using neural networks [GVS17].

We take a randomized numerical linear algebra approach to this problem by modifying the classical randomized Kaczmarz algorithm to this setting. This approach allows us to process very large data sets while only accessing single rows of our data set at a time.

4.2 Multi-Randomized Kaczmarz Method

We propose a novel iterative method motivated by the randomized Kaczmarz (RK) algorithm for simultaneously solving all $n+1$ systems, Algorithm 5. This approach is motivated by the assumption that the closer an iterate is to a hyperplane defined by a row of the combined system, the more likely that row belongs to the class of that iterate. Since Kaczmarz methods converge monotonically this is a reasonable assumption.

At each iteration, the multi-randomized Kaczmarz (MRK) method selects a hyperplane as in the standard RK algorithm. Then the Kaczmarz update for all iterates is computed. The update with the smallest magnitude is selected, denoted s_k, with the respective magnitude denoted as c_{s_k}. Given a swap probability r we then update iterate t_k chosen to be s_k with probability $1-r$ and t_k chosen from all iterates uniformly at random with total probability r. The selected iterate t_k is updated by the magnitude c_{s_k} in the direction the t_k-th iterate would have been updated given the standard Kaczmarz update.

We state two convergence results for this method, which we will prove in the following section. The first theorem, Theorem 18, proves a linear convergence result for the MRK

Algorithm 5: Multi-Randomized Kaczmarz (MRK) Algorithm

Input: System M, right hand side b, number of iterations N, initial iterates $x_0^{(0)}$, $x_0^{(1)}, \ldots, x_0^{(n)}$, swap probability r, sampling distribution $\mathcal{D}$.

for $k = 0, 1, \ldots, N-1$ **do**

  Sample row $i_k \sim \mathcal{D}$

  $c_{i,k} = \frac{M_{i_k} x_k^{(i)} - b_{i_k}}{||M_{i_k}||^2}$, $i = 0, 1, \ldots, n$

  $s_k = \text{argmin}_{i \in \{0,1,\ldots,n\}}(|c_{i,k}|)$

  $$t_k = \begin{cases} s_k & \text{with probability } 1 - r \\ t & \text{with probability } \frac{r}{n+1} \text{ for all } t \in \{0, \ldots, n\} \end{cases}$$

  ▷ The total probability that $t_k = s_k$ is $1 - r + \frac{r}{n+1}$.

  $x_{k+1}^{(t_k)} = x_k^{(t_k)} - |c_{s_k}| \operatorname{sgn}(c_{t_k}) M_{i_k}^T$

  $x_{k+1}^{(j)} = x_k^{(j)}, j \neq t_k$

end

algorithm in expectation under certain conditions. The second theorem, Theorem 19, is an almost sure convergence result for the MRK algorithm. Other almost sure convergence results have been shown for Kaczmarz type algorithms [CP12] under the assumption that measurements (rows of the matrix) are drawn from independent but not necessarily identical distributions.

To prove these theorems, we will make a uniqueness assumption on the problem.

Assumption 1. *The solution to the set of systems is unique up to relabeling. That is, suppose there are x_i, $i \in \{0, 1, \ldots, n\}$ so that for each row in the combined system (indexed by k) there is i_k where*

$$M_k x_{i_k} = b_k.$$

Then there is a permutation σ on $\{0, 1, \ldots, n\}$ so that $x_*^{(i)} = x_{\sigma(i)}$ for all i.

In particular, this means all systems in the problem are full rank, even if rows which

consistently belong to two or more classes are removed.

Theorem 18 (Conditional expected MRK Convergence)**.** *Define*

$$e_k = \sum_{i=0}^{n} \left\| x_k^{(i)} - x_*^{(i)} \right\|^2 .$$

Let $r \geq 0$ be sufficiently small. Choose $c \in (C_0, 1)$ and $\delta > 0$. There exists $\varepsilon > 0$ so that if $e_k < \varepsilon$ then

$$\mathbb{E}(e_{k+b} | A_\delta) \leq c^b e_k$$

where A_δ is an event that happens with probability at least $1 - \delta$. The constant $C_0 < 1$ depends on M, b, n, and r.

This theorem shows the convergence will be linear in expected squared error after a certain point. Limiting the initial squared error before convergence allows us to identify which solution each iterate is converging towards. The failure probability reflects the possibility that the iterates may still converge towards a different labeling of the solutions and iterates. In the case where the initial squared error is too large or the failure probability is triggered, we will still see convergence, as shown in the next theorem.

Theorem 19 (Almost sure MRK Convergence)**.** *There is $r' \in (0, 1)$ so that if $r \in (0, r')$, each iterate of the algorithm converges almost surely to a different solution of the subsystems.*

The convergence rate given by the proof of this theorem is slow. In practice, we find the convergence rate quickly achieves the linear rate given in the previous theorem.

4.3 Proofs

4.3.1 Conditional Convergence in Expectation

Proof of Theorem 18. At the k-th iteration, we select a row from a system. Suppose we select a row ℓ_k from system i. There are three possibilities for how we update.

(a) We update $x_k^{(i)}$ fully, setting

$$\left\|x_{k+1}^{(i)} - x_*^{(i)}\right\|^2 = C_k^{(i)} \left\|x_k^{(i)} - x_*^{(i)}\right\|^2$$

for some random variable $C_k^{(i)}$ taking value in the range $[0, 1]$. The expectation of $C_k^{(i)}$ is just the Kaczmarz constant for the subset of rows which we are allowed to make a full and correct update with.

(b) We update $x_k^{(i)}$ partially. We bound the error here as

$$\left\|x_{k+1}^{(i)} - x_*^{(i)}\right\| \leq \left\|x_k^{(i)} - x_*^{(i)}\right\|.$$

(c) We update $x_k^{(j)}$ for some $j \neq i$. Regardless of how this happens, we always update by a magnitude bounded above in norm by the correct update:

$$\frac{\left|M_{\ell_k} x_k^{(i)} - b_{\ell_k}\right|}{\|M_{\ell_k}\|} \leq \left\|x_k^{(i)} - x_*^{(i)}\right\|.$$

Therefore the new error satisfies

$$\left\|x_{k+1}^{(j)} - x_*^{(j)}\right\| \leq \left\|x_k^{(j)} - x_*^{(j)}\right\| + \left\|x_k^{(i)} - x_*^{(i)}\right\|$$

and by Cauchy-Schwarz and Young's inequality

$$\left\|x_{k+1}^{(j)} - x_*^{(j)}\right\|^2 \leq 2\left\|x_k^{(j)} - x_*^{(j)}\right\|^2 + 2\left\|x_k^{(i)} - x_*^{(i)}\right\|^2.$$

There are two ways for us to land in case (c). Either we trigger our swap probability and select iterate j, or we do not trigger our swap probability but we selected iterate j anyway. The second happens only when

$$\frac{\left|M_{\ell_k} x_k^{(j)} - b_{\ell_k}\right|}{\|M_{\ell_k}\|} \leq \frac{\left|M_{\ell_k} x_k^{(i)} - b_{\ell_k}\right|}{\|M_{\ell_k}\|}$$

We can bound the left side below by

$$\frac{|M_{\ell_k}(x_*^{(i)} - x_*^{(j)})|}{\|M_{\ell_k}\|} - \left\|x_k^{(j)} - x_*^{(j)}\right\|$$

and the right hand side above by

$$\left\| x_k^{(i)} - x_*^{(i)} \right\|.$$

So this can only happen when

$$\begin{aligned}
\frac{|M_{\ell_k}(x_*^{(i)} - x_*^{(j)})|}{\|M_{\ell_k}\|} &\leq \left\| x_k^{(i)} - x_*^{(i)} \right\| + \left\| x_k^{(j)} - x_*^{(j)} \right\| \\
&\leq \sum_{a=0}^{n} \left\| x_k^{(a)} - x_*^{(a)} \right\| \\
&\leq \sqrt{(n+1) \cdot e_k}.
\end{aligned}$$

We only need to consider the case when $M_{\ell_k}(x_*^{(i)} - x_*^{(j)}) \neq 0$, as otherwise we could consider this row ℓ_k as coming from the j-th system anyway. Therefore, the probability of this happening goes to 0 as e_k goes to 0. Suppose ε is small enough so that whenever $e_k < \varepsilon$ the probability this mistake happens for any pair is less than q.

We will also assume that ε is small enough so that, assuming we do not trigger our swap probability, there is a full rank set of rows for each system so that whenever $e_k < \varepsilon$ all these rows will make a correct update and that cannot be added to another system consistently. This is a consequence of Assumption 1. Then, for each system, the condition number for the set of rows that can make a correct update is bounded above, and the RK constant is bounded above by some value strictly less than 1. Let $c < 1$ bound above the RK constant for each system.

Now, whenever $e_k < \varepsilon$, we can bound

$$\begin{pmatrix} \left\| x_{k+1}^{(0)} - x_*^{(0)} \right\|^2 \\ \vdots \\ \left\| x_{k+1}^{(n)} - x_*^{(n)} \right\|^2 \end{pmatrix} \leq \mathbf{A} \begin{pmatrix} \left\| x_k^{(0)} - x_*^{(0)} \right\|^2 \\ \vdots \\ \left\| x_k^{(n)} - x_*^{(n)} \right\|^2 \end{pmatrix}$$

where $\leq$ is interpreted component-wise and

$$\begin{aligned}
\mathbf{A}_{ii} &= 1 + \frac{m_j}{m}(c-1)(1 - q - \tfrac{nr}{n+1}) + \frac{m - m_j}{m}(q + \tfrac{r}{n+1}) \\
\mathbf{A}_{ij} &= 2\frac{m_j}{m}(q + \tfrac{r}{n+1}) \text{ if } i \neq j.
\end{aligned}$$

Here m_j is the number of rows in the j-th system and $m = \sum_j m_j$.

By induction,

$$\begin{pmatrix} \left\| x_{k+b}^{(0)} - x_*^{(0)} \right\|^2 \\ \vdots \\ \left\| x_{k+b}^{(n)} - x_*^{(n)} \right\|^2 \end{pmatrix} \leq \mathbf{A}^b \begin{pmatrix} \left\| x_k^{(0)} - x_*^{(0)} \right\|^2 \\ \vdots \\ \left\| x_k^{(n)} - x_*^{(n)} \right\|^2 \end{pmatrix}$$

for all $b \in \mathbb{N}$ provided that $e_{k+a} < \varepsilon$ for all $a \in \{0, \ldots, b-1\}$. We wish to show the ℓ_1 operator norm of $\mathbf{A}$ is less than 1. This happens when

$$\begin{aligned} &\frac{m_j}{m}(c-1)\left(1 - q - \frac{nr}{n+1}\right) \\ &+ \left(q + \frac{r}{n+1}\right)\left(\frac{m - m_j}{m} + 2n\frac{m_j}{m}\right) \end{aligned}$$

is negative. This occurs when $q + \frac{nr}{n+1}$ is small enough. So then, for r sufficiently small, we can choose ε to make $\|\mathbf{A}\|_{\ell^1 \to \ell^1} = d < 1$.

Suppose $e_k < \delta < \varepsilon$. By our previous bound, $e_{k+b} < d^b\delta$, conditioned on the intermediate values $e_{k+a} < \varepsilon$. By Markov's inequality, the probability that $e_{k+a} \geq \varepsilon$ is at most $\frac{1}{\varepsilon}d^a\delta$. The total probability of this happening is at most $\frac{\delta}{\varepsilon}\frac{d}{1-d}$. Therefore our total error remains bounded above by ε with probability at least $1 - \frac{\delta}{\varepsilon}\frac{d}{1-d}$, and in this case we have convergence in expectation.

□

4.3.2 Convergence with full probability

An outline of the proof for Theorem 19:

1. We use Theorem 18 to define "convergence basins": regions where, if the iterates fall into, there is some positive probability that they never escape and converge in expectation.
2. We show $\left\| x_k^{(i)} - x_*^{(i)} \right\|$ is bounded by some constant independent of i and k.

3. We show we can bound the probability of falling into a basin eventually below by some positive number.

We will begin by proving the following lemma, which will be used in the second part of the outline above.

Lemma 20. *Let R be a sequence of rows from the problem. The sequence of Kaczmarz updates corresponding to R defines an affine transformation $v \mapsto T_R v + v_R$. There are constants $c_r \in (0,1), B_r \in \mathbb{R}_+$ for $r \in \{1, \dots, d\}$ so that $\|v_R\| \leq B_{\dim \operatorname{span} R}$ and $\|T_R\|_R \leq c_{\dim \operatorname{span} R}$, where $\|\cdot\|_R$ is the ℓ^2 operator norm when the operator is restricted to* $\operatorname{span} R$.

Proof of Lemma 20. We proceed by induction on r. The Kaczmarz update for the ℓ-th row is

$$K_\ell : v \mapsto \left(I - \frac{1}{\|M_\ell\|^2} M_\ell^T M_\ell\right) v + \frac{b_\ell}{\|M_\ell\|^2} M_\ell^T$$

If $r = 1$, then T_R is the zero operator restricted to $\operatorname{span} R$. We can take $c_1 = 0$ and $B_1 = \max_\ell \frac{|b_\ell|}{\|M_\ell\|}$.

Now, assume the lemma is true for all $r < r'$. Let $R = (M_{\ell_1}, \dots, M_{\ell_k})$ be a sequence of rows where $\dim \operatorname{span} R = r'$. We can group

$$\begin{aligned} K_{\ell_k} \circ \cdots \circ K_{\ell_1} =& (K_{\ell_k} \circ \cdots \circ K_{\ell_{a_N}}) \\ & \circ (K_{\ell_{a_N - 1}} \circ \cdots \circ K_{\ell_{a_{N-1}}}) \\ & \vdots \\ & \circ (K_{\ell_{a_2 - 1}} \circ \cdots \circ K_{\ell_{a_1}}) \\ & \circ (K_{\ell_{a_1 - 1}} \circ \cdots \circ K_{\ell_{a_0}}) \end{aligned}$$

so that $a_0 = 1$ and a_i for $i \in \{0, \dots, N\}$ is a strictly increasing sequence where the following

are true:

$$\begin{aligned}
\dim \operatorname{span}\{M_{\ell_{a_{i-1}}}, \ldots, M_{\ell_{a_i-1}}\} &= r' && \forall i \in \{1, \ldots, N\} \\
\dim \operatorname{span}\{M_{\ell_{a_{i-1}}}, \ldots, M_{\ell_{a_i-2}}\} &= r' - 1 && \forall i \in \{1, \ldots, N\} \\
\dim \operatorname{span}\{M_{\ell_{a_n}}, \ldots, M_{\ell_k}\} &< r'.
\end{aligned}$$

Consider a grouping $(K_{\ell_{a_i}-1} \circ \cdots \circ K_{\ell_{a_{i-1}}})$, the linear part of the transformation is $A = \left(I - \frac{1}{\left\|M_{\ell_{a_i}}\right\|^2} M_{\ell_{a_i}}^T M_{\ell_{a_i}}\right)$ composed with an operator T that sends $\mathcal{S} = \operatorname{span}\{M_{\ell_{a_{i-1}}}, \ldots, M_{\ell_{a_i-2}}\}$ to itself and has operator norm less than $c_{r'-1}$ on this space. Consider a unit vector $v \in \operatorname{span} R$. We can decompose $v = v_1 + v_2$ with $v_1 \in \mathcal{S}$ and $v_2 \in \mathcal{S}^\perp \cap \operatorname{span} R$. This allows us to bound

$$\|ATv\| \leq \|Tv\| \leq \sqrt{c_{r'-1}^2 \|v_1\|^2 + \|v_2\|^2}$$

using that the operator norm of A is at most 1 and that T is the identity on $\mathcal{S}^\perp$. Another bound is

$$\|ATv\| \leq \|ATv_1\| + \|Av_2\| \leq \|v_1\| + \left(1 - \frac{|M_{\ell_{a_i}} v_2|}{\left\|M_{\ell_{a_i}}\right\|}\right)$$

obtained with the triangle inequality and again T being the identity on $\mathcal{S}^\perp$.

These bounds combine to give a bound for the ℓ^2 operator norm of AT on $\operatorname{span} R$ that depends only on $\mathcal{S}$ and ℓ_{a_i}. There are finitely many possible choices for $\mathcal{S}$ and ℓ_{a_i}, so there is some bound $c' < 1$ on the operator norm of AT independent of what $\mathcal{S}$ and ℓ_{a_i} are. Next we turn to the affine part. By the induction hypothesis, this is a vector with norm at most $B' = B_{r'-1} + \max_\ell \frac{|b_\ell|}{\|M_\ell\|}$.

Next we turn to the last grouping, which is not of this form. We will not analyze the linear part, and note the affine part v' is bounded above in norm by $B'' = \max_{r<r'} B_r$.

Since all linear operators here have operator norm at most 1, a final bound for the operator norm of T_R restricted to $\operatorname{span} R$ is c', which we can take to be $c_{r'}$. We bound

$$\|v_R\| \leq B'' + \left[\sum_{i=1}^{\ell} c_{r'}^{i-1} B'\right] \leq B'' + \frac{1}{1 - c_{r'}} B'.$$

So we can let $B_{r'}$ be this bound. □

Proof of Theorem 19. We first bound the norm of the iterates above by some constant D.

Consider the evolution of a single iterate $x_k^{(i)}$ after finitely many steps. At the last update for $x_k^{(i)}$, we perform a Kaczmarz update with respect to some system, and move the iterate towards, but not past, the update. There is a line segment of possible choices for the next update once we have selected the row. The potential next iterate with largest norm is one of the end points of the line segment, corresponding to either doing a full update or no update at all. So we can either remove this last update or replace it with a full update to yield a final iterate with norm at least as large. We repeat with each update in reverse order, noting the image of a line segment after a series of affine transformations is still a line segment, choosing the one that will yield the largest norm at the end. Hence, to bound the norm of the iterates, we only need to consider sequences where we only ever make full Kaczmarz updates.

Using Lemma 20, we see that if the initial norm of the iterates are bounded above by A, after a sequence of rows R the norm is at most $c_{\dim \operatorname{span} R} A + B_{\dim \operatorname{span} R}$, which is bounded above by $D = A + \max_{r \in \{1,\dots,d\}} B_r$.

As given by Theorem 18, let ε be such that if $e_k \leq \varepsilon$, we converge with some positive probability t.

Let C be the set of all possible iterates such that $\frac{\varepsilon}{n+1} \leq \left\| x_k^{(i)} - x_*^{(i)} \right\|^2 \leq D^2$ for all $i \in \{0, \dots, n\}$ with D given above. This set is compact. If our iterates do not lie in C, then already $e_k < \varepsilon$, so we only need to look at what happens if our iterates lie in C.

Define

$$g(x^{(0)}, \dots, x^{(n)}) = \max_{\ell \in \{1,\dots,m\}} \min_{i \in \{0,\dots,n\}} \frac{|M_\ell x^{(i)} - b_\ell|}{\|M_\ell\|}.$$

This function is the norm of the largest possible iterates in our algorithm if the iterates currently take the values $x^{(0)}, \dots, x^{(n)}$. This is a continuous function on $\mathbb{R}^{d(n+1)}$, so it achieves a minimum c on C. This minimum c is positive, as by our assumption g cannot be zero anywhere on C.

Now, for all value of iterates, we have probability at least $\frac{1}{m}$ of choosing a row where we

will make an update with norm at least c. The probability of updating the correct system with a chosen row is at least $\frac{r}{n+1}$. The squared error of the corresponding iterate decreases by *at least* the norm squared of the update, which is at least c^2, because the resulting triangle between the previous iterate, next iterate, and solution is obtuse. We can keep doing this as long as our iterates remain in C. Therefore, if we make no more than

$$A = (n+1)\left\lceil \frac{N - \frac{\varepsilon}{n+1}}{c^2} \right\rceil$$

of these updates, we have $e_k < \varepsilon$. The probability of this happening is

$$\left(\frac{r}{m(n+1)}\right)^B,$$

where m is the number of rows of M. This is a fixed positive value independent of the iterate. □

4.4 Experimental Results

We test the MRK method on synthetic and real world data to verify the merits of the method. First we construct a problem in two-dimensional space and visualize how our iterates move in space in Figure 4.1. From Figure 4.1 we see how the iterates converge to solutions, moving closer with each projection. The problem is defined by two 10×2 systems where $M_1 \in \mathbb{R}^{10\times 2}$ with entries drawn i.i.d. from $\mathcal{N}(0.8, 0.3)$ and $M_2 \in \mathbb{R}^{10\times 2}$ with entries drawn i.i.d. from $\mathcal{N}(-0.8, 0.3)$.

Next, in Figure 4.2 we use the MRK method on a large synthetic data set. We plot the log norm squared error per iteration of the two iterates for the two class system. The system is defined by two matrices $M_1 \in \mathbb{R}^{1000\times 10}, M_2 \in \mathbb{R}^{1000\times 10}$ where the matrices have entries drawn i.i.d. from $\mathcal{N}(0, 1)$. Each initial iterate $x_0^0, x_0^1 \sim \mathcal{N}(0, 1)$ with zero swap probability. We plot the median and shade the interquartile range in Figure 4.2. We observe that both iterates converge to machine precision and the method succeeds at solving both systems simultaneously.

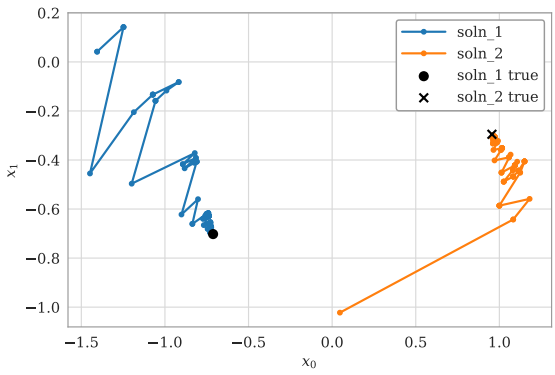


Figure 4.1: Here we plot the evolution of two iterates in the two-dimensional plane. Our system is defined by two matrices $M_1 \in \mathbb{R}^{10\times 2}$ with entries drawn i.i.d. from $\mathcal{N}(0.8, 0.3)$ and $M_2 \in \mathbb{R}^{10\times 2}$ with entries drawn i.i.d. from $\mathcal{N}(-0.8, 0.3)$. Each initial iterate $x_0^0, x_0^1 \sim \mathcal{N}(0, 1)$. We let our swap probability $r = 0$ and sample rows uniformly at random.

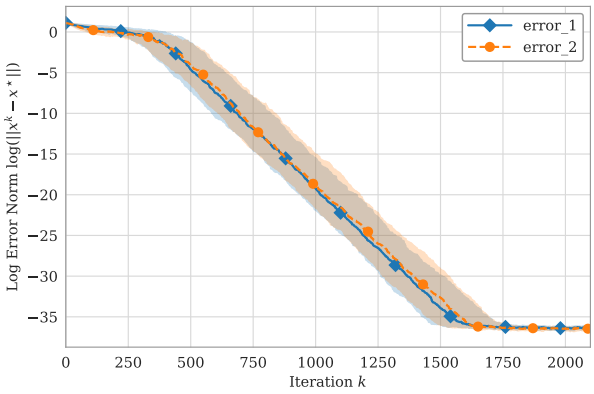


Figure 4.2: System defined by two matrices $M_1 \in \mathbb{R}^{1000\times 10}, M_2 \in \mathbb{R}^{1000\times 10}$ where the matrices have entries distributed $M_1, M_2 \sim \mathcal{N}(0, 1)$. Each initial iterate $x_0^0, x_0^1 \sim \mathcal{N}(0, 1)$. We let our swap probability $r = 0$, sample rows uniformly at random and plot the median and interquartile range over the 100 trials.

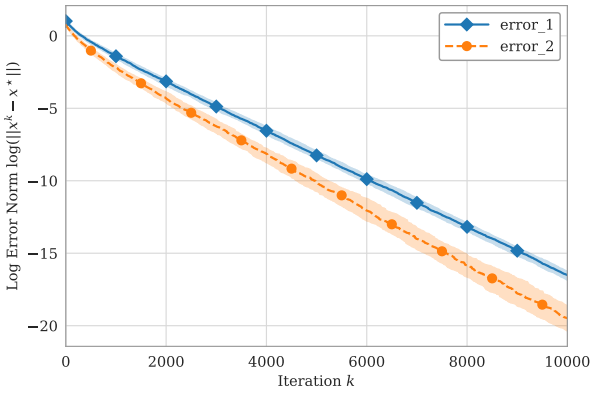


Figure 4.3: System defined by matrices $M_1 \in \mathbb{R}^{300\times 10}, M_2 \in \mathbb{R}^{399\times 10}$ submatrices of the Wisconsin breast cancer data set. Each initial iterate $x_0^0, x_0^1 \sim \mathcal{N}(0,1)$. We let our swap probability $r = 0$, sample rows uniformly at random and plot the median and interquartile range over the 100 trials.

Finally, in Figure 4.3 we define a two problem system for the MRK method using real world data [WSM95]. We construct two systems defined by matrices $M_1 \in \mathbb{R}^{300\times 10}, M_2 \in \mathbb{R}^{399\times 10}$ submatrices of the Wisconsin breast cancer data set. We start two initial iterates with standard normal entries and let our swap probability be zero. We observe that in Figure 4.3 both iterates converge to their respective solutions. We plot the median and interquartile range for the iterates' log error norm per iteration over 100 trials and observe that the method is able to solve this two system problem.

4.5 Conclusion and Future Work

In this chapter we introduce the novel multi-randomized Kaczmarz algorithm, Algorithm 5, to solve the consistent latent class regression problem. We prove linear convergence for the algorithm in expectation with high probability under some constraints in Theorem 18 and almost surely in Theorem 19. Additionally, we observe promising results when applying the algorithm to test data sets. We plan on extending this work to inconsistent and noisy systems by leveraging the extended Kaczmarz method [ZF13], which converges to the least squares

solution [Nee10]. Additionally, we would like to explore using Kaczmarz variants such as max distance [NSL16], sampling Kaczmarz-Motzkin [DHN17] and selectable set [YMS22] methods in this setting. Finally, we are interested in adaptively marking and assigning which rows belong to which system in real time based on the iterate projection values.

CHAPTER 5

Stratified Non-negative Matrix Factorization

This chapter is an adaptation of [CYN23] which is joint work with James Chapman and Professor Needell. James and I are joint first authors on this work. We proposed this idea, implemented the method, proved its convergence and contributed the experimental results under the guidance of Professor Needell.

We aim to factor a set of non-negative matrices each of which contain samples by variables. This factorization is an unsupervised learning method meant to capture the underlying structure within the data. The factorization that we propose is comprised of a local vector for each matrix and a product of two low rank matrices, a local weight matrix and a global dictionary matrix.

In the remainder of this chapter, we first propose an objective function corresponding to a factorization and a multiplicative update algorithm to construct this factorization. Then we prove convergence of the objective function under the multiplicative update rules and experimentally test the performance of the factorization on synthetic, tabular, image and text datasets highlighting the merits of the method.

5.1 Introduction

Non-negative matrix factorization (NMF) is a classical unsupervised machine learning method used in dimensionality reduction and topic modeling [KCP15, WZ12, SBP06]. NMF best addresses data which could be grouped into topics. For example, one could break down a collection of books by topics and then further break down the topics by words associated

to them [GST07, Wal06]. In a more general setting, we refer to books, topics, and words as *samples*, *features*, and *variables*, respectively. This is modeled by

$$\underset{W,H}{\operatorname{argmin}} ||A - WH||_F^2, \; W \geq 0, H \geq 0$$

where $||\cdot||_F$ denotes the Frobenius norm, A is the data matrix of samples by variables, H is the topics matrix which associates topics to variables, and W is a matrix which associates samples with topics. These names come from the fact that each sample is approximated by a linear combination of the rows in H with coefficients from the rows of W. The foundational work by Seung and Lee showed that the standard NMF objective can be optimized efficiently with a multiplicative update [SL01, LS99]. Additionally, W and H are chosen to be low rank so that the method learns to compress the data into features. The low rank, efficient multiplicative updates allow this method to scale to large datasets. Additionally, NMF is sought after for its sparse and interpretable feature learning [Gil14].

A potential drawback of the standard NMF model is that it does not directly account for stratified data, e.g. data drawn from multiple sources [ASM18]. Data collection methods, along with geographic or time differences, can introduce heterogeneity into the dataset. One common solution is to stratify data in order to obtain accurate results for subgroups of the data [HHM53, NOG20, RFC12]. In this chapter, we augment the NMF objective to account for stratified data.

We consider the setting where multiple groups, or strata, share a common topics dictionary with positive, strata dependent shifts. In this work, strata will be given in the problem, but can be obtained via meta-data or other common, external information. For example, articles written in various regions or across time periods differ in dialect or semantic progression. Although NMF may attribute the words "pop" and "soda" to the topic "beverage", it will be unable to explain the regional differences in the use of this word. Using Stratified-NMF, we can learn the strata dependent shifts and obtain a topic dictionary which measures the underlying commonalities between strata.

In this work, we develop a novel extension of NMF called Stratified-NMF, which is

able to account for heterogeneous data. We also derive multiplicative updates and show that the Stratified-NMF objective is non-increasing under our multiplicative update rules. Next we analyze our method on four datasets, spanning synthetic data, image data, census data, and natural language text. We experimentally validate the convergence properties of the method and empirically demonstrate the interpretability of Stratified-NMF. Code for our experiments is also publicly available and can be found at `https://github.com/chapman20j/Stratified-NMF`.

5.2 Proposed Method

We consider the case where the dataset has s strata, each with a data matrix $A(i) \in \mathbb{R}^{m_i \times n}$ ($\mathbb{R}$ denotes non-negative real numbers). The data is stored row-wise so that $A(i)_{jk}$ denotes the k'th attribute of data point j in stratum i. Each data point is assumed to be sampled from a distribution

$$\mathcal{D}_i = \mathcal{D} + z_i$$

where $\mathcal{D}$ is a shared, non-negative distribution and $z_i \in \mathbb{R}^n$ is a strata dependent shift. To account for this, we want to perform NMF on the matrix

$$A(i) - \begin{bmatrix} - & z_i^T & - \\ - & z_i^T & - \\ & \vdots & \\ - & z_i^T & - \end{bmatrix} = A(i) - \mathbf{1} z_i^T \tag{5.1}$$

but over all the strata, where $\mathbf{1}_i \in \mathbb{R}^{m_i}$ is the all ones vector. This leads to the following Stratified-NMF objective.

$$\underset{v(i),W(i),H}{\operatorname{argmin}} \sum_{i=1}^{s} ||A(i) - \mathbf{1} v(i)^T - W(i)H||_F^2, \tag{5.2}$$

$$v(i) \geq 0,\ W(i) \geq 0,\ H \geq 0,\ \forall 1 \leq i \leq s \tag{5.3}$$

Note that the low rank update to each stratum keeps the number of additional parameters in the NMF objective relatively low. This property is appealing since it prevents overfitting

to each stratum.

This formulation allows the model to naturally learn a global representation, H, of the data and local representations, $v(i)$, of each stratum. This relieves the model of inductive biases which may be introduced by other stratification techniques. One might consider more simple approaches like centering the means of the different strata or unit normalization of individual data points. The mean centering approach fails because the objective is non-linear and won't behave well under arbitrary additive shifts. Similarly, the unit normalization approach may flatten out data at different scales.

Algorithm 6: Stratified-NMF Multiplicative Update Algorithm

Input: Data $A(i) \in \mathbb{R}^{m_i \times n}$ for $1 \leq i \leq s$, number of iterations N, number of v updates per iteration M.

Construct initial $v(i) \in \mathbb{R}^n, W(i) \in \mathbb{R}^{m_i \times r}, H \in \mathbb{R}^{r \times n}$ for $1 \leq i \leq s$

for $k = 0, 1, \ldots, N-1$ **do**

  for $j = 0, 1, \ldots, M-1$ **do**

    $v(i) \leftarrow v(i) \frac{A(i)^T \mathbf{1}}{v(i) m_i + H^T W(i)^T \mathbf{1}} \; \forall i$

  end

  $W(i) \leftarrow W(i) \frac{A(i) H^T}{(W(i)H + \mathbf{1} v(i)^T) H^T} \; \forall i$

  $H \leftarrow H \frac{\sum_i W(i)^T A(i)}{\sum_i W(i)^T (W(i)H + \mathbf{1} v(i)^T)}$

end

5.3 Theoretical Results

Due to the similar structure of the Stratified-NMF objective and the standard NMF objective, we are able to develop multiplicative update rules described in Algorithm 6. Note that these multiplicative update rules satisfy the constraints in Equation 5.3 throughout the optimization procedure. Additionally, we state and prove Theorem 23 showing that the multiplicative updates satisfy similar convergence properties to those of the standard NMF

objective [SL01, DLR77].

Definition 3. *(From [SL01]) $G(h, h')$ is an auxiliary function for $F(h)$ if the conditions $G(h, h') \geq F(h)$, $\forall h'$ and $G(h, h) = F(h)$.*

Lemma 21. *(From [SL01]) If G is an auxiliary function, then F is nonincreasing under the update*

$$h^{t+1} = \operatorname*{argmin}_h G(h, h^t).$$

Theorem 22. *(From [SL01]) The Euclidean distance $||A - WH||_F$ is nonincreasing under the update rules*

$$H_{ij} \leftarrow H_{ij} \frac{(W^T A)_{ij}}{(W^T W H)_{ij}} \quad W_{ij} \leftarrow W_{ij} \frac{(AH^T)_{ij}}{(WHH^T)_{ij}} \tag{5.4}$$

Theorem 23 (Convergence of Multiplicative Update Rules)**.** *The Stratified-NMF objective defined in Equation 5.2 is non-increasing under the update rules defined in Lines 5, 7, and 8 of Algorithm 6.*

Proof. We reformulate the Stratified-NMF objective into a large block formulation resulting in a standard NMF objective with s additional columns in W and s additional rows in H:

$$\left\| \hat{A} - \hat{W}\hat{H} \right\|_F^2 \tag{5.5}$$

with

$$\hat{W} = \begin{bmatrix} \mathbf{1}_1 & \mathbf{0}_1 & \dots & \mathbf{0}_1 & W(1) \\ \mathbf{0}_2 & \mathbf{1}_2 & \dots & \mathbf{0}_2 & W(2) \\ \vdots & \vdots & \dots & \vdots & \vdots \\ \mathbf{0}_s & \mathbf{0}_s & \dots & \mathbf{1}_s & W(s) \end{bmatrix} \in \mathbb{R}^{\left(\sum_{i=1}^{s} m_i\right) \times (r+s)}, \tag{5.6}$$

where $\mathbf{0}_i \in \mathbb{R}^{m_i}$ denotes the zero vector and $\mathbf{1}_i \in \mathbb{R}^{m_i}$ denotes the all-ones vector, and

$$\hat{A} = \begin{bmatrix} A(1) \\ A(2) \\ \vdots \\ A(s) \end{bmatrix} \in \mathbb{R}^{(\sum_{i=1}^{s} m_i) \times n}, \ \hat{H} = \begin{bmatrix} v(1)^T \\ v(2)^T \\ \vdots \\ v(s)^T \\ H \end{bmatrix} \in \mathbb{R}^{(r+s) \times n}. \tag{5.7}$$

Note that the block matrix $\hat{H}$ contains parameters, all of which are optimized. Lee and Seung show the non-increasing property of the NMF objective for the W and H multiplicative updates separately [SL01]. Therefore to show convergence on the v and H, it is enough to show that we obtain the same update formulas from performing NMF on the block formulation. In particular, the standard NMF update says

$$\hat{H} \leftarrow \hat{H} \frac{\hat{W}^T \hat{A}}{\hat{W}^T \hat{W} \hat{H}}.$$

Applying this update rule gives

$$\begin{aligned} v(i)^T &= \hat{H}_i \\ &\leftarrow v(i)^T \frac{(0, ..., 0, \mathbf{1}^T(i), 0, ..., 0)\hat{A}}{(0, ..., 0, \mathbf{1}^T(i), 0, ..., 0)\hat{W}\hat{H}} \\ &= v(i)^T \frac{\mathbf{1}^T(i)A(i)}{(0, \cdots, 0, \mathbf{1}^T(i)\mathbf{1}(i), 0, \cdots, 0, \mathbf{1}^T(i)W(i))\hat{H}} \\ &= v(i)^T \frac{\mathbf{1}^T(i)A(i)}{m_i v(i)^T + \mathbf{1}^T(i)W(i)H}. \end{aligned}$$

Taking transposes on both sides recovers the Stratified-NMF update rule. Let $[H]$ denote the rows corresponding to H in $\hat{H}$. Then

$$\begin{aligned} H &\leftarrow H \frac{\hat{W}_{[H]}^T \hat{A}}{\hat{W}_{[H]}^T \hat{W} \hat{H}} \\ &= H \frac{(W(1)^T, \cdots, W(s)^T)\hat{A}}{(W(1)^T, \cdots, W(s)^T)\hat{W}\hat{H}} \\ &= H \frac{\sum_i W(i)^T A(i)}{(W(1)^T \mathbf{1}(1), \cdots, W(s)^T \mathbf{1}(s), \sum_i W(i)^T W(i))\hat{H}} \\ &= H \frac{\sum_i W(i)^T A(i)}{\sum_i W(i)^T \mathbf{1}(i) v(i)^T + W(i)^T W(i) H}. \end{aligned}$$

By Theorem 22, the Stratified-NMF objective is non-increasing under the multiplicative updates defined in Equation 5 and Equation 8.

Now we look at the multiplicative update for W. Following the proof technique in [SL01], consider the function

$$F_{\sigma,r}(w) = \frac{1}{2}\sum_{c}\left(a_c - v_c - \sum_{j} w_j(\sigma)H_{j,c}\right)^2,$$

where $w_j = W_{rj}(\sigma)$, $a_c = A(\sigma)_{rc}$, $v_c = v(c)$. We define K to be the diagonal matrix with entries

$$K_{aa} = \frac{(wHH^T + v^T H^T)_a}{w_a} = \frac{(wHH^T)_a}{w_a} + \frac{(v^T H^T)_a}{w_a} = K^1_{aa} + K^2_{aa}.$$

We also define

$$G_{\sigma,r}(w, w^t) = F_{\sigma,r}(w^t) + (w - w^t)\nabla F_{\sigma,r}(w^t) + \frac{1}{2}(w - w^t)K(w^t)(w - w^t)$$

and expand $F_{\sigma,r}$ to obtain

$$F_{\sigma,r}(w) = F_{\sigma,r}(w^t) + (w - w^t)\nabla F_{\sigma,r}(w^t) + \frac{1}{2}(w - w^t)HH^T(w - w^t).$$

To show that $G_{\sigma,r}$ is an auxiliary function of $F_{\sigma,r}$, it suffices to show that

$$K - HH^T$$

is positive semi-definite. We see that

$$K^1 + K^2 - HH^T \succeq K^1 - HH^T \succeq 0,$$

where the first inequality comes from the fact that K^2 is non-negative and diagonal. The second inequality was shown in the proof of Lemma 2 from [SL01] as K^1 is the same as K in their proof. Summing over the strata and rows, we obtain

$$G_{\sigma,r} \geq F_{\sigma,r},\ \forall \sigma, r \implies G = \sum_{\sigma,r} G_{s,r} \geq \sum_{\sigma,r} F_{\sigma,r} = F,$$

where F is the Stratified-NMF objective and minimizing G recovers the multiplicative update for W defined in Equation 7. In other words, G is an auxiliary function for F. By Lemma 21, the Stratified-NMF objective is non-increasing under the multiplicative update rule defined in Equation 7. □

5.4 Experimental Results

Next we experimentally analyze the performance of Stratified-NMF on synthetic, housing, image and text datasets. We show the merits of Stratified-NMF as a tool for interpretable unsupervised learning in these unsupervised learning applications. In all experiments H, $W(i)$ $\forall i$ are initialized as random uniform in $[0, 1/\sqrt{r}\,]$ entries, independently and identically distributed (iid) for all parameters. The $v(i)$ $\forall i$ are initialized as random uniform $[0, 1]$ iid entries for all parameters. The number of updates for v at each iteration (denoted M in Algorithm 6) is set to 2 for all experiments. We add 10^{-9} to all the entries in the denominator of each update to avoid division by zero. In this work, *loss* denotes the square root of the Stratified-NMF objective and *normalized loss* refers to the following quantity:

$$\sqrt{\frac{\sum_{i=1}^{s} ||A(i) - \mathbf{1}v(i)^T - W(i)H||_F^2}{\sum_{i=1}^{s} ||A(i)||_F^2}}.$$

5.4.1 Synthetic

We test the Stratified-NMF objective on a synthetic dataset containing 4 strata, $A(1)$, $A(2)$, $A(3)$, $A(4)$. Each matrix, $A(i) \in \mathbb{R}^{100\times 100}$ is constructed by multiplying a 100×5 matrix with a 5×100 matrix each of which have entries that are drawn uniformly from $[0, 1]$. We then add a synthetic strata feature of $\mathbf{1}v_{\text{true}}^T(i)$ where $v_{\text{true}}(i)$ has entries drawn uniformly from $[i-1, i]$. We run stratified-NMF for 10000 iterations and analyze the results.

Figure 5.1 shows the log-log plot of the normalized stratified NMF objective versus iterations. Observe that the multiplicative update detailed in Algorithm 6 converges on the synthetic dataset in under 2000 iterations and obtains a final normalized loss of $9.7e-4$. Additionally, in Figure 5.2, we measure the success of recovering the learned strata features by plotting the mean value of each $v(i)$ per iteration. Since $v_{\text{true}}^T(i)$ has entries drawn uniformly from $[i-1, i]$ we expect the $v(i)$'s to converge to $i-0.5$, which corresponds to 0.5 for $v(1)$, 1.5 for $v(2)$, 2.5 for $v(3)$, and 3.5 for $v(4)$. Figure 5.2 shows that the means converge to 0.51, 1.47, 2.53, 3.57, respectively. These means are very close to the expected means, which

highlights the utility of the $v(i)$'s in recovering information about the individual strata.

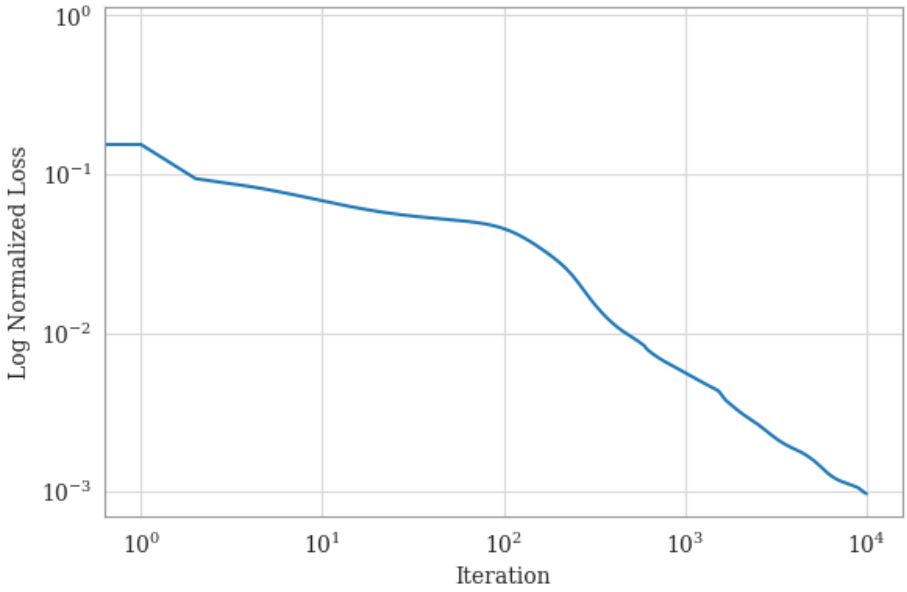


Figure 5.1: Log of normalized loss versus number of iterations for the synthetic experiment. The final normalized loss is $9.7e-4$.

5.4.2 California Housing

We use Stratified-NMF to analyze the California housing dataset originally compiled in 1997 available on scikit-learn [PB97, PVG11]. This dataset was curated from the 1990 US census data and contains average income, housing average age, average rooms, average bedrooms, population, average occupation, latitude, and longitude fields. We stratify the data by splitting it into low, medium and high income groups and drop the latitude and longitude fields due to the positivity constraint. The stratified dataset has 815 samples for the low income stratum all of which make less than $15k$ dollar household income per year, 1492 samples for the medium income stratum all of which make between $45k - 50k$ household income per year, and 308 samples for the high income stratum which make more than $100k$ per year. Although the sizes of the stratum are imbalanced, Stratified-NMF is able to capture meaningful local features. We run Stratified-NMF on the three strata for 100 iterations with

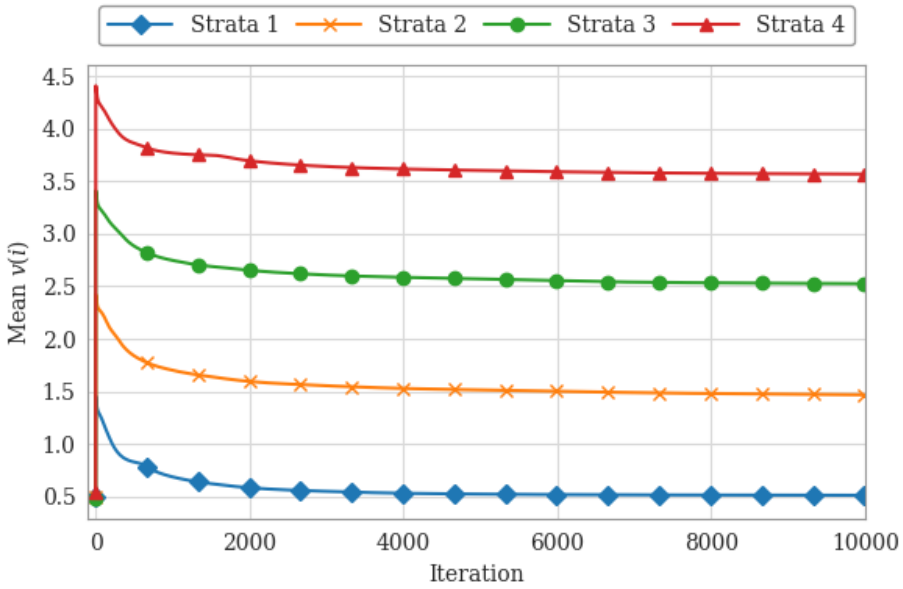


Figure 5.2: Means of each strata feature $v(i)$ over the four strata of the synthetic experiment. The final means are 0.51, 1.47, 2.53, 3.57, respectively. We expect to approximately recover the true means of 0.5, 1.5, 2.5, 3.5.

a rank of 5. We then analyze the interpretability of the learned strata features, $v(i)$, in Figure 5.3.

Figure 5.3 shows the normalized strata features $v(i)$ for the California dataset. This plot was obtained by taking the resulting $v(i)$'s and normalizing them so that $\sum_j v(i)_j = 1$ for all j. This depicts the relative differences between strata on the same plot. The median income variable shows an obvious trend that low, medium, and high income is preserved. Interestingly, house age and average bedrooms are relatively constant across strata, while the average rooms seems to increase with wealth. Lastly, the population is highest for the medium income stratum. An analysis by the Brookings Institute reported a similar trend found in survey data from 2017 [Ber18]. Note that average occupancy was removed from the plot because it was near zero across all strata and the normalization causes one stratum to visually dominate this variable.

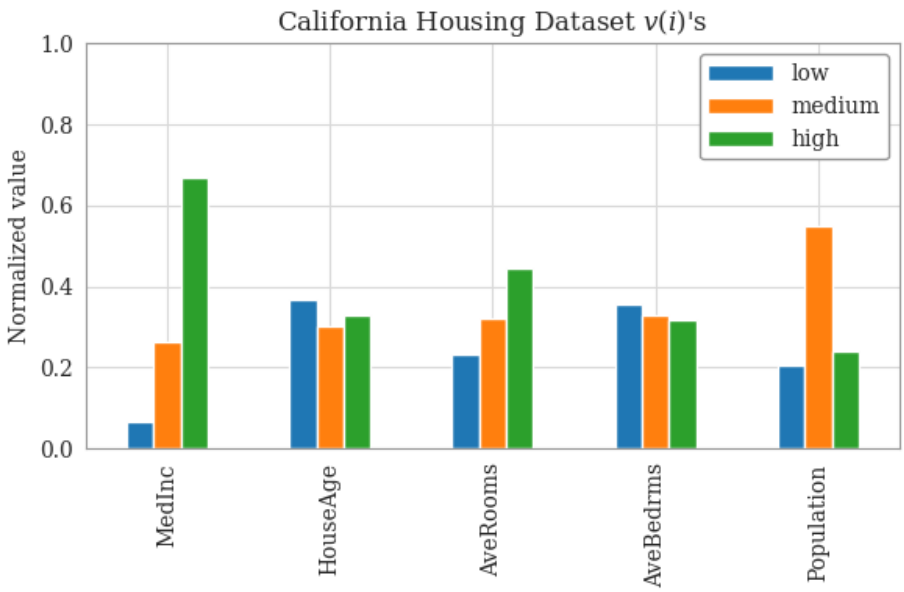


Figure 5.3: Normalized strata features $v(i)$ for the California dataset. This plot was obtained by taking the resulting $v(i)$'s and normalizing them so that $\sum_j v(i)_j = 1$ for all j. The median income and average rooms increase with income, as expected. Interestingly, median income individuals tend to live in more populous regions [Ber18]. Trends for the other variables are neutral.

5.4.3 MNIST

We also test our method on MNIST, a dataset of handwritten digits sourced from the Torchvision library [LCB10, MR10]. We stratify the data into two strata S_1, S_2. The S_1 stratum consists of 100 images of 1's and 100 images of 2's, while the S_2 stratum consists of 100 images of 2's and 100 images of 3's. Note that the samples in S_1 and S_2 are disjoint. We expect H to capture the 2's as global features since they are in both strata while we expect the v's to capture the 1's and 3's which are unique to their respective strata. Stratified-NMF is run on the dataset for 100 iterations with a rank of 5. The learned strata features, $v(i)$, are displayed in Figure 5.4 and the global features, H, are displayed in Figure 5.5.

In Figure 5.4 we observe that $v(1)$ captures the 1, which is unique to S_1, and $v(2)$ captures

the 3, which is unique to S_2. Many of the shared features shown in Figure 5.5 resemble 2's, which is the common data to both stratum. This suggests that the strata features learn information specific to each stratum while the global dictionary H learns features common to the whole dataset. The remaining feature in H resembles a 3. We suspect that this is due to the 2 and 3 not having enough commonality, which means that H must store features corresponding to 3's in order to fit the data. Note also that rows of H are features of the residuals $A(i) - \mathbf{1}v(i)^T$. This can be seen in Figure 5.5 as many of the features resemble numbers with missing lines or faint regions. The missing parts correspond to the parts extracted from $\mathbf{1}v(i)^T$.

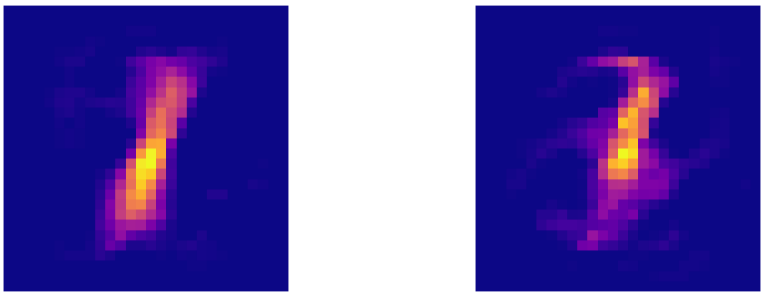


Figure 5.4: Learned strata features on the MNIST experiment. The left image is a plot of $v(1)$ and it resembles a one. The right image is a plot of $v(2)$ and it resembles parts of a three.

5.4.4 20 newsgroups dataset

Finally, we test our method on the 20 newsgroups dataset sourced from scikit-learn [PVG11, Ren08]. This dataset is composed of 20 categories of news articles encompassing a variety of topics including hockey, cryptography, space, and medicine. Each category contains between 799 and 999 articles from each news group. We preprocess the data by removing all stop words, headers, footers and quotes and subsequently compute a global text frequency inverse document frequency matrix. This yields a length 51840 sparse non-negative feature vector for each article. We stratify the data according to the news group and run Stratified-NMF

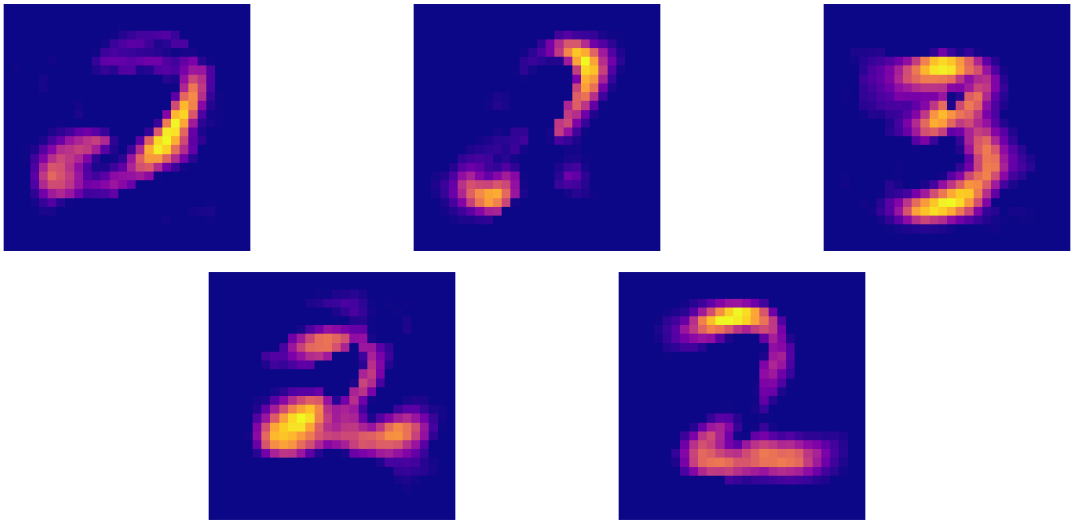


Figure 5.5: Learned topics matrix for the MNIST experiment. Each image is a row of H. These images capture global features across the residuals $A(i) - \mathbf{1}v(i)^T$. Four of the five learned global features are 2's with missing segments which appear to be captured by the $v(i)$'s. The remaining feature in H resembles a 3.

for 100 iterations with a rank of 20.

Figure 5.1 displays the top three, highest weighted, words captured by the strata features for each category. We observe that for each of the twenty stratum, the top words captured by the strata features correspond exactly to their topic. This highlights the interpretability of the local features. For example, the top three words in the autos newsgroup are "driving", "v6", and "transmission", all of which may be used when discussing automobiles. Additionally, the top words in the hockey newsgroup are "Toronto", "sabres", and "coach", all of which are associated with professional hockey teams. Finally, the top words in the medicine newsgroup are "kidney", "symptoms", and "doctors", all of which are associated with medicine. Upon looking at the top thirty words (contained in the Github), we were manually able to discern which category the words corresponded to. Finally, we observed that the local strata features were sparse with approximately 20% nonzero, highlighting sparsity preservation properties of Stratified-NMF.

Newsgroup	1	2	3
atheism	cheers	exist	bobbe
graphics	convert	points	vesa
misc computer	norton	tried	truetype
pc hardware	work	set	com
mac hardware	clock	upgrade	hardware
windows x	hp	colormap	r5
forsale	following	looking	manual
autos	driving	v6	transmission
motorcycles	tony	cop	wheel
baseball	hitting	jays	phillies
hockey	toronto	sabres	coach
cryptography	random	public	voice
electronics	detector	supply	chip
medicine	kidney	symptoms	doctors
space	new	long	astronomy
christian	churches	read	body
politics guns	jim	shot	clinton
politics mideast	west	civilians	countries
politics misc	national	laws	house
religion misc	life	cult	context

Table 5.1: Top three words learned for each newsgroup in the 20 newsgroups dataset. These are obtained by looking at the largest values in each $v(i)$. We observe that the forsale newsgroup has the words "following", "looking" and "manual", which are associated with selling or shipping items. Similarly, the baseball newsgroup has the words "hitting", "jays", and "phillies", which correspond to baseball or major league baseball teams.

5.5 Discussion

NMF is a popular unsupervised learning method with many extensions and variations [Gil14]. A very natural question is if one can apply these stratification techniques to existing NMF formulations. We would like to explore extending our method to other NMF variants such as sparsity constrained NMF [KP08, KKL23], semi-supervised NMF [WGW15, HKL20, VHR21], online NMF [GTL12], and tensor NMF [KKL23, Zaf09]. Augmenting these methods to include stratification could provide more insight into subgroups within the data. There is also additional work to be done on the theoretical side. Work by Lin further analyzes the convergence properties of NMF and proves convergence under slight modifications to the original NMF multiplicative update [Lin07]. Similar results should hold for Stratified-NMF, but this is outside of the scope of this chapter. We believe that the following present promising and important future work:

- Variable number of learnable parameters for each stratum
- More extensive testing on real world datasets
- Initialization of Stratified-NMF Matrices [WCD04]
- Automatic hyper-parameter tuning [LSM21]

Stratification of unsupervised algorithms may yield more interpretable results on smaller subgroups of data, while preserving global trends in the data.

An additional direction that we would like to explore is federated methods in the context of stratification. We note that the Stratified-NMF algorithm can be adapted to the centralized federated setting. In this setting we have a single centralized server and s sites, each of which have their own data matrix. First the central server constructs H and sends it to the sites. Then each site constructs and updates $v(i)$ and $W(i)$ based on their own data. The sites then set $W(i)^T A(i)$, and $W(i)^T(W(i)H + \mathbf{1}v(i)^T)$ to central server allowing the central

server to update H. We note that in this method, the data matrices $A(i)$ are never exposed to other sites or the central server.

5.6 Conclusion

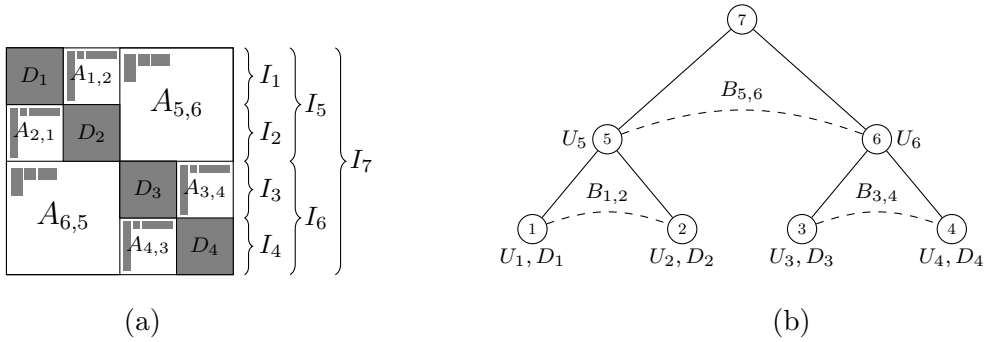
In this chapter we propose Stratified-NMF to extend NMF to capture heterogeneous data. We first propose the Stratified-NMF objective function and derive multiplicative update rules. Then, we prove that the Stratified-NMF objective is non-increasing with respect to the derived, multiplicative updates. Finally, we explore the application of our method to unsupervised learning tasks spanning different modalities including synthetic data, census data, image data, and natural language text data. Stratified-NMF is able to simultaneously capture local and global strata information in the dataset. We highlight the interpretability of the local and the global features learned by the method.

CHAPTER 6

Hierarchically Semi-Separable matrix Construction Algorithm

This chapter is an adaptation of Section 2.1 and Appendix C in [YMG23] which explains a hierarchically semi-separable (HSS) matrix construction algorithm. In this framework we are given a dense matrix and we construct a compressed version of this matrix in a nested hierarchical format. After constructing the HSS representation of a matrix, down stream tasks can be done more efficiently. For example, given an $n \times n$ matrix with an HSS rank of k, the rank of the off diagonal matrix blocks, we can apply a matrix-vector multiplication in $O(nk)$ via the HSS format while this same operation would require $O(n^2)$ operations in the standard dense format [Mar11]. Usually, we assume that k is much smaller than n yielding a large speedup of downstream tasks, while incurring an up front construction cost of $O(nk^2)$. Additional downstream tasks that can be implemented more efficiently with access to the HSS format include Quadratic complexity explicit Cholesky factorization of symmetric positive definite (SPD) matrices, Linear complexity explicit ULV factorization of SPD matrices, and efficient computation of system solution when given a right hand side [XCG10]. The HSS construction algorithm that we propose leverages matrix sketching, probing the column space and row space of the matrix by multiplying it with random test vectors which are drawn from various distributions. The goal of this chapter is to explain and trace our proposed HSS construction algorithm in detail which is a generalization of the algorithm proposed in [GCG19]. We extend the algorithm in [GCG19] which uses Gaussian sketching to a broader class of random matrices called Johnson-Lindenstrauss sketching matrices.

6.1 HSS Format



(a) (b)

Figure 6.1: (a) Illustration of a symmetric HSS matrix using 3 levels. Diagonal blocks are partitioned recursively. Gray blocks denote the basis matrices. (b) Tree for the HSS matrix from (a), using topological ordering. All nodes except the root store U_i (and V_i for the non-symmetric case). Leaves store D_i, non-leaves B_{ij} (and B_{ji} for the non-symmetric case).

Consider a square matrix $A \in \mathbb{C}^{n \times n}$ and index set $I_A = \{1, \ldots, n\}$. The HSS matrix representation is a hierarchical block 2×2 partitioning of the matrix, where all off-diagonal blocks are compressed, or approximated, using a low-rank product, see Figure 6.1a. The hierarchical structure is succinctly described by a binary tree $\mathcal{T}$, called *cluster tree*, as depicted in Figure 6.1b. The recursive partitioning stops at the leaf level, which corresponds to the smallest block size of the partition. The leaves do not need to be of uniform size, because for certain input matrices a non-uniform partition may be preferable for better compression.

Each node $\tau \in \mathcal{T}$ is associated with a contiguous subset $I_\tau \subset I_{\text{root}(\mathcal{T})}$. We use $\#I_\tau$ to denote the cardinality of I_τ. For two children ν_1 and ν_2 of τ, it holds that $I_{\nu_1} \cup I_{\nu_2} = I_\tau$ and $I_{\nu_1} \cap I_{\nu_2} = \emptyset$. It follows that $\cup_{\tau \in \text{leaves}(\mathcal{T})} I_\tau = I_{\text{root}(\mathcal{T})} = I_A$. The same tree $\mathcal{T}$ is used for the rows and the columns of A. Commonly, the tree nodes are numbered in a *postorder*, and most of the HSS algorithms, such as construction, matrix-vector multiplication, factorization and solve etc., can be described as traversing the cluster tree following this postorder. However, in the parallel implementation and throughout this chapter, we traverse the cluster tree following a bottom-up *topological order*, i.e., from the leaf level to the root, level by level,

displayed in Figure 6.1b.

Each leaf node τ of $\mathcal{T}$ corresponds to a diagonal blocks of A, denoted as D_τ, and is stored as a dense matrix : $D_\tau = A(I_\tau, I_\tau)$. At each node τ, the off-diagonal block $A(I_\tau, I_A \setminus I_\tau)$ is called a row *Hankel block*, and the off-diagonal block $A(I_A \setminus I_\tau, I_\tau)$ is a column Hankel block. The compression algorithm sweeps through the tree bottom-up. At each tree node, it computes the column basis for the row Hankel block and row basis for the column Hankel block. Note that all the blocks within a row (column) Hankel block share the same column (row) basis. The HSS algorithm reduces complexity further: each internal node recycles the bases computed at the two children nodes, thus, the basis at each internal node has the nested structure (see Equation Equation (6.2)), called *nested basis* property, which we describe now. For a node τ with two children ν_1 and ν_2, the off-diagonal block $A_{\nu_1,\nu_2} = A(I_{\nu_1}, I_{\nu_2})$ is factored (approximately) as

$$A_{\nu_1,\nu_2} \approx U^{\text{big}}_{\nu_1} B_{\nu_1,\nu_2} \left(V^{\text{big}}_{\nu_2}\right)^* , \tag{6.1}$$

where $U^{\text{big}}_{\nu_1}$ has dimensions $\#I_{\nu_1} \times r^r_{\nu_1}$, B_{ν_1,ν_2} is a submatrix of A_{ν_1,ν_2} with dimensions $\#I_{\nu_1} \times \#I_{\nu_2}$ and $V^{\text{big}}_{\nu_2}$ has dimensions $\#I_{\nu_2} \times r^c_{\nu_2}$(superscripts r and c are used to denote that $U^{\text{big}}/V^{\text{big}}$ are column/row bases for the row/column Hankel blocks of A). The *HSS-rank* r is defined as the maximum of r^r_τ and r^c_τ over all off-diagonal blocks, where typically $r \ll N$. B_{ν_1,ν_2} and B_{ν_2,ν_1} are stored at the parent node. For a node τ with children ν_1 and ν_2, U^{big}_τ and V^{big}_τ are represented hierarchically as

$$U^{\text{big}}_\tau = \begin{bmatrix} U^{\text{big}}_{\nu_1} & 0 \\ 0 & U^{\text{big}}_{\nu_2} \end{bmatrix} U_\tau \quad \text{and} \quad V^{\text{big}}_\tau = \begin{bmatrix} V^{\text{big}}_{\nu_1} & 0 \\ 0 & V^{\text{big}}_{\nu_2} \end{bmatrix} V_\tau \,. \tag{6.2}$$

Note that for a leaf node $U^{\text{big}}_\tau = U_\tau$ and $V^{\text{big}}_\tau = V_\tau$. Hence, every node τ, except the root, keeps matrices U_τ and V_τ. The top two levels of the example shown in Figure 6.1a can be written out explicitly as

$$A = \begin{bmatrix} D_1 & U_1 B_{1,2} V_2^* & \begin{bmatrix} U_1 & 0 \\ 0 & U_2 \end{bmatrix} U_5 B_{5,6} V_6^* \begin{bmatrix} V_3^* & 0 \\ 0 & V_4^* \end{bmatrix} \\ U_2 B_{2,1} V_1^* & D_2 & \\ \begin{bmatrix} U_3 & 0 \\ 0 & U_4 \end{bmatrix} U_6 B_{6,5} V_5^* \begin{bmatrix} V_1^* & 0 \\ 0 & V_2^* \end{bmatrix} & D_3 & U_3 B_{3,4} V_4^* \\ & U_4 B_{4,3} V_3^* & D_4 \end{bmatrix}. \tag{6.3}$$

Only at the leaf nodes, where $U_\tau^{\text{big}} \equiv U_\tau$, is the U_τ^{big} stored explicitly. A similar relation holds for the V_τ basis matrices. For symmetric matrices, $U_i \equiv V_i$ and $B_{ij} \equiv B_{ji}$.

HSS matrix construction based on randomized sampling techniques has attracted a lot of attention in recent years. Compared to standard HSS construction techniques [XCG10, WLX13] which assume that an explicit matrix is given as input, randomized techniques allow the design of *matrix-free* construction algorithms. A *fully matrix-free* construction algorithm relies solely on the availability of a matrix-vector product routine [LM22].

A *partially matrix-free* algorithm relies on a matrix-vector product routine and additionally requires access to some entries of the matrix [Mar11, GCG19]. For certain applications, for example Toeplitz systems, where fast (e.g., linear time) matrix-vector products exist, a randomized algorithm typically has linear or log-linear complexity instead of quadratic complexity with the standard construction algorithms.

We describe and improve upon a partially matrix-free algorithm and its adaptive version, which is presented in Algorithm 7 and Algorithm 8. Note that the description of Algorithm 7 is for a symmetric matrix, which is easier to understand. Our implementation in STRUMPACK [str] is for nonsymmetric matrices.

6.2 Adaptive HSS Algorithm

We describe the adaptive HSS algorithm originally proposed in [GCG19] which is partially matrix-free and leverages sketching which is generalized in Algorithm 7 and Algorithm 8. One of the benefits of this algorithm is that the matrix A does not need to be explicitly formed, only a matrix-vector computation routine and access to $O(nr)$ entries of A is required.

Instead of compressing the Hankel block itself at each node, we compress a sketch of the Hankel block from which we can recover the compressed version of the off diagonal block. Then, as we traverse up the tree we combine local sketches from both of the children Hankel blocks, and subtract off the already compressed low rank blocks to recover a local sketch for the parent Hankel block that is written in the basis of the children blocks. Finally, this local sketch can be compressed, leveraging the nested basis property. This procedure is described in equations (2.5)-(2.9) of [GCG19] and in detail in Section 6.3.

We leverage an interpolative decomposition to compress the off diagonal Hankel blocks. Given a matrix A with dimensions $m \times m$ with rank $r \ll m$. We can write an interpolative decomposition of A as $A = UA(J,:)+O(\varepsilon)$. Where U has dimensions $m \times r$ and J is an index set of r rows. This interpolative decomposition can be computed using a rank revealing QR factorization, detailed in equation (2.4) of [GCG19].

Remark 1. *In practice, the interpolative decomposition is computed using a rank revealing QR factorization as $A = A(:,J)V$ which computes a column basis. To compute a row basis, we compute the interpolative decomposition of $A^* = A^*(:,J)V$ and apply the conjugate transpose so $A = V^*A(J,:)$, then we can rename $V^* = U$ so $A = UA(J,:)$ resulting in a row basis.*

We can represent a low rank matrix or Hankel block in our case as a basis matrix U and a sampling of the rows. Once we compute an interpolative decomposition for both Hankel row block and Hankel column block that intersect a low rank off diagonal HSS block we can combine the bases and the selected row and column samplings to create a low rank approximation. By leveraging the interpolative decomposition to do our local compression we recover an approximate basis for the Hankel block and a sampling of the most important r rows of our sketch. The r rows of the sketch correspond to r rows of the original matrix A, allowing us to only use our sketch to compress the Hankel blocks as long as the sketch of the Hankel block is representative of the original Hankel block.

In most practical problems, the HSS rank, rank of the low dimensional off diagonal blocks,

is not known *a priori*, hence, the size of the sketching operator needs to be chosen adaptively. Previously, Gorman et al. [GCG19] developed a *blocked incrementing strategy* which fully reuses the already-computed basis set in two ways: (1) at each HSS tree node τ, if the initial samples are not sufficient, we increase a block of samples Δd, and augment τ's orthogonal basis by this amount; (2) This augmented basis will cause basis sets of the ancestor nodes to have sizes at least as large as that of τ, while the basis sets of the descendant nodes are not affected. Algorithm 7 and Algorithm 8 illustrates the high level HSS compression procedure with adaptation built in. The algorithm traverses the cluster tree in a topological order bottom-up. Initially each node τ is assigned the UNTOUCHED state. At the time when a node τ is to be compressed, all its descendant nodes are already COMPRESSED, and all its ancestral nodes are UNTOUCHED. Line 7 of Algorithm 8 tests to see whether the sketch for τ is sufficiently representative. If so, τ is compressed and its state is changed to COMPRESSED. If not, in lines 15-17 of the else-branch, we extend the sketching operator by Δd columns, change τ's state to PARTIALLY_COMPRESSED, and traverse the tree again with the following actions: (1) for the nodes below τ, we only subtract the newly added sketch from the diagonal blocks (lines 12 and 14 of Algorithm 7); (2) for the current PARTIALLY_COMPRESSED node τ, we augment the already-computed basis set with the new Δd columns from the sketch (lines 9-10 of Algorithm 8); (3) for τ's ancestral nodes, compression proceeds with the entire sketch (line 3 of Algorithm 8).

In the original adaptive compression algorithm from [GCG19] the global sketch of the matrix A was computed using a Gaussian sketching operator. This sketching operator is dense so it requires $O(n^2)$ time to compute an additional column when trying to expand the sketch. Algorithm 7 and Algorithm 8 are an extension of the algorithm to any Johnson–Lindenstrauss sketching operator. One such sketching operator is the sparse JL sketching operators, which can be applied faster. In [YMG23] we state and prove the theory that generalizes our stated algorithm. Additionally, we use sparse sketching operators as a case study, highlighting a 1.5-2.5 times speedup over the original Gaussian sketching operators with similar performance. Our code is available through the STRUMPACK C++ library

`cols`(A)	number of columns in matrix A
`JL-Operator`(d, n)	a $d \times N$ matrix drawn from a JL Distribution
`isleaf`(τ)	`true` if τ is a leaf node, `false` otherwise
`children`(τ)	a list with the children of node τ, always zero or two
`isroot`(τ)	`true` if τ is a root node, `false` otherwise
$\{Q, \Omega\} \leftarrow$ `QR`(S)	$S = Q\Omega$ where Q is orthogonal, Ω is upper triangular
`level`(τ)	level of node τ, starting from 0 at the root
$\{Y, J\} \leftarrow$ `ID`$(S, \varepsilon_r, \varepsilon_a)$	interpolative decomposition: $S \approx S(:, J)Y$

Table 6.1: List of helper functions for Algorithm 7 and Algorithm 8.

[str]. This broader class of sketching operators allows practitioners to use random matrices tailored to specific applications with similar time complexity guarantees. Next, we formally define and discuss the random matrices that we consider.

We begin by stating the classical Johnson–Lindenstrauss lemma [JL84]. The particular version below is from [DG03].

Lemma 24 (Johnson–Lindenstrauss (JL) Lemma [JL84])**.** *Given $\varepsilon \in (0, 1)$ and an integer m, let d be a positive integer such that $d \geq 4(\varepsilon^2/2 - \varepsilon^3/3)^{-1} \log m$. For any set P of m points in $\mathbb{R}^n$ there exists $f : \mathbb{R}^n \to \mathbb{R}^d$ such that for all $u, v \in P$*

$$(1 - \varepsilon)\|u - v\|^2 \leq \|f(u) - f(v)\|^2 \leq (1 + \varepsilon)\|u - v\|^2. \tag{6.4}$$

Lemma 24 does not say anything about *how* to construct f and what form it might take. In practice, f is usually chosen to be a linear map in the form of a matrix which is drawn randomly from an appropriate distribution. The following definition captures this idea.

Definition 4 (JL Sketching Operator)**.** *Suppose $\mathcal{D}$ is a distribution over matrices of size $d \times n$. We say that a matrix $R \sim \mathcal{D}$ is a $(n, d, \delta, \varepsilon)$-JL sketching operator if for any vector*

$x \in \mathbb{R}^n$ *it satisfies*

$$\Pr_{R \sim \mathcal{D}} \left[\left| \|Rx\|^2 - \|x\|^2 \right| > \varepsilon \|x\|^2 \right] < \delta.$$

The condition in Definition 4 considers length preservation of a single vector. A standard union bound argument can be used to show that a JL matrix with probability $1 - \delta$ satisfies (6.4) for all $u, v \in P$ where P contains m points, provided that d is chosen to be sufficiently large; see Remark 2.2 of [BKW21] for a discussion about this.

Remark 2. *For low rank matrices A with dimension $m \times n$, a highly accurate approximation can be computed with $d \ll n$ allowing us to compute our sketch $S = AR^T$ using only matrix-vector products by iterating over the d columns of R^T.*

Next, we introduce three popular JL sketching operator distributions. All three satisfy the condition in Definition 4 provided that d is large enough.

Gaussian Sketching Operator: a Gaussian sketching operator R of size $d \times n$ has entries which are drawn independently from a normal distribution with mean zero and variance $1/d$. We indicate that R is drawn from such a distribution by writing $R \sim \text{Gaussian}(n, d)$. Gaussian sketching operators are JL sketching operators if the dimension d is sufficiently large [DG03]. Key advantages of Gaussian sketching operators are ease of construction and that they lend themselves to simple and clean theoretical analysis [MT20, Remark 8.2]. The main downside of the Gaussian sketching operator is that it is relatively slow to apply since it has no particular structure and is dense. The sketching operators described below address this issue by using fast structured or sparse operators, respectively.

Subsampled Randomized Hadamard Transform: a subsampled randomized Hadamard transform (SRHT) of size $d \times n$ takes the form $R = PHD$. The matrix $D \in \mathbb{R}^{n \times n}$ is diagonal with the diagonal entries drawn independently from the Rademacher distribution, i.e., each entry is $+1$ with probability $1/2$ and -1 with probability $1/2$. The matrix $H \in \mathbb{R}^{n \times n}$ is the normalized Hadamard matrix, a deterministic unitary matrix which can be applied to a vector in $O(n \log(n))$ time instead of $O(n^2)$. The normalized Hadamard matrix can be

defined recursively via $H_0 = [1]$ and $H_{2n} = [H_n, H_n; H_n, -H_n]$. Finally, $P \in \mathbb{R}^{d \times n}$ is a sparse random sampling matrix whose rows are chosen independently and uniformly at random from the set $\{\sqrt{n/d} \cdot e_j^T\}_{j=1}^n$ where $e_j \in \mathbb{R}^n$ is the jth canonical basis vector. We indicate that R is drawn in this fashion by writing $R \sim \text{SRHT}(n, d)$. An early version of the SRHT appeared in [AC06] where each entry of P was independently chosen to be either zero or nonzero, with the nonzero entries drawn from an appropriately scaled normal distribution.

Sparse Johnson–Lindenstrauss Transform (SJLT): The sparse Johnson-Lindenstrauss transform (SJLT) was first introduced in [KN14] with subsequent further analysis in [NN13, CJN18]. An SJLT matrix R of size $d \times n$ has a fixed number $\alpha \in [d]$ of nonzero entries per column. The nonzero entries are drawn independently from a scaled Rademacher distribution, taking values in $\{1/\sqrt{\alpha}, -1/\sqrt{\alpha}\}$ uniformly at random. The paper [KN14] proposes two different methods for randomly drawing the position of the nonzero entries in R. The first method draws the α nonzero positions for each column of R uniformly at random from $[d]$. The second method divides the length-d columns of R into d/α chunks, and for each chunk a single entry is selected uniformly at random to be nonzero. This method requires d/α to be an integer. For both methods, sampling is done for each column independently of the nonzero positions in the other columns. The two approaches to constructing an SJLT are referred to as the *graph construction* and *block construction*, respectively. Throughout the paper, we will denote an SJLT drawn using either construction by $R \sim \text{SJLT}(n, d, \alpha)$. We implement both approaches in our software and allow the user to select which one to use.

In the next section we trace the HSS algorithm that we describe in Algorithm 7 and Algorithm 8. We hope that by tracing through the algorithm, we are able to simplify many of the complexities. We show where and how the adaptive steps occur. Additional theoretical and experimental details can be found in [YMG23].

Algorithm 7: Adaptive HSS compression of $A \in \mathbb{C}^{n\times n}$ using cluster tree $\mathcal{T}$ with tolerances ε_{rel} and ε_{abs}, see Table 6.1 for helper function details.

function $H =$ `HSSCompressAdaptive`(A, $\mathcal{T}$, d_0, Δd)

  $d \leftarrow d_0$, $n \leftarrow$ `cols`(A), $R \leftarrow$ `JL-Operator`$(d+\Delta d, n)$, $S \leftarrow AR^T$

  foreach $\tau \in \mathcal{T}$ **do** τ.state $\leftarrow$ `UNTOUCHED`

  while `root`$(\mathcal{T})$.state $\neq$ `COMPRESSED` ***and*** $d < d_{\max}$ **do**

    foreach $\tau \in \mathcal{T}$ ***in topological order*** **do**

      if τ.state $=$ `UNTOUCHED` **then**

        if `isleaf`(τ) **then** $D_\tau \leftarrow A(I_\tau, I_\tau)$

        else $\nu_1, \nu_2 \leftarrow$ `children`(τ) $B_\tau \leftarrow A(\widetilde{I}_{\nu_1}, \widetilde{I}_{\nu_2})$

        $\iota \leftarrow 1 : d + \Delta d$

      else $\iota \leftarrow d+1 : d+\Delta d$

      if `isroot`(τ) **then** τ.state $\leftarrow$ `COMPRESSED` **break**

      if `isleaf`(τ) **then** $S_\tau(:, \iota) \leftarrow S(I_\tau, \iota) - D_\tau\ R^T(I_\tau, \iota)$

      else

$$S_\tau(:,\iota) \leftarrow \begin{bmatrix} S_{\nu_1}(J_{\nu_1}, \iota) - B_\tau\ R_{\nu_2}(:,\iota) \\ S_{\nu_2}(J_{\nu_2}, \iota) - B_\tau^*\ R_{\nu_1}(:,\iota) \end{bmatrix}$$

      if τ.state $\neq$ `COMPRESSED` **then**

        `CompressAdaptiveNode` (τ) `// Algorithm 8`

      if `isleaf`(τ) **then**

        $R_\tau(:,\iota) \leftarrow U_\tau^*\ R^T(I_\tau, \iota)$; $\quad \widetilde{I}_\tau \leftarrow I_\tau(J_\tau)$

      else

$$R_\tau(:,\iota) \leftarrow U_\tau^* \begin{bmatrix} R_{\nu_1}(:,\iota) \\ R_{\nu_2}(:,\iota) \end{bmatrix}; \quad \widetilde{I}_\tau \leftarrow \begin{bmatrix} I_{\nu_1} & I_{\nu_2} \end{bmatrix}(J_\tau)$$

    end

  end

  return $\mathcal{T}$

Algorithm 8: Adaptive HSS compression of cluster tree element $\tau \in \mathcal{T}$ see Table 6.1 for helper function details.

1 **function** τ = `CompressAdaptiveNode(`τ`)`

2 **if** τ.state = `UNTOUCHED` **then**

3 $\{Q_\tau, \Omega_\tau\} \leftarrow$ `QR(`$S_\tau(:, 1:d)$`)`

4 $\widetilde{S} \leftarrow S_\tau(:, d+1:d+\Delta d)$ `// last` Δd `columns`

5 $\widehat{S} \leftarrow (I - Q_\tau Q_\tau^*)\widetilde{S}$

6 $\varepsilon_{\text{abs}}^\tau \leftarrow \varepsilon_{\text{abs}}/$`level(`$\tau$`)`; $\varepsilon_{\text{rel}}^\tau \leftarrow \varepsilon_{\text{rel}}/$`level(`$\tau$`)`

7 **if** $\|\widehat{S}\|_F < \varepsilon_{\text{abs}}^\tau$ ***or*** $\|\widehat{S}\|_F < \varepsilon_{\text{rel}}^\tau \|\widetilde{S}\|_F$ **then**

8 **goto** line 12

9 $\{\widehat{Q}, \widehat{\Omega}\} \leftarrow$ `QR(`$\widehat{S}$`)`

10 $Q_\tau \leftarrow \begin{bmatrix} Q_\tau & \widehat{Q} \end{bmatrix}$

11 **if** `min(diag(`$|\widehat{\Omega}|$`))` $< \varepsilon_{\text{abs}}^\tau$ ***or*** `min(diag(`$|\widehat{\Omega}|$`))` $< \varepsilon_{\text{rel}}^\tau |(\Omega_\tau)_{11}|$ **then**

12 $\{U_\tau^*,\ J_\tau\} \leftarrow$ `ID(`S_τ^*, $\varepsilon_{\text{rel}}^\tau$, $\varepsilon_{\text{abs}}^\tau$`)`

13 τ.state $\leftarrow$ `COMPRESSED`

14 **else**

15 $\bar{R} \leftarrow$ `JL-Operator(`$\Delta d, n$`)` `// extending sketch`

16 $d \leftarrow d + \Delta d$; $S \leftarrow \begin{bmatrix} S & A\bar{R}^T \end{bmatrix}$; $R^T \leftarrow \begin{bmatrix} R^T & \bar{R}^T \end{bmatrix}$

17 τ.state $\leftarrow$ `PARTIALLY_COMPRESSED`

18 **break** `// update all` τ`s required`

6.3 Tracing Adaptive HSS Algorithm

Here we describe the steps to compress a symmetric HSS matrix A with dimensions $4k \times 4k$ and HSS rank $r \ll k$ represented by a three level HSS tree shown in Figure 6.2 using Algorithm 7 and Algorithm 8. Assume that R^T has dimensions $4k \times l_1$. Initially, we compute $S = AR^T$ which has dimensions $4k \times l_1$.

We begin at the leaf level of the HSS tree where we can compress nodes one through four in parallel. We will compress the first node, corresponding to the first Hankel row block, whose rows we have highlighted in Figure 6.3. By symmetry this also corresponds to the columns of the first Hankel column block.

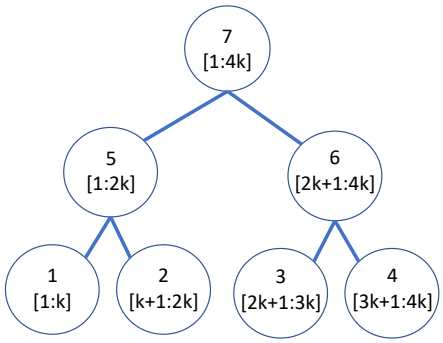


Figure 6.2: Three level HSS tree for our compression example with the nodes labeled and the corresponding indices in brackets.

6.3.1 Compression of a Leaf Node

First, we store the dense diagonal matrix D_1 in our leaf node 1 this is line 7 of Algorithm 7. Next, since we do not have the matrix A but instead just the sketch $S = AR^T$ we must figure out what the local sketch of the Hankel row block $H_1 = A(1:k, 1:4k \setminus 1:k) = A(1:k, k+1:4k)$ is (the first k rows excluding the dense diagonal). We compute a sketch of our Hankel row block $S^1_{\text{loc}} = [0, H_1]R^T$ by writing $[0, H_1]R^T = ([D_1, H_1] - [D_1, 0])R^T = (A(1:k, :) - [D_1, 0])R^T = S(1:k, :) - D_1R^T(1:k, :)$ which is line 12 of Algorithm 7.

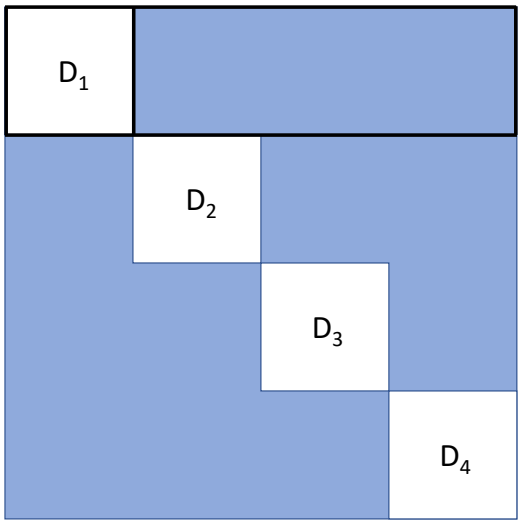


Figure 6.3: Leaf level of HSS tree with the first node rows in a box.

Next, to compress our approximation of H_1 which is S_1^{loc} with dimensions $k \times l_1$ lines 2-11 of Algorithm 8 verify that S_1^{loc} is a good enough approximation of H_1. For now, we will assume that it is and skip these lines. Later we will see how if the sketch is not accurate enough, we extend the sketching operator R^T (lines 15-18 of Algorithm 8) by appending columns to it which will require a small modification to the local sketches. We compute an interpolative decomposition of S_1^{loc} on line 12 of Algorithm 8 such that $S_1^{\text{loc}} \approx U_1 S_1^{\text{loc}}(J_1, :)$ where U_1 has dimensions $k \times r$ and J_1 is a subset of r distinct indices in $[1 : k]$. Then we set the state of node one to compressed (line 13). The interpolative decomposition cleverly gives us a low rank factorization for all of H_1 where U_1 could be thought of as a basis for the Hankel block and J_1 is an index set of rows which define the block. Since $S_1^{\text{loc}} = [0, H_1]R^T \approx U_1 S_1^{\text{loc}}(J_1, :) = U_1[0, H_1](J_1, :)R^T$ and R^T is full column rank with high probability we have that $[0, H_1] \approx U_1[0, H_1](J_1, :)$. So we have found a low rank factorization for the Hankel row block which we display in Figure 6.4.

We can now repeat this process for the rest of the leaf nodes which would result in matrices U_2, U_3, U_4 (dimensions $k \times r$) and index sets J_2, J_3, J_4 (of size r) being computed

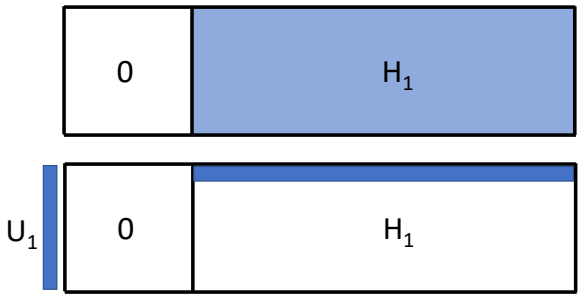


Figure 6.4: Compression of the first Hankel block H_1 into U_1, a basis matrix, and r rows of the original Hankel block, denoted by the thin horizontal stripe (not necessarily the first r rows) and indexed by index set J_1.

and stored. For the non-symmetric case we would also compress all of the leaf nodes for the column Hankel blocks as well. We display the result in Figure 6.5 where we additionally denote the low rank blocks L_1–L_4 which we would like to have compressed.

Remark 3. *The Hankel block does not need to be a contiguous nonzero block, for example* $H_2 = [A(k+1:2k, 1:k), 0, A(k+1:2k, 2k+1:4k)]$ *because* D_2 *is subtracted to compute* H_2.

Next, We show that we have already computed a low rank factorization for L_1–L_4 based on the interpolative decompositions of both the row and column of the two Hankel blocks that intersect at the low rank block. We detail how to compress L_1 in Figure 6.6. Since we have a row basis for H_1 we can just take the indices of the rows that intersect with L_1. So we have the factorization $L_1 \approx U_1 A(J_1, k+1:2k)$. Similarly, we have basis for the column Hankel block H_2^T which intersected with L_1 because we assumed that our matrix A was symmetric. So the column factorization for L_1 is the conjugate transpose of the row factorization for L_2 which we have already computed, thus $L_1 \approx A(1:k, J_2)U_2^*$ we can rename U_2^* as V_2 for clarity in the non-symmetric case where the second column Hankel block does not correspond to the conjugate transpose of the second row Hankel block. Combining the row and column

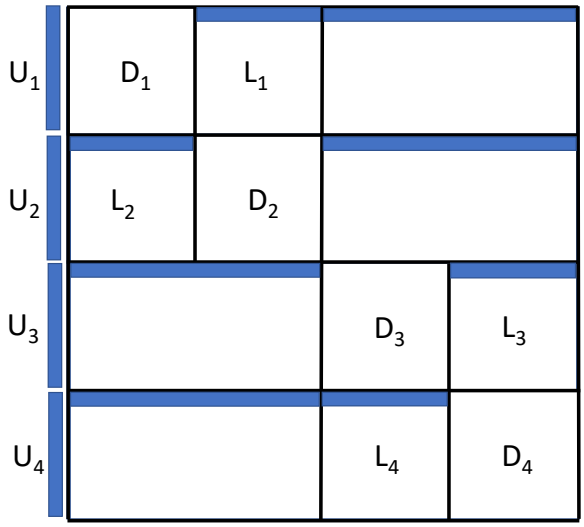


Figure 6.5: HSS matrix after all four row leaves have been compressed with the low rank blocks, L_1–L_4 sections listed .

factorizations, we have the low rank factorization $L_1 \approx U_1 A(J_1, J_2) U_2^* = U_1 A(J_1, J_2) V_2$. Notice that we currently do not have $A(J_1, J_2)$, the small $r \times r$ matrix of entries of A. This will be queried and stored in the parent node in the next level of the algorithm (line 8 in Algorithm 7). For completeness we can factorize $L_2 \approx U_2 A(J_2, J_1) U_1^*$, $L_3 \approx U_3 A(J_3, J_4) U_4^*$ and $L_4 \approx U_4 A(J_4, J_3) U_3^*$.

The final step that occurs at each leaf node is to compute R_i^{loc} which corresponds to the sketching operator R^T in the local column basis for the low rank block we have compressed. This will allow us to re-use the computation from our leaf nodes and subtract off the already compressed low rank blocks when trying to compress the parent nodes. Additionally, this allows us to leverage the nested basis property. So for the first leaf node, we compute and store $R_1^{\text{loc}} = U_1^* R^T(1:k, :)$.

We have completed our compression for the first node, we store five variables: 1. D_1, 2. U_1, 3. J_1 which is the dense diagonal block and what we use to represent the Hankel row block for rows $[1:k]$ and part of the low rank factorization for L_1 and we store 4. S_1^{loc}, 5. R_1^{loc}

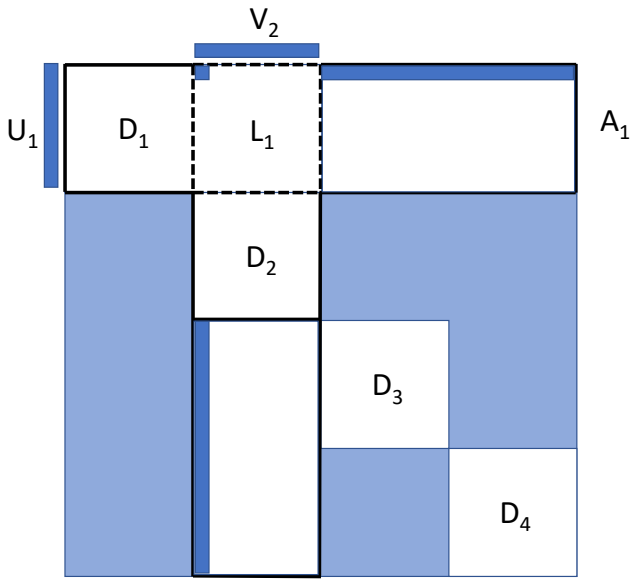


Figure 6.6: HSS matrix illustration of how the off diagonal low rank block L_1 is computed and stored.

which we use to represent the sketch for the Hankel row block and the sketching operator for the Hankel row block in the column basis of L_1 which we use for the computation of the parent node.

6.3.2 Compression of Internal Node

We move on to compressing the second level of the HSS tree whose Hankel row blocks are shown in Figure 6.7. Before we describe the compression of H_5, we explain the **nested basis property** which all internal (non-leaf, non-root) nodes in the HSS tree use. This property explains the hierarchical in HSS matrices.

The nested basis property states that for a non-leaf Hankel block, H_5 with children nodes $H_1,\ H_2$ we can write a row (or column) basis U_5^{big} of dimension $2k \times r$ as a product of the bases of $U_1^{\text{big}},\ U_2^{\text{big}}$ (dimensions $k \times r$) of $H_1,\ H_2$ respectively and a small matrix U_5 of dimension $2r \times r$:

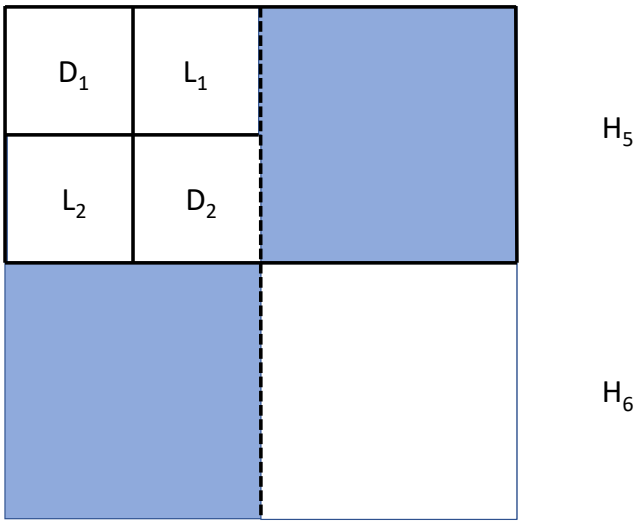


Figure 6.7: HSS matrix with the second level of row Hankel blocks highlighted in blue.

$$U_5^{\text{big}} = \begin{bmatrix} U_1 & 0 \\ 0 & U_2 \end{bmatrix} U_5.$$

Remark 4. *For leaf node i, $U_i = U_i^{big}$.*

The intuition behind this property is that by constructing a basis U_1^{big} for the first k rows and U_2^{big} for the next k rows, when we want to construct a basis U_5^{big} for the $2k$ rows we should be able to use the basis information from our earlier constructions. When constructing HSS matrices we assume that this property holds.

Now that we have the nested basis property we can explain how this reduces the computation for the compression for node 5 (and any internal node) in Algorithm 7. We would like to have a sketch of H_5 depicted in Figure 6.7 and compute U_5, of dimension $2r \times r$. If we consider the matrix $\begin{bmatrix} S_1^{\text{loc}} \\ S_2^{\text{loc}} \end{bmatrix}$ then we have an approximation for the block depicted in the top of Figure 6.8 because when we computed S_1^{loc} and S_2^{loc} we subtracted the diagonal blocks D_1 and D_2 respectively.

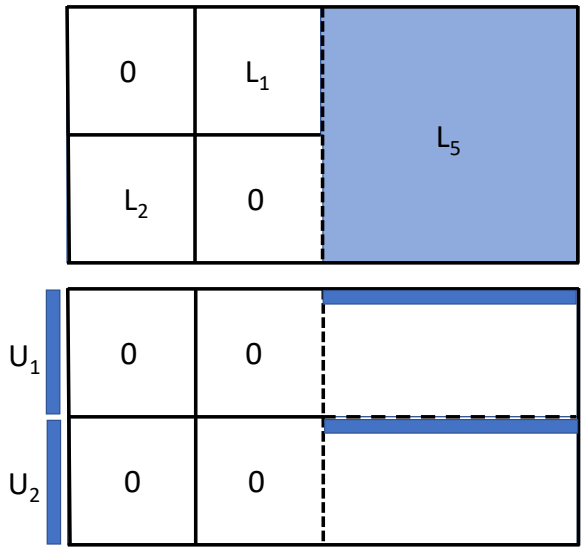


Figure 6.8: Node 5 row Hankel block being prepared for compression.

We show how we use the nested basis property and information from the children nodes to compute a local sketch of H_5. We can subtract our compression of the low dimension blocks $L_1,\ L_2$ which we computed in the children nodes.

$$
\begin{aligned}
S_5 &= \left(\begin{bmatrix} 0 & 0 & H_5(1:k,:) \\ 0 & 0 & H_5(k+1:2k,:) \end{bmatrix} \right) R^T \\
&= \left(A(1:2k,:) - \begin{bmatrix} D_1 & L_1 & 0 \\ L_2 & D_2 & 0 \end{bmatrix} \right) R^T \\
&= A(1:2k,:)R^T - \begin{bmatrix} D_1 & L_1 \\ L_2 & D_2 \end{bmatrix} \begin{bmatrix} R^T(1:k,:) \\ R^T(k+1:2k,:) \end{bmatrix} \\
&= \begin{bmatrix} S_1^{\text{loc}} \\ S_2^{\text{loc}} \end{bmatrix} - \begin{bmatrix} 0 & L_1 \\ L_2 & 0 \end{bmatrix} \begin{bmatrix} R^T(1:k,:) \\ R^T(k+1:2k,:) \end{bmatrix} \\
&\approx \begin{bmatrix} U_1^{\text{big}} & 0 \\ 0 & U_2^{\text{big}} \end{bmatrix} \begin{bmatrix} S_1^{\text{loc}}(J_1,:) \\ S_2^{\text{loc}}(J_2,:) \end{bmatrix} - \begin{bmatrix} U_1^{\text{big}} A(J_1,J_2) V_2^{\text{big}} R^T(k+1:2k,:) \\ U_2^{\text{big}} A(J_2,J_1) V_1^{\text{big}} R^T(1:k,:) \end{bmatrix}.
\end{aligned}
$$

Then,

$$\begin{aligned}
&\begin{bmatrix} U_1^{\text{big}} & 0 \\ 0 & U_2^{\text{big}} \end{bmatrix} \begin{bmatrix} S_1^{\text{loc}}(J_1,:) \\ S_2^{\text{loc}}(J_2,:) \end{bmatrix} - \begin{bmatrix} U_1^{\text{big}} A(J_1,J_2) V_2^{\text{big}} R^T(k+1:2k,:) \\ U_2^{\text{big}} A(J_2,J_1) V_1^{\text{big}} R^T(1:k,:) \end{bmatrix} \\
&= \begin{bmatrix} U_1^{\text{big}} & 0 \\ 0 & U_2^{\text{big}} \end{bmatrix} \left(\begin{bmatrix} S_1^{\text{loc}}(J_1,:) \\ S_2^{\text{loc}}(J_2,:) \end{bmatrix} - \begin{bmatrix} A(J_1,J_2) R_2^{\text{loc}} \\ A(J_2,J_1) R_1^{\text{loc}} \end{bmatrix} \right) \\
&:= \begin{bmatrix} U_1^{\text{big}} & 0 \\ 0 & U_2^{\text{big}} \end{bmatrix} S_5^{\text{loc}}.
\end{aligned} \tag{6.5}$$

Since HSS matrices satisfy the nested basis property to compute a row basis for node 5 we use S_5^{loc} which has dimensions $2r \times l_1$ and contains the nested basis prefactor seen in the second to last row of the above computation which generalizes to any internal HSS tree node. S_5^{loc} corresponds to a sketch of the two dark blue horizontal strips in the bottom of Figure 6.8 and only requires information already computed in the children nodes.

We go through the steps of compressing H_5 using Algorithm 7 and Algorithm 8. First, since node 5 is the parent node of nodes 1 and 2, it stores the small sub-blocks of A used to compute L_1 and L_2 which in this case is $A(J_1, J_2)$ and $A(J_2, J_1)$, by symmetry only storing the $r \times r$ matrix $A(J_1, J_2)$ is required, line 8 of Algorithm 7. Then on line 14 of Algorithm 7 a local sketch S_5^{loc} as in Equation (6.5) is computed using the sub-blocks of A that we just stored and the information in the children nodes. We then check if the local sketch, S_5^{loc}, is sufficient to approximate H_5 and adaptively increase the size of the sketching operator in Algorithm 8 lines 2-10 and lines 15-16. We discuss how this adaptation is done in the following section. Assuming that S_5^{loc} is sufficiently accurate, on line 12 of Algorithm 8 we compute our basis U_5 and row indices J_5 in the nested basis defined by U_1 and U_2. Finally, on line 20 of Algorithm 7 we compute a local sketching operator, R_5^{loc}, in the basis of U_5 which we will use to subtract the block which we have compressed in higher levels of the tree. So we have computed and stored: 1. $A(J_1, J_2)$, 2. S_5^{loc}, 3. U_5, 4. J_5, and 5. R_5^{loc} which are the five components that define an internal node.

We can similarly compress H_6 which would now give us all the information to compress

L_5 and L_6 by symmetry then move up to the root node.

Remark 5. *When compressing the root node we do not do any compression but instead store the two $r \times r$ blocks of A ($A(J_5, J_6)$ and $A(J_6, J_5)$ here) that are required to compute the low rank factorization for the two largest low rank off diagonal blocks (L_5 and L_6 here).*

6.3.3 Adaptation

At each non-root node of the HSS tree we verify that the sketch of our current node, S_i^{loc}, is sufficiently accurate before we compress it. If S_i^{loc} is sufficiently accurate, which is checked by the computation and stopping criteria on lines 2-10 of Algorithm 8 then we can compress node i, otherwise we increase the size of our global sketching operator and global sketch on lines 15 and 16 (from l_1 to $l_1 + \Delta d$ in our example). We then mark the state of the current node, i, as partially compressed and restart our compression loop for all of the nodes.

For the compressed nodes we will update their local sketches and sketching operators to have $l_1 + \Delta d$ instead of just l_1 columns. This operation is computed in Algorithm 7 as follows. First on line 9 we set the columns we will be modifying as the final Δd that we added to the global sketch and sketching operator in Algorithm 8, line 16. Then on lines 12-14 we update the local sketch information, finally on lines 18-21 the local sketching operators are updated.

For the one partially compressed node we will update the sketching operator as for the compressed nodes but we will also check the stopping criteria on lines 7 and 11 in Algorithm 8. If either is met then node i can now be compressed and the algorithm can continue. Otherwise, lines 15-17 will trigger again (in Algorithm 8), expanding the global sketch and sketching operator then marking node j as partially compressed again. Finally, for uncompressed nodes we do not need to update anything, we will use the updated sketching operator and sketches.

This concludes our tracing of Algorithm 7 and Algorithm 8. Our main contribution was the extension of the HSS algorithm described in [GCG19] which leverages Gaussian sketching

operators to a boarder class of Johnson–Lindenstrauss sketches. We hope that by taking a step-by-step approach others will be able to understand the algorithm and extend it or use it in their work. We refer the reader to [YMG23] if they are interested in the theory and experimental results for the generalized HSS construction algorithm.

CHAPTER 7

Conclusion

In this dissertation, we developed novel methods in randomized numerical linear algebra for solving problems at scale. In the first part of this dissertation we discussed variants of the Kaczmarz method which are used to solve large, out of core, systems of equations. Afterwords, in the second part of this dissertation we discussed structured matrix factorizations for both distributed data and structured data.

First we proposed a new variant of the randomized Kaczmarz method to solve large linear systems of equations called selectable set randomized Kaczmarz. This method requires additional space overhead while saving time on computation. We showed that a general selectable set framework can be applied to many Kaczmarz variants. We then proved convergence of the selectable set method highlighting the theoretical improvements over the method which does not leverage a selectable set. Then we experimentally verified this improved performance highlighting that this improvement can be large if the Gramian of the matrix is know apriori. Finally, we highlighted a theory gap in which our method theoretically performs as well as other methods but experimentally this was not always the case.

Next, we developed and analyzed a novel method for solving the online signal recovery problem. This method, online heavy ball Kaczmarz, leveraged both the Kaczmarz method and heavy ball momentum. We proved a linear convergence bound for this method and then experimentally verified its improved performance over the non-heavy ball counterpart on coherent data. We highlighted that in medical and compressed sensing applications the data is often highly coherent emphasizing that this method yields improvements in applications.

Afterwords, we posed a latent class regression problem which we solved via a Kaczmarz method. In this setting there are multiple subgroups that require different treatments (solutions) that are not know apriori. We assume that these subgroups have similar data distributions that are distinct from each other. We are able to simulate this latent class regression problem by scrambling rows of multiple linear systems together. We proposed a Kaczmarz based solution to this latent class problem in which we start with an initial guess for each subgroup and then iteratively sample a single row of our joint matrix. We then only update the solution which lies closest to the row hyperplane. We call this method Multi-Randomized Kaczmarz. After we proposed this solution we proved its convergence and experimentally analyzed its performance on both synthetic and real world data.

Next, we discussed two matrix factorization methods that leverage structure within our matrices. First we proposed stratified non-negative matrix factorization. This is an unsupervised learning method in which we explicitly take into account if data is collected at different times, stored at different locations or contains heterogeneity. This method first separates a matrix into multiple strata and then simultaneously learns strata level statistics and a low rank global representation of the data. This method is also adaptable to the federated setting. After we proposed our objective function, we defined a new method for solving this objective via multiplicative updates. Then, we proved that the stratified NMF objective function is non-increasing under our proposed multiplicative update rules. Finally, we tested our method on synthetic, text, image and tabular data all of which provide interpretable results and highlighted the merits of the method.

Finally, we extended an existing hierarchically semi-separable matrix factorization method which leveraged matrix sketching. The original algorithm used Gaussian sketching matrices and our extension was to use a broader class of Johnson–Lindenstrauss sketching matrices. We were able to show that this broader class of random sketching matrices yield similar construction properties. The benefit of this generalization is the ability to use random sketching matrices which are faster to apply or are tailored specifically to the application.

REFERENCES

[AC06] N. Ailon and B. Chazelle. "Approximate Nearest Neighbors and the Fast Johnson-Lindenstrauss Transform." In *Proceedings of the thirty-eighth annual ACM symposium on Theory of Computing (STOC)*, pp. 557–563, Portsmouth, Virginia, May 2006.

[ASM18] Ehab A AlBadawy, Ashirbani Saha, and Maciej A Mazurowski. "Deep learning for segmentation of brain tumors: Impact of cross-institutional training and testing." *Medical physics*, **45**(3):1150–1158, 2018.

[Ber18] Alan Berube. "Where does the American middle class live." *Brookings Institute Report*, 2018.

[BKW21] Stefan Bamberger, Felix Krahmer, and Rachel Ward. "Johnson-Lindenstrauss Embeddings with Kronecker Structure." *arXiv preprint arXiv:2106.13349*, 2021.

[BLS18] Andrea L Bertozzi, Xiyang Luo, Andrew M Stuart, and Konstantinos C Zygalakis. "Uncertainty quantification in graph-based classification of high dimensional data." *SIAM/ASA Journal on Uncertainty Quantification*, **6**(2):568–595, 2018.

[Bot10] Léon Bottou. "Large-scale machine learning with stochastic gradient descent." In *COMPSTAT*, 2010.

[BW18a] Zhong-Zhi Bai and Wen-Ting Wu. "On greedy randomized Kaczmarz method for solving large sparse linear systems." *SIAM Journal on Scientific Computing*, **40**(1):A592–A606, 2018.

[BW18b] Zhong-Zhi Bai and Wen-Ting Wu. "On relaxed greedy randomized Kaczmarz methods for solving large sparse linear systems." *Applied Mathematics Letters*, **83**:21–26, 2018.

[CJN18] Michael B Cohen, TS Jayram, and Jelani Nelson. "Simple Analyses of the Sparse Johnson-Lindenstrauss Transform." In *1st Symposium on Simplicity in Algorithms (SOSA 2018)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2018.

[CP12] Xuemei Chen and Alexander Powell. "Almost sure convergence of the Kaczmarz algorithm with random measurements." *Journal of Fourier Analysis and Applications*, **18**, 12 2012.

[CYN23] James Chapman, Yotam Yaniv, and Deanna Needell. "Stratified-NMF for Heterogeneous Data." In *2023 57th Asilomar Conference on Signals, Systems, and Computers*, pp. 614–618. IEEE, 2023.

[DG03] Sanjoy Dasgupta and Anupam Gupta. "An Elementary Proof of a Theorem of Johnson and Lindenstrauss." *Random Structures & Algorithms*, **22**(1):60–65, 2003.

[DH11] Timothy A Davis and Yifan Hu. "The University of Florida sparse matrix collection." *ACM Transactions on Mathematical Software (TOMS)*, **38**(1):1–25, 2011.

[DHN17] Jesus A De Loera, Jamie Haddock, and Deanna Needell. "A sampling Kaczmarz–Motzkin algorithm for linear feasibility." *SIAM Journal on Scientific Computing*, **39**(5):S66–S87, 2017.

[DLR77] Arthur P Dempster, Nan M Laird, and Donald B Rubin. "Maximum likelihood from incomplete data via the EM algorithm." *Journal of the Royal Statistical Society: Series B (Methodological)*, **39**(1):1–22, 1977.

[Du19] Kui Du. "Tight upper bounds for the convergence of the randomized extended Kaczmarz and Gauss–Seidel algorithms." *Numerical Linear Algebra with Applications*, **26**:e2233, 2019.

[EK12] Y.C. Eldar and G. Kutyniok. *Compressed sensing: theory and applications*. Cambridge University Press, 2012.

[FR13] Simon Foucart and Holger Rauhut. *A mathematical introduction to compressive sensing*. Birkhäuser Basel, 2013.

[GBH70] Richard Gordon, Robert Bender, and Gabor T Herman. "Algebraic reconstruction techniques (ART) for three-dimensional electron microscopy and X-ray photography." *Journal of Theoretical Biology*, **29**(3):471–481, 1970.

[GCG19] Christopher Gorman, Gustavo Chávez, Pieter Ghysels, Théo Mary, François-Henry Rouet, and Xiaoye Sherry Li. "Robust and Accurate Stopping Criteria for Adaptive Randomized Sampling in Matrix-Free Hierarchically Semiseparable Construction." *SIAM Journal on Scientific Computing*, **41**(5):S61–S85, 2019.

[Gil14] Nicolas Gillis. "The why and how of nonnegative matrix factorization." *Regularization, optimization, kernels, and support vector machines*, **12**(257):257–291, 2014.

[GLZ19] Igor Gitman, Hunter Lang, Pengchuan Zhang, and Lin Xiao. "Understanding the role of momentum in stochastic gradient methods." *Advances in Neural Information Processing Systems*, **32**, 2019.

[GMM21] Robert M Gower, Denali Molitor, Jacob Moorman, and Deanna Needell. "On adaptive sketch-and-project for solving linear systems." *SIAM Journal on Matrix Analysis and Applications*, **42**(2):954–989, 2021.

[GPS18] Sébastien Gadat, Fabien Panloup, and Sofiane Saadane. "Stochastic heavy ball." *Electronic Journal of Statistics*, **12**(1):461–529, 2018.

[GR15] Robert M Gower and Peter Richtárik. "Randomized iterative methods for linear systems." *SIAM Journal on Matrix Analysis and Applications*, **36**(4):1660–1690, 2015.

[GST07] Thomas L Griffiths, Mark Steyvers, and Joshua B Tenenbaum. "Topics in semantic representation." *Psychological review*, **114**(2):211, 2007.

[GTL12] Naiyang Guan, Dacheng Tao, Zhigang Luo, and Bo Yuan. "Online nonnegative matrix factorization with robust stochastic approximation." *IEEE Transactions on Neural Networks and Learning Systems*, **23**(7):1087–1099, 2012.

[GVS17] Klaus Greff, Sjoerd Van Steenkiste, and Jürgen Schmidhuber. "Neural expectation maximization." *Advances in Neural Information Processing Systems*, **30**, 2017.

[GYN22] Erin George, Yotam Yaniv, and Deanna Needell. "Multi-Randomized Kaczmarz for Latent Class Regression." In *2022 56th Asilomar Conference on Signals, Systems, and Computers*, pp. 1367–1371. IEEE, 2022.

[HHM53] Morris H Hansen, William N Hurwitz, and William G Madow. *Sample survey methods and theory. Vol. I. Methods and applications.* John Wiley, 1953.

[HKL20] Jamie Haddock, Lara Kassab, Sixian Li, Alona Kryshchenko, Rachel Grotheer, Elena Sizikova, Chuntian Wang, Thomas Merkh, RWMA Madushani, Miju Ahn, et al. "Semi-supervised nmf models for topic modeling in learning tasks." *arXiv preprint arXiv:2010.07956*, 2020.

[HM93] G.T. Herman and L.B. Meyer. "Algebraic reconstruction techniques can be made computationally efficient (positron emission tomography application)." *IEEE Transactioreconstructionns on Medical Imaging*, **12**(3):600–609, 1993.

[HM21] Jamie Haddock and Anna Ma. "Greed works: an improved analysis of sampling Kaczmarz–Motzkin." *SIAM Journal on Mathematics of Data Science*, **3**(1):342–368, 2021.

[HNR22] J. Haddock, D. Needell, E. Rebrova, and W. Swartworth. "Quantile-based iterative methods for corrupted systems of linear equations." *SIAM Journal on Matrix Analysis and Applications*, 2022.

[IKW16] Arie Israel, Felix Krahmer, and Rachel Ward. "An arithmetic–geometric mean inequality for products of three matrices." *Linear Algebra and its Applications*, **488**:1–12, 2016.

[JL84] William B Johnson and Joram Lindenstrauss. "Extensions of Lipschitz Mappings into a Hilbert Space." *Contemporary mathematics*, **26**:28, 1984.

[JYN22] Benjamin Jarman, Yotam Yaniv, and Deanna Needell. "Online Signal Recovery via Heavy Ball Kaczmarz." In *2022 56th Asilomar Conference on Signals, Systems, and Computers*, pp. 276–280. IEEE, 2022.

[Kar37] S Karczmarz. "Angenäherte Auflösung von Systemen linearer Gleichungen." *Bull. Internat. Acad. Polon.Sci. Lettres A*, pp. 355–357, 1937.

[KB14] Diederik P Kingma and Jimmy Ba. "Adam: A method for stochastic optimization." *International Conference on Learning Representations*, 2014.

[KCP15] Da Kuang, Jaegul Choo, and Haesun Park. "Nonnegative matrix factorization for interactive topic modeling and document clustering." *Partitional clustering algorithms*, pp. 215–243, 2015.

[KKL23] Lara Kassab, Alona Kryshchenko, Hanbaek Lyu, Denali Molitor, Deanna Needell, Elizaveta Rebrova, and Jiahong Yuan. "Sparseness-constrained Nonnegative Tensor Factorization for Detecting Topics at Different Time Scales." *arXiv preprint arXiv:2010.01600*, 2023.

[KN14] D.M. Kane and J. Nelson. "Sparser Johnson-Lindenstrauss Transforms." *Journal of the ACM*, **61**(1), 2014.

[KP08] Jingu Kim and Haesun Park. "Sparse nonnegative matrix factorization for clustering." Technical report, Georgia Institute of Technology, 2008.

[KSH17] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. "ImageNet classification with deep convolutional neural networks." *Commun. ACM*, **60**(6):84–90, 2017.

[KYB19] Jason M Klusowski, Dana Yang, and WD Brinda. "Estimating the coefficients of a mixture of two linear regressions by expectation maximization." *IEEE Transactions on Information Theory*, **65**(6):3515–3524, 2019.

[LCB10] Yann LeCun, Corinna Cortes, and CJ Burges. "MNIST handwritten digit database." *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, **2**, 2010.

[Lin07] Chih-Jen Lin. "On the convergence of multiplicative update algorithms for nonnegative matrix factorization." *IEEE Transactions on Neural Networks*, **18**(6):1589–1596, 2007.

[LL20] Zehua Lai and Lek-Heng Lim. "Recht-Re noncommutative arithmetic-geometric mean conjecture is false." In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 5608–5617. PMLR, 13–18 Jul 2020.

[LM22] James Levitt and Per-Gunnar Martinsson. "Linear-Complexity Black-Box Randomized Compression of Hierarchically Block Separable Matrices." *arXiv preprint arXiv:2205.02990*, 2022.

[LR20] Nicolas Loizou and Peter Richtárik. "Momentum and stochastic momentum for stochastic gradient, newton, proximal point and subspace descent methods." *Computational Optimization and Applications*, **77**(3):653–710, 2020.

[LS99] Daniel D Lee and H Sebastian Seung. "Learning the parts of objects by non-negative matrix factorization." *Nature*, **401**(6755):788–791, 1999.

[LSM21] Yang Liu, Wissam M Sid-Lakhdar, Osni Marques, Xinran Zhu, Chang Meng, James W Demmel, and Xiaoye S Li. "Gptune: Multitask learning for autotuning exascale applications." In *Proceedings of the 26th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming*, pp. 234–246, 2021.

[LWS14] Dirk A. Lorenz, Stephan Wenger, Frank Schöpfer, and Marcus A. Magnor. "A sparse Kaczmarz solver and a linearized Bregman method for online compressed sensing." *2014 IEEE International Conference on Image Processing (ICIP)*, pp. 1347–1351, 2014.

[LZ15] Junhong Lin and Ding-Xuan Zhou. "Learning theory of randomized Kaczmarz algorithm." *J. Mach. Learn. Res.*, **16**(1):3341–3365, 2015.

[LZ18] Yunwen Lei and Ding-Xuan Zhou. "Learning theory of randomized sparse Kaczmarz Method." *SIAM Journal on Imaging Sciences*, **11**(1):547–574, 2018.

[Mar11] Per-Gunnar Martinsson. "A Fast Randomized Algorithm for Computing a Hierarchically Semiseparable Representation of a Matrix." *SIAM Journal on Matrix Analysis and Applications*, **32**(4):1251–1274, 2011.

[Moo96] Todd K Moon. "The expectation-maximization algorithm." *IEEE Signal processing magazine*, **13**(6):47–60, 1996.

[MR10] Sébastien Marcel and Yann Rodriguez. "Torchvision the machine-vision package of torch." In *Proceedings of the 18th ACM international conference on Multimedia*, pp. 1485–1488, 2010.

[MS54] Theodore Samuel Motzkin and Isaac Jacob Schoenberg. "The relaxation method for linear inequalities." *Canadian Journal of Mathematics*, **6**:393–404, 1954.

[MT20] Per-Gunnar Martinsson and Joel A. Tropp. "Randomized Numerical Linear Algebra: Foundations and Algorithms." *Acta Numerica*, **29**:403–572, 2020.

[MTM21] Jacob D Moorman, Thomas K Tu, Denali Molitor, and Deanna Needell. "Randomized Kaczmarz with averaging." *BIT Numerical Mathematics*, **61**(1):337–359, 2021.

[MV04] Jay Magidson and Jeroen K Vermunt. "Latent class models." *The Sage handbook of quantitative methodology for the social sciences*, pp. 175–198, 2004.

[Nat01] Frank Natterer. *The mathematics of computerized tomography*. SIAM, 2001.

[Nee10] Deanna Needell. "Randomized Kaczmarz solver for noisy linear systems." *BIT Numerical Mathematics*, **50**(2):395–403, 2010.

[NN13] Jelani Nelson and Huy L Nguyên. "OSNAP: Faster Numerical Linear Algebra Algorithms via Sparser Subspace Embeddings." In *2013 ieee 54th annual symposium on foundations of computer science*, pp. 117–126. IEEE, 2013.

[NOG20] Alfredo Nazabal, Pablo M Olmos, Zoubin Ghahramani, and Isabel Valera. "Handling incomplete heterogeneous data using vaes." *Pattern Recognition*, **107**:107501, 2020.

[NSL16] Julie Nutini, Behrooz Sepehry, Issam Laradji, Mark Schmidt, Hoyt Koepke, and Alim Virani. "Convergence rates for greedy Kaczmarz algorithms, and faster randomized Kaczmarz rules using the orthogonality graph." In *Proceedings of the Thirty-Second Conference on Uncertainty in Artificial Intelligence*, p. 547–556. AUAI Press, 2016.

[NSW16] Deanna Needell, Nathan Srebro, and Rachel Ward. "Stochastic gradient descent, weighted sampling, and the randomized Kaczmarz algorithm." *Mathematical Programming*, **155**(1-2):549–573, 2016.

[PB97] R Kelley Pace and Ronald Barry. "Sparse spatial autoregressions." *Statistics & Probability Letters*, **33**(3):291–297, 1997.

[PJM21] Vivak Patel, Mohammad Jahangoshahi, and Daniel A Maldonado. "An implicit representation and iterative solution of randomly sketched linear systems." *SIAM Journal on Matrix Analysis and Applications*, **42**(2):800–831, 2021.

[Pol64] Boris T Polyak. "Some methods of speeding up the convergence of iteration methods." *Ussr computational mathematics and mathematical physics*, **4**(5):1–17, 1964.

[PVG11] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. "Scikit-learn: Machine Learning in Python." *Journal of Machine Learning Research*, **12**:2825–2830, 2011.

[Ren08] Jason Rennie. "20 Newsgroups Text Dataset.", 2008.

[RFC12] Richard G Roetzheim, Karen M Freund, Don K Corle, David M Murray, Frederick R Snyder, Andrea C Kronman, Pascal Jean-Pierre, Peter C Raich, Alan EC Holden, Julie S Darnell, et al. "Analysis of combined data from heterogeneous study designs: an applied example from the patient navigation research program." *Clinical Trials*, **9**(2):176–187, 2012.

[Ros64] M Rosenfeld. "Independent sets in regular graphs." *Israel Journal of Mathematics*, **2**(4):262–272, 1964.

[RR12] Benjamin Recht and Christopher Re. "Toward a noncommutative arithmetic-geometric mean inequality: conjectures, case-studies, and consequences." In *Proceedings of the 25th Annual Conference on Learning Theory*, volume 23 of *Proceedings of Machine Learning Research*, pp. 11.1–11.24. PMLR, 25–27 Jun 2012.

[SBP06] Farial Shahnaz, Michael W Berry, V Paul Pauca, and Robert J Plemmons. "Document clustering using nonnegative matrix factorization." *Information Processing & Management*, **42**(2):373–386, 2006.

[Sep16] Behrooz Sepehry. *"Finding a maximum weight sequence with dependency constraints."*. Master's thesis, University of British Columbia, Vancouver, BC, Canada, 2016.

[SHS01] Andreas Savvides, Chih-Chieh Han, and Mani B Strivastava. "Dynamic fine-grained localization in ad-hoc networks of sensors." In *Proceedings of the 7th annual international conference on mobile computing and networking*, pp. 166–179, 2001.

[SL01] H Sebastian Seung and Daniel D Lee. "Algorithms for non-negative matrix factorization." *Advances in neural information processing systems*, **13**:556–562, 2001.

[SMD13] Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. "On the importance of initialization and momentum in deep learning." In *International conference on machine learning*, pp. 1139–1147. PMLR, 2013.

[str] "STRUMPACK: STRUctured Matrix PACKage." `http://portal.nersc.gov/project/sparse/strumpack/`.

[SV09] Thomas Strohmer and Roman Vershynin. "A randomized Kaczmarz algorithm with exponential convergence." *Journal of Fourier Analysis and Applications*, **15**(2):262–278, 2009.

[TV18] Yan Shuo Tan and Roman Vershynin. "Phase retrieval via randomized Kaczmarz: theoretical guarantees." *Information and Inference: A Journal of the IMA*, **8**:97–123, 2018.

[VHR21] Joshua Vendrow, Jamie Haddock, Elizaveta Rebrova, and Deanna Needell. "On a guided nonnegative matrix factorization." In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 3265–32369. IEEE, 2021.

[Wal06] Hanna M Wallach. "Topic modeling: beyond bag-of-words." In *Proceedings of the 23rd international conference on Machine learning*, pp. 977–984, 2006.

[WCD04] Stefan Wild, James Curry, and Anne Dougherty. "Improving non-negative matrix factorizations through structured initialization." *Pattern recognition*, **37**(11):2217–2232, 2004.

[WD94] Michel Wedel and Wayne S DeSarbo. "A review of recent developments in latent class regression models." *Advanced methods of marketing research*, pp. 352–388, 1994.

[WGW15] Di Wang, Xinbo Gao, and Xiumei Wang. "Semi-supervised nonnegative matrix factorization via constraint propagation." *IEEE transactions on cybernetics*, **46**(1):233–244, 2015.

[WLX13] Shen Wang, Xiaoye S Li, Jianlin Xia, Yingchong Situ, and Maarten V De Hoop. "Efficient Scalable Algorithms for Solving Dense Linear Systems with Hierarchically Semiseparable Structures." *SIAM Journal on Scientific Computing*, **35**(6):C519–C544, 2013.

[WSM95] W Wolberg, W Street, and O Mangasarian. "Breast cancer Wisconsin (diagnostic) UCI machine learning repository." *Irvine, CA, USA*, 1995.

[Wu83] CF Jeff Wu. "On the convergence properties of the EM algorithm." *The Annals of statistics*, pp. 95–103, 1983.

[WZ12] Yu-Xiong Wang and Yu-Jin Zhang. "Nonnegative matrix factorization: A comprehensive review." *IEEE Transactions on knowledge and data engineering*, **25**(6):1336–1353, 2012.

[XCG10] Jianlin Xia, Shivkumar Chandrasekaran, Ming Gu, and Xiaoye S Li. "Fast Algorithms for Hierarchically Semiseparable Matrices." *Numerical Linear Algebra with Applications*, **17**(6):953–976, 2010.

[YLL16] Tianbao Yang, Qihang Lin, and Zhe Li. "Unified convergence analysis of stochastic momentum methods for convex and non-convex optimization." *arXiv preprint arXiv:1604.03257*, 2016.

[YMG23] Yotam Yaniv, Osman Asif Malik, Pieter Ghysels, and Xiaoye S Li. "Construction of Hierarchically Semi-Separable matrix Representation using Adaptive Johnson-Lindenstrauss Sketching." *arXiv preprint arXiv:2302.01977*, 2023.

[YMS22] Yotam Yaniv, Jacob D Moorman, William Swartworth, Thomas Tu, Daji Landis, and Deanna Needell. "Selectable set randomized Kaczmarz." *Numerical Linear Algebra with Applications*, p. e2458, 2022.

[Zaf09] Stefanos Zafeiriou. *Algorithms for Nonnegative Tensor Factorization*, pp. 105–124. Springer London, London, 2009.

[ZF13] Anastasios Zouzias and Nikolaos M Freris. "Randomized extended Kaczmarz for solving least squares." *SIAM Journal on Matrix Analysis and Applications*, **34**(2):773–793, 2013.