Team Members: Liam Kelz, Rockwell P. Jackson, Jake Truong-Jones, and Vijval Rajan
Advisor: Rodrigo B. Platte

An Investigation of Kaczmarz Algorithm Variations and Preconditioning

The Kaczmarz method is an iterative algorithm used for solving linear systems of equations ($Ax = b$, where A is an appropriately sized matrix). Introduced by Polish mathematician Stefan Kaczmarz in 1937, the Kaczmarz method has increased in popularity over the recent years due to its flexibility and wide range of application. Due to its efficiency, memory-saving properties, and ability to handle sparse linear systems, the Kaczmarz method and its variants can be used in various different fields such as image and signal processing, tomography, computer graphics, machine learning, geophysics, and generally anywhere large datasets are involved. However, while Kazcmarz methods often present significant advantages, there are important limitations to consider when applying them to specific problems. In our project we sought to explore these limitations and investigate techniques for optimizing and adapting Kaczmarz methods to overcome some of these implicit constraints, thus improving the efficiency and versatility of these methods for real-world application.

In particular, we worked with the block version of the methods where multiple rows are used in each iteration instead of just one. The way in which rows of a linear system are selected and used to update the approximation can affect the speed at which the Kaczmarz Algorithm converges. Rows can be selected randomly, based on the highest magnitude residual entries, $Ax - b$, or some combination of the two. They can then be used to update the approximation by averaging the projections of the current iteration onto all the rows, or projecting the current iteration onto the subspace of the selected rows. All the combinations were used to perform polynomial interpolation on the function $f(x) = e^{x}\sin(x)$. CPU Time and iterations taken were measured against the degree level (the size of the matrix) to see in what situations each version is most effective. Figure 1 shows that while the Targeted version of the algorithms (the one that calculates the full residual vector) is best in terms of iterations taken, it eventually is slower than the Random or Random-Targeted hybrid version because the residual becomes too expensive to calculate, showing the Random selection is best as the size of the system becomes very large. We also see that the Average Block Kaczmarz does better overall than its Projected counterpart because the pseudo-inverse is an expensive calculation that is needed for the Projected version.

High dimensional linear systems can be computationally expensive to solve with iterative solvers particularly when their condition number is large. Preconditioning can help reduce the number of iterations required by transforming the original system into a system which can more easily be solved. But, preconditioning can be a computationally expensive task itself. We explored preconditioning methods which are fast and scalable. Algebraic Multigrid and QR factorizations with sketching were applied to sparse random matrices to see if preconditioning a matrix system can speed up the time to convergence for Kaczmarz solvers. We tested out the Algebraic Multigrid preconditioner on different types of sparse and sparse random matrices to see how matrix structure affected the preconditioner. Algebraic Multigrid preconditioning was effective in reducing the condition number on tridiagonal matrices and gaussian random matrices, but increased the condition number of sparse random matrices. Algebraic multigrid decreased the time to convergence for Kaczmarz methods on tridiagonal and gaussian random matrices. Introduced by Katrutsa and Oseledets, preconditioning Kaczmarz by sketching is an effective method for improving convergence rates. By compressing matrices into smaller sketched matrices we are able to precondition with QR decomposition without the exorbitant computational cost. In this project, we examined the effectiveness of sketching on different types of matrices by adjusting matrix characteristics such as matrix dimension, sketch size and sparsity. Additionally, we experimented with utilizing preconditioning with improved variations of the Kaczmarz method, particularly randomized average block Kaczmarz (RABK). Figure 2 compares five different methods of iterative solvers and shows the benefits of preconditioning through sketching. We were able to demonstrate that our preconditioned RABK method out performed the preconditioned RKM, as well as the Least Squares iterative solver contained in the SciPy library.

Our project also investigated inconsistent systems, adaptive choices of step sizes, and situations with corrupted data sets. These sections are written in more detail on a separate report made by the group that is available upon request (rplatte@asu.edu).

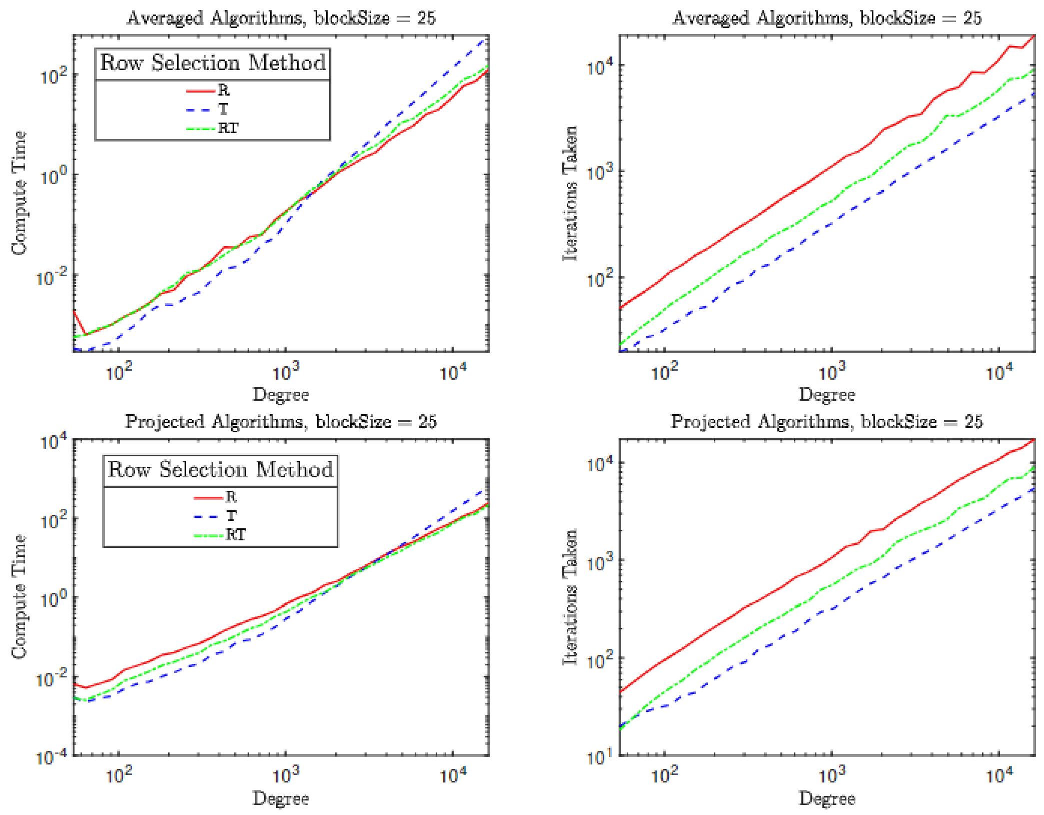

Figure 1: CPU Time in seconds (left) and Iterations Taken (right) against degree level. R denotes Random, T denotes Targeted, and RT denotes Random-Targeted Hybrid row selection.

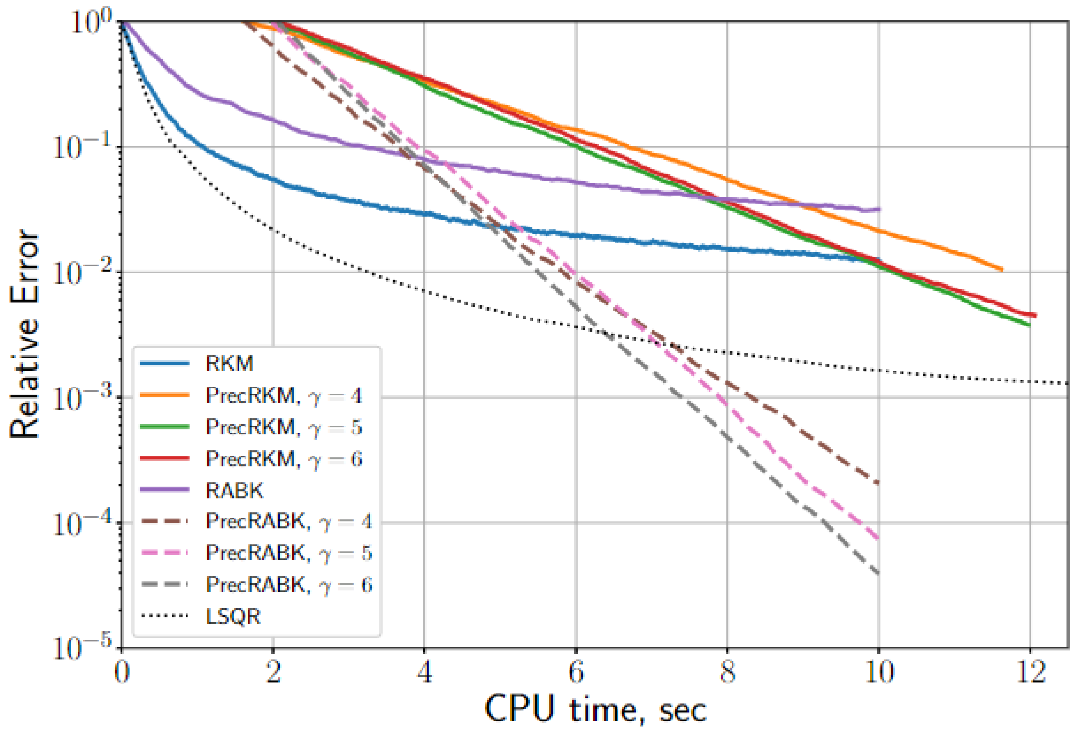

Figure 2: Comparison of convergence speed between sRKM, preconditioned RKM, RABK, preconditioned RABK, and SciPy Least Squares iterative solver on a random matrix.