CONSTRAINED LEAST SQUARES OPTIMIZATIONS:

AN INVESTIGATION OF DAMPED SPECTRAL SQUARE ROOT PRECONDITIONING IN GRADIENT PROJECTION METHODS

Numerical Analysis

Fall 2024

Keenan Powers

Duke University

April 23, 2024

Summary

This paper will give an analysis of Constrained Least Squares Optimizations:

$$\underset{\mathbf{x}\in\mathbb{R}^n}{\text{minimize}}\, f(\mathbf{x}) = \|A\mathbf{x}-\mathbf{b}\|^2 = \sum_{i=1}^{k}(\mathbf{a}_i^T\mathbf{x}-b_i)^2 \qquad (1)$$

$$\text{subject to } q_i \leq x_i \leq s_i$$

where $A \in \mathbb{R}^{k\times n}$, $k \geq n$, $\mathbf{b} \in \mathbb{R}^k$, $\mathbf{a}_i^T$ are the rows of A and q_i and s_i are user-specified bounds on the components of the solution.

The solution will be analytically derived using the Gradient Projection Method with Steepest Descent. The first experiment uses the trivial problem with A as the identity and arbitrarily selected q and s. Its simplicity lends it little interest to us, as it is mostly a means of verifying that the code works. Therefore, the other examples will be investigated in greater detail. Moreover, rather than just focusing on using the gradient projection method, this paper will investigate how preconditioning methods can effect the convergence rate. For this reason, the second involves randomly generated symmetric positive definite, *SPD*, nxn A matrices and investigating the Damped Spectral Square Root, *SQR* preconditioning method. Simulations are then run using randomly generated A preconditioned with the SQR preconditioner at various sizes of A and levels of dampening.

Background and Theory

Gradient Projection Method Overview

When the constraints can be represented as convex sets (such as box constraints), The Gradient Projection Method can be used. In this method, an unconstrained step is taken first, and then the result is projected onto the feasible set defined by the constraints. There is flexibility within this method, particularly in how the unconstrained step is taken, though there is also flexibility in how the unconstrained step is projected back into the feasible set. The general Gradient Projection Method using Steepest Descent is outlined here:

1. Take the Unconstrained Step (in this case, Steepest Descent):

$$g_k = \nabla f(x_k) y_{k+1} = x_k - \alpha_k g_k$$

2. Project the interim solution y_{k+1} onto the feasible set:

$$x_{k+1} = \text{Proj}(y_{k+1})$$

where α_k is the step size, and Proj denotes the projection operation.

Unconstrained Step

While Steepest Descent is the simplest application for the unconstrained step, it has its drawbacks. In particular, it has a slow convergence rate for functions that have a large condition number. We run into this issue for our randomly generated A matrix case. There are three changes we can make to improve our algorithm's convergence time. First, we can improve our starting guess; second, we can precondition the matrix; and third, we can use another unconstrained step method, like Newton or Conjugate Gradients. For our more simple A matrices, such as in Example 1, the Steepest Descent with a random starting point is sufficient. However, in a randomly generated matrix, modifications become useful if not necessary for computational feasibility. For those cases, we will consider preconditioning the matrix. To calculate Steepest Descent step size, the backtracking algorithm provided by Professor Wilkins Aquino will be used.

The Projection Step

In the simplest application of this method, Steepest Descent is used for the unconstrained step, and then the resulting step is "clipped" to fit into the box. This clipping is done by iteratively checking each x entry and taking the upper (or lower) bound if the entry is above (or below) the bound.

$$Proj(x) = Clip(x)$$

Which can be expressed component-wise as:

$$Proj(x_i) = \max(l_i, \min(u_i, x_i))$$

where l_i and u_i are the lower and upper bounds for the i-th component, respectively.

This ensures that each component of $\mathbf{x}$ adheres to the constraints specified by the bounds. The clipping method is sufficient for box constrained Gradient Projection problems. Therefore, the clipping method is used for all problems in this paper.

Preconditioning

In order to scale up the size of our A matrix, we will have to improve our algorithm. One method to do so is to improve the condition number. When A is square, the simplest way is to find a matrix M such that the operation A' = M*A*M improves the condition number of our matrix. We can then likely solve the optimization problem with A' in fewer iterations. The first method attempted in this investigation was the inverted diagonal of A. However, since A is randomly generated, it is unlikely the matrix's eigenvalues are diagonally dominant. Therefore, this often did little to improve the condition number. The second method was to use the Spectral Square Root. Since $A^{-1/2} * A * A^{-1/2} = I$ we dampen the Spectral Square Root, $A_{damped}^{-1/2}$ as follows.

$$M = A_{\text{damped}}^{-1/2} = Q\Lambda_{\text{damped}}^{-1/2}Q^T$$

where Q is a matrix of orthogonal eigenvectors and Λ is a diagonal matrix of eigenvalues, then:

$$\Lambda_{\text{damped}}^{-1/2} = \text{diag}\left(\frac{1}{\sqrt{\lambda_1 + \epsilon}}, \frac{1}{\sqrt{\lambda_2 + \epsilon}}, \ldots, \frac{1}{\sqrt{\lambda_n + \epsilon}}\right)$$

This matrix M, or $A_{\text{damped}}^{-1/2}$ in this case, is used to precondition a system by applying it in a manner that modifies the matrix's condition number. When applied, it effectively scales down the steepness of the eigenvalues, thereby making each direction in the problem space more uniform in terms of the system's response to iterative updates. This uniformity allows for an improvement in Steepest Descent, dampening the yo-yo effect that causes Steepest Descent steps to overshoot in problems with a poorly conditioned matrix. Additionally, the added ϵ allows for an improvement of the condition number without collapsing the problem into a trivial one.

It is important to note that the Dampened Spectral Square Root preconditioning method is very costly. Methods such as Incomplete Cholesky Factorization are less costly. Regardless, the Dampened Spectral Square Root provides a high level of mathematical accuracy, making it a suitable candidate for the study of how dampening affects convergence rates in the Gradient Projection with Steepest Descent Method.

1 Results

Example 1

This example, where A is the identity matrix, condenses our problem into the simple one of finding each x_i closest to its b_i counterpart whilst still being feasible. In the unconstrained case, the answer is simply x = b. In the constrained case,

$$x_i = \begin{cases} q_i & \text{if } b_i < q_i, \\ b_i & \text{if } q_i \leq b_i \leq s_i, \\ s_i & \text{if } b_i > s_i. \end{cases}$$

This can be easily verified computationally by the reader.

Example 2

$$\underset{\mathbf{x} \in \mathbb{R}^n}{\text{minimize}} \, \|\mathbf{Ax} - \mathbf{b}\|^2$$

subject to:

$$5 \leq x_i \leq 15 \quad \text{for all } i = 1, 2, \ldots, n$$

for randomly generated $n \times n$ SPD matrix **A** and likewise randomly generated n-dimensional vector **b**. Since **A** and **b** are randomly generated, our best initial guess is simply in the center of our box. $x_i^{init} = \frac{q_i + s_i}{2}$.

2 Results

Data

Computationally solving Example 1 verified what has already been shown algebraically. For Example 2, First, 50 SPD matrices are randomly generated for each n value of 10, 20, 50, and 100. The data from these trials is then collected for analysis.

First, an average number of iterations for various sizes of n is generated. As you can see in Figure 1a, the number of iterations by size of the matrix is roughly quadratic. One may notice there is a

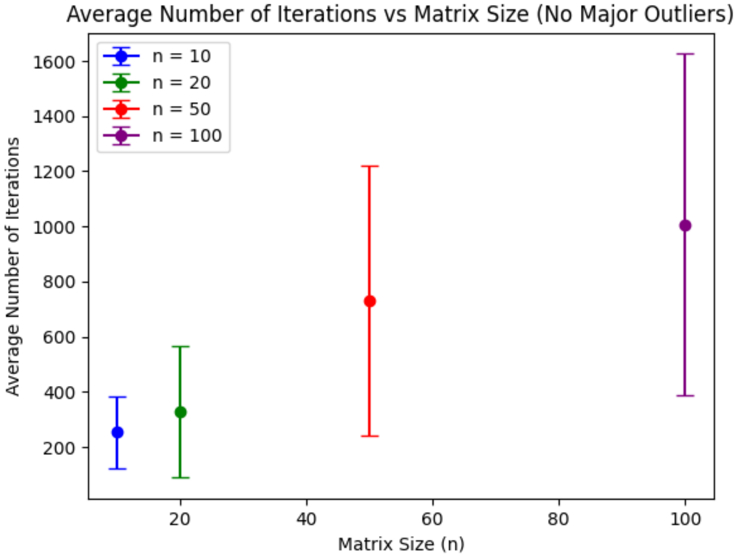
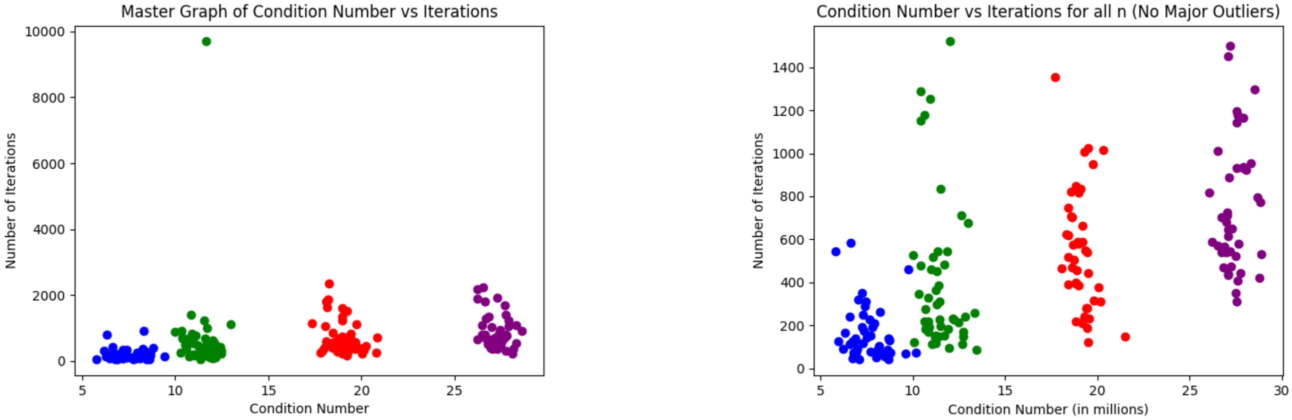


Figure 1: Average Number of Iterations vs Matrix Size without the Major Outlier in the n = 20 case

large relative variation for size n = 20. This is because of a singular outlier point. Figure 1b takes out the outlier.

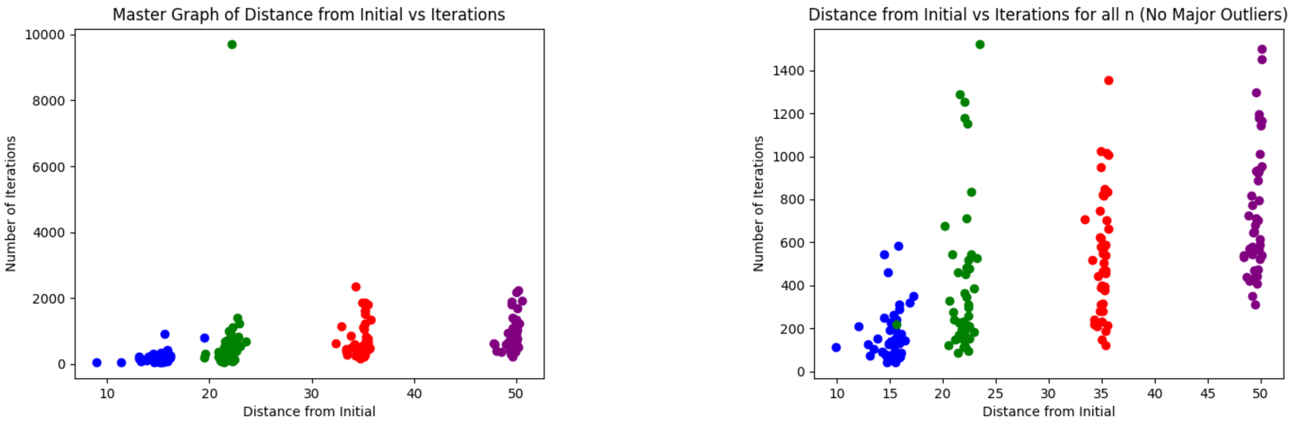


(a) Condition Number of Matrix A vs Number of Iterations

(b) Condition Number of Matrix A vs Number of Iterations without the Major Outlier)

Figure 2: Plot of iteration counts by condition number both with (a) and without (b) the major outlier

Additionally, the condition number of A and the corresponding number of iterations to solve the associated Example 2 is investigated. Figure 2 does not show any significant correlation, which is surprising. However, upon further investigation of the graph, one will see that the condition number is in the millions for each matrix. Since the condition number is so high for all matrices, there is relatively low variation. Therefore, the condition number is not likely to be correlated strongly with the number of iterations. This high condition number is likely due to the fact that A is randomly generated.



(a) Condition Number of Matrix A vs Number of Iterations

(b) Final Distance from Starting Point vs Number of Iterations without the Major Outlier)

Figure 3: Distance from the final x value, $x*$, from the initial starting guess x_o both with (a) and without (b) the major outlier. The starting guess for all trials was at the center of the box

The graphs in Figure 3 show compare the number of iterations to the distance from the initial starting point (the center of the box) to the x value at the calculated minimum. There is a statistically significant positive correlation between distance and the number of iterations.

Next, the Damped Spectral Square Root Preconditioning Method is used. Simulations. Data is first collected for the first 10 matrices from each size n using dampening scale values 0.1, 0.25, 0.5, 0.75, and 1.

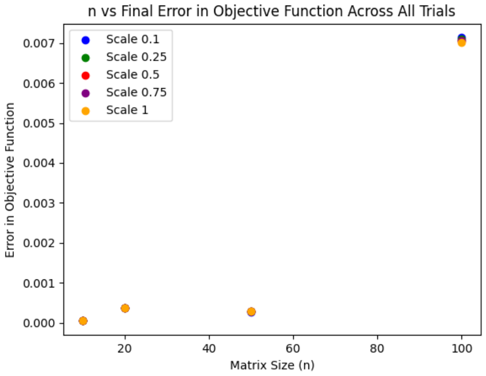
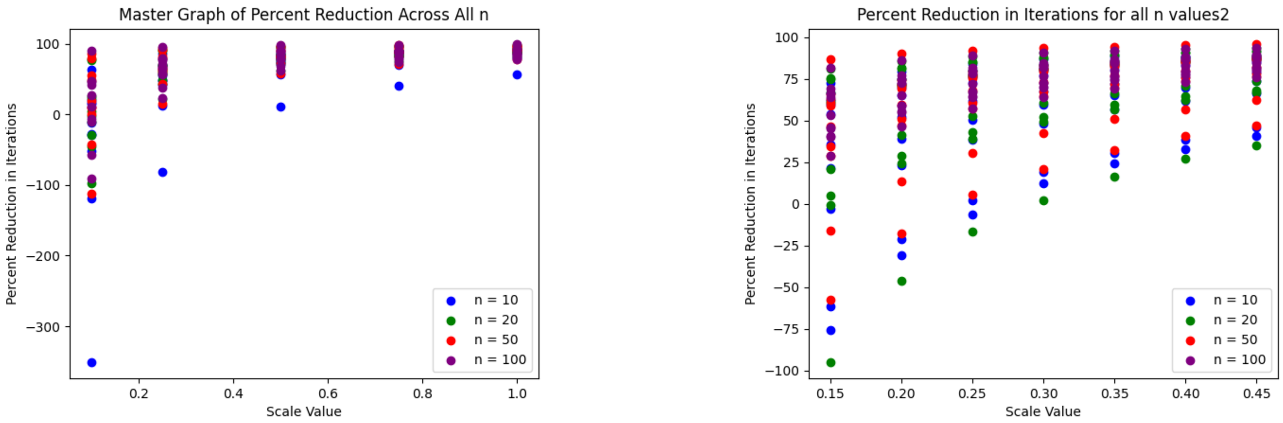


Figure 4: Matrix Size vs Error in Objective Function

First, the error is measured using the value calculated in the unmodified case. As one can see in Figure 4, though the error seems to increase with matrix size, it remains very low. Therefore, the small differences between the calculated values are likely due to machine error.



(a) Condition Number of Matrix A vs Number of Iterations

(b) Final Distance from Starting Point vs Number of Iterations without the Major Outlier)

Figure 5: Distance from the final x value, $x*$, from the initial starting guess x_o both with (a) and without (b) the major outlier. The starting guess for all trials was at the center of the box

Thus, after verifying that the preconditioned methods are calculating the correct values, the graphs in Figures 5 measure their effect on reducing the number of iterations in the Gradient Projection Method. The graph shows a logarithmic relation between the dampening value and the number of iterations taken. It also suggests an optimum value since, as seen in the 0.1 cases for all sizes n, too low of a scale value has the opposite of the intended effect: increasing the number of iterations rather than decreasing it. This observation suggests there is an optimum scale value that minimizes the expected number of iterations. This optimal scale value is inversely related to the size of A.

Conclusions

Overall, this experiment shows that the Damped Spectral Square Root can be an effective preconditioner for reducing the number of iterations required in the Gradient Projection with the Steepest Descent method to solve the minimization problem posed in this paper. However, it depends on the scale value used. The optimal scale value for reducing the number of iterations is inversely related to the matrix size.

For future projects, other faster preconditioning methods (i.e., Incomplete Cholesky Factorization, Symmetric Successive Over Relaxation) should be investigated. Additionally, an investigation comparing different unconstrained steps (i.e., Newton, Quasi-Newton, Conjugate Gradients) could yield valuable insights into optimizing the Gradient Projection Method for the problem presented in this paper.

References

[1] N. Doikov, S. U. Stich, and M. Jaggi. Spectral preconditioning for gradient methods on graded non-convex functions. *Preprint*, 1(1), 2021.

[2] J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer, 2006.

[3] R. Wilkinson. Multivariate statistics. `https://rich-d-wilkinson.github.io/MATH3030/3.2-spectraleigen-decomposition.html`, 2024. Accessed: 2024-04-23.