Math 612- Computational Methods Final Project Report

Comparative Analysis of Hessian Modification Strategies in Unconstrained Minimization

Juneeth Kumar Padarti

ABSTRACT :

This report explains a brief summary of the Comparative Analysis of Hessian Modification Strategies in Unconstrained Minimization.The whole point is to provide a comprehensive comparison of three distinct Hessian modification strategies applied to a variety of benchmark optimization problems.The goal was to determine which strategy(1, 2 or 3) was efficient in making the Hessian matrix a positive definite based on the number of iterations and the function evaluations it took to converge to the global minimum values of x. The test functions used were 11 namely: Rosenbrock[n variables], Beagle[2], Matyas[2], Powell singular function [4], Sphere[n], Booth[2], Styblinski-Tang function[n], McCormick[2], Easom[2], Wood function[4], Helical valley function[3] encompassing a wide range of complexities, dimensions and algorithmic parameters.

METHODOLOGIES :

This optimization code uses Newton's method, a powerful iterative technique in numerical optimization to find minima of the functions defined. This method uses the gradient(1st order derivative) and Hessian(the 2nd-order derivative matrix) to update its estimates. In Newton's method, the update rule is Xnew= Xold - H^−1·∇f, where H is the Hessian matrix at the current estimate and ∇f is the gradient. The power of Newton's method is in large part due to assuming that the Hessian is positive definite, i.e. all eigenvalues are positive. If the hessian fails this test, the method may converge very poorly or not at all.

To overcome this problem, three methods to touch the matrix in order to guarantee the positive definite Hessian are:

Strategy 1: Eigenvalue Shifting

This strategy begins by checking if the minimum eigenvalue (EV) of the Hessian is at least a small fraction (denoted as delta, set to 1e-5) of the maximum eigenvalue. If this condition is not met, a variable tau is calculated as the difference between the maximum eigenvalue and a scaled minimum eigenvalue, divided by a predefined condition number (in this case, max_num = 2). This tau is then added to all eigenvalues to shift them upwards, ensuring that the minimum eigenvalue is adjusted adequately above zero. The resulting eigenvalues are used to construct the modified Hessian matrix which is guaranteed to be positive definite.

Strategy 2: Diagonal Adjustment and Cholesky Factorization

In the second strategy, the focus is on the diagonal elements of the Hessian matrix. The minimum diagonal value is identified, and if it is not positive, an adjustment scalar (tau) is calculated as the difference between a small positive constant mu and this minimum value. The Hessian matrix is then adjusted by adding tau times the identity matrix to it, forming H_adj. This matrix undergoes Cholesky factorization, which is only feasible if the matrix is positive definite. If the factorization fails, tau is increased (either doubled or set to mu, whichever is larger), and the process is repeated until successful factorization is achieved. The successfully factorized matrix confirms the positive definiteness. The search direction is found using the Cholesky, by solving y and p (search direction).

Strategy 3: Eigenvalue Clamping

The third strategy involves setting a lower bound on the minimum eigenvalue relative to the maximum eigenvalue, using the ratio defined by max_num. This method clamps the minimum eigenvalue to this bound, ensuring that none of the eigenvalues are excessively small or negative relative to the maximum eigenvalue. The adjusted eigenvalues, which ensure a more stable and reliable Hessian matrix, are then utilized in the optimization process. Each of these strategies is designed to handle the Hessian matrix in Newton's method, ensuring that the optimization algorithm remains robust and converges effectively to the desired solution.

We check for convergence during the optimization by looking at the norm of the gradient that essentially measures how far we are from a zero-valued gradient. If this norm falls below a given tolerance, the process is halted and it is said that a local minimum has been found, or if it becomes too small the step cannot be correctly made in which case there can not be further progress. Furthermore, to avoid infinite calculations the process will halt after a certain given number of iterations.

Firstly, the Hessian matrix is symmetrized. This ensures that the Hessian matrix is perfectly symmetric.Depending on the mod_flag, the code follows different strategies to compute the search direction:

If mod_flag == 2, the code directly utilizes a strategy function 2 that adjusts the Hessian matrix to ensure it is positive definite and then uses the cholesky factorization of the modified Hessian to compute the search direction.

For mod_flag == 1 or mod_flag == 3, the strategy function 1 or 2 is called to adjust the eigenvalues and the eigenvectors of the Hessian matrix. These eigenvalues and eigenvectors are used to ensure the Hessian matrix is positive definite. The search direction is then computed using a formula that involves matrix-vector multiplication: -eigenvectors @ ((eigenvectors.T @ grad) / eigenvalues). This expression computes the inverse of the modified Hessian in a numerically stable manner using its eigen-decomposition and then multiplies it by the negative gradient.

Then, in each iteration, the line search algorithm then tunes the step size, which is the amount in this direction that you should move to progress at each iteration by striking a balance between progression and stability. This step size is great because a large step can pass the minimum, and too small of one might not allow you to get there.

Results for each strategy tested are recorded along with the number of iterations taken, the total number of function evaluations, and the final gradient norm, and can be displayed if specified by the display flags. This detailed tracking is not only meant to guarantee clarity in the process of optimization but also to explicitly appraise the efficiency of each strategy when dealing with a complex landscape of functions. This methodology emphasizes a disciplined approach toward optimization, specially designed to fully exploit Newton's method while reducing its typical pitfalls of Hessian modification.

RESULTS AND DISCUSSION :

These optimization methods tested across different mathematical functions reveal varying degrees of efficiency and effectiveness, as evidenced by their ability to approach optimal values, number of iterations, and function evaluations required. The analysis focuses on comparing the performance of three strategies (Strategy 1, Strategy 2, and Strategy 3) applied to each function. Some are mentioned below,

Rosenbrock Function:

Showed significant convergence differences among the strategies. While all strategies converged to near-optimal solutions, the iterations and function evaluations varied, indicating different efficiency levels.

Rastrigin Function:

Known for its complex, multimodal landscape, posed a challenge, yet strategies consistently managed to reach low gradient norms quickly, suggesting effectiveness in escaping local minima.

Raydan2 Function:

Displayed rapid convergence in terms of gradient norm across all strategies, highlighting the function's simpler landscape or the strategies' effectiveness in handling its specific challenges.

Sphere Function:

As a simpler function with a global minimum that is easy to locate, all strategies performed exceedingly well, reaching the minimum in just a few iterations.

Styblinski-Tang Function:

This function showed variation in the number of iterations required, which reflects the strategies' ability to deal with a function that has multiple local minima and a complex landscape.

Functions like the **Matyas, McCormick, Booth and Easom** exhibited straightforward, allowing all strategies to converge rapidly and effectively, demonstrating that simpler optimization problems do not heavily depend on the choice of strategy.

On the other hand, **Powell singular** did extremely better with particularly Strategy 2, outperforming others by achieving convergence with fewer iterations and evaluations. This implies that for complex functions with steep gradients or isolated peaks, precision in step size and the capability to adjust to the function's demands critically influence the optimization success.

Additionally, functions such as the **Wood and Helical Valley function**, revealed a need for more nuanced or adaptive approaches as all strategies quickly reached step tolerance, indicating potential areas for strategy refinement to better explore and optimize in complex scenarios.

Insights and Observations:

Efficiency: Strategy 1 tended to require more iterations and function evaluations across most functions compared to Strategies 2 and 3, suggesting that it may be using a more conservative approach, which, while stable, is less efficient.

Adaptability: Strategy 2 appears to be the most adaptable across different types of functions, often achieving the lowest gradient norms and requiring fewer iterations and function evaluations. This suggests a potential focus on adaptive techniques within this strategy that tailor the approach based on the function's landscape.

Robustness: Strategy 3, although sometimes faster, did not consistently outperform Strategy 2 in terms of reaching a lower gradient norm, which implies that the modifications or parameters used in Strategy 2 might strike a better balance between aggressiveness and stability.

Overall, from the grouped performance across these functions In my opinion, the testing highlights the importance of choosing the right optimization strategy based on the function's characteristics and the specific goals of the optimization process (e.g., speed vs. stability). Strategy 2 emerges as generally the most effective across the tested functions, offering a good compromise between rapid convergence and handling of complex function landscapes.

Table 1: Optimization Results Summary

Function Name	Best Strategy	Iterations	Function Evaluations	Final Gradient Norm	Reason for Stopping
Rosenbrock Function	Strategy 1	1000	5896	4.19×10^{-5}	Maximum iterations reached
Rastrigin Function	Strategy 2	3	48	2.77×10^{-8}	Step tolerance reached
Raydan2 Function	Strategy 1, 2	9	52	4.20×10^{-10}	Gradient tolerance reached
Beale Function	Strategy 1	28	147	2.19×10^{-9}	Gradient tolerance reached
Matyas Function	Strategy 1, 3	2	5	1.26×10^{-17}	Gradient tolerance reached
Powell Singular Function	Strategy 2	22	127	4.76×10^{-9}	Gradient tolerance reached
Sphere Function	Strategy 1, 3	2	5	0.0	Gradient tolerance reached
Booth Function	Strategy 1, 2, 3	2	5	3.97×10^{-15}	Gradient tolerance reached
Styblinski Tang Function	Strategy 3	12	87	7.11×10^{-14}	Gradient tolerance reached
McCormick Function	Strategy 1, 2, 3	4	13	6.58×10^{-13}	Gradient tolerance reached
Easom Function	Strategy 1, 2, 3	3	12	6.57×10^{-13}	Gradient tolerance reached
Wood Function	Did not converge	1	15	4.0	Step tolerance reached
Helical Valley Function	Did not converge	1	15	66.53	Step tolerance reached