Effectiveness Of Three Scalar Optimization Methods

***Abstract* - The objective was to test the effectiveness of three different optimization methods used to estimate local minimizers between the brackets of three unique functions. The three optimization methods investigated were quadratic approximation given a single point, fixed stepsize search, and backtracking line search. The effectiveness was determined by comparing the estimated minimizers to the actual minimizers, along with how many iterations it took for each method. The most effective method would be the method which got an estimated value close to the minimizer in the least amount of iterations. The resulting MATLAB code output ran without error or warning and was confirmed to be correct by Emma Ritcey. The results showed that one step of quadratic approximation was the most inaccurate when comparing the estimate and the actual minimizer of each function. As well when comparing fixed stepsize search to backtracking line search both achieved identical estimations equal to each function's actual minimizer. Backtracking line search though was able to complete the estimations in less iterations. Backtracking line search proved to be the most accurate and most efficient of the three methods evaluated.**

INTRODUCTION

The objective was to estimate the local minimizer between two specific values using three different approximation techniques for three functions. The three techniques investigated were one step of quadratic approximation, approximation with a fixed stepsize, and backtracking line search. In order to estimate the minimizers, the functions and approximation techniques were implemented in MATLAB.

A local minimizer is defined as a vector where if the function is evaluated at any other vector nearby, the value will be greater or equal to the value at the minimizer. More formally, this can be written as: for any w Rn and any f:RnR , a vector t* Rn is a local minimizer of *f*, which can be defined as rR++(||w-t*||<r)(f(t*)f(w)). The first technique investigated was quadratic approximation using a single point, the first derivative, and the second derivative. Approximation methods can be used on difficult functions where one or more estimates of the minimizer are known. The methodology is that a simple enough model can have its local minimizer easily found, and the minimizer for the model can be used as a new estimate for the minimizer of interest. The second technique investigated was a fixed stepsize search. This optimization method takes a prescribed step value and uses line search to compute a new estimate of the minimizer. The final technique evaluated was approximation using backtracking line search. Similar to fixed stepsize search, an initial stepsize is provided, but in this case the assumption is made that the provided stepsize may be too large, but never too small. The idea of backtracking is that the user supplied stepsize is exponentially backed off, which also exponentially reduces one side of the inequality constraint.

The underlying scientific question is to determine which optimization technique proved most effective in approximating the function’s local minimizer. This will be determined by analyzing the difference between the estimated minimizer and the actual minimizer, along with how many iterations of the algorithm it took to generate this estimate.

METHODS

In order to begin comparing techniques, the functions of interest needed to be loaded into MATLAB along with their first and second derivatives. For each function, starting points were determined by evaluating the function at both bracket values and taking the larger of the two results.

The first technique evaluated was quadratic approximation given a singular point. This was done by generating a Vandermonde matrix, which lets us solve for the function’s coefficients. The Vandermonde matrix can be separated into three matrices, a 3 x 3 matrix where each entry is based on the equation for a second order polynomial, a 3 x 1 matrix where each entry represents a coefficient, and a 3 x 1 matrix representing the function values. Because only a singular point was provided, the second row of the 3 x 3 matrix will be equal to the first derivative of the second order polynomial and the third row will be equal to the second derivative. To get the unknown coefficients, a matrix left division is performed between the 3 x 3 matrix and the 3 x 1 matrix of function values. The stationary point approximation is then evaluated using the equation -coefficient_matrix(2)/(2*coefficient_matrix(1)). Only one iteration of quadratic approximation was completed.

The second technique evaluated was approximation using a fixed stepsize search. The stepsize was calculated by multiplying 1/100 by the width of the function brackets. To begin, both the function and gradient were evaluated at the initial estimate. The direction was then calculated by taking the negative value of the gradient. A new value is estimated by adding the current estimate to the stepsize, multiplied by the direction. The function and gradient are then re-evaluated at this new estimate and the algorithm iterates until the minimizer is found, or the iteration count equals 500.

The third technique evaluated was approximation with backtracking line search. The initial stepsize was evaluated to be 1/10 of the width of the brackets. Also, both beta and alpha were set to 0.5 by default. To begin, the algorithm, the function, and gradient are evaluated at the initial estimate. The direction is then calculated by taking the negative gradient. A single step is then evaluated from the initial point. Alpha is set equal to the gradient at the estimate divided by 2. The new estimate is now compared to the comparison value, which is equal to the function at the initial estimate plus alpha multiplied by the stepsize multiplied by the direction vector. If the function at the estimate is greater than the comparison value, backtracking begins and the stepsize gets scaled by beta. The function is then re-evaluated at the estimate using the new stepsize. This iteration continues until the function value at the new estimate is less than the comparison value. Once this comparison evaluates to false, a new step is taken and the function and gradient are re-evaluated. The direction is calculated at this new step and the algorithm loops through again until the minimizer is found or the iteration count equals to 50000.

The results were evaluated by comparing the difference between the estimated minimizer and the actual minimizer for each function, as well as the number of iterations needed. The most effective method will have the lowest difference between the estimated minimizer and the actual minimizer along with a low iteration count.

RESULTS

Table 1: The following table contains the results from each optimization technique: one step of quadratic approximation, fixed stepsize search, and backtracking line search, on three distinct functions, *f1*, *f2*, and, *f3*. The approximated minimizer along with the number of iterations are displayed in brackets.

Method	**Function (iterations)**		
	f1	*f2*	*f3*
Quadratic	0.679 (1)	9.830 (1)	6.566 (1)
Fixed	0.241 (43)	8.624 (500)	2.707 (38)
Backtracking	0.241 (13)	8.624 (42)	2.707 (6)

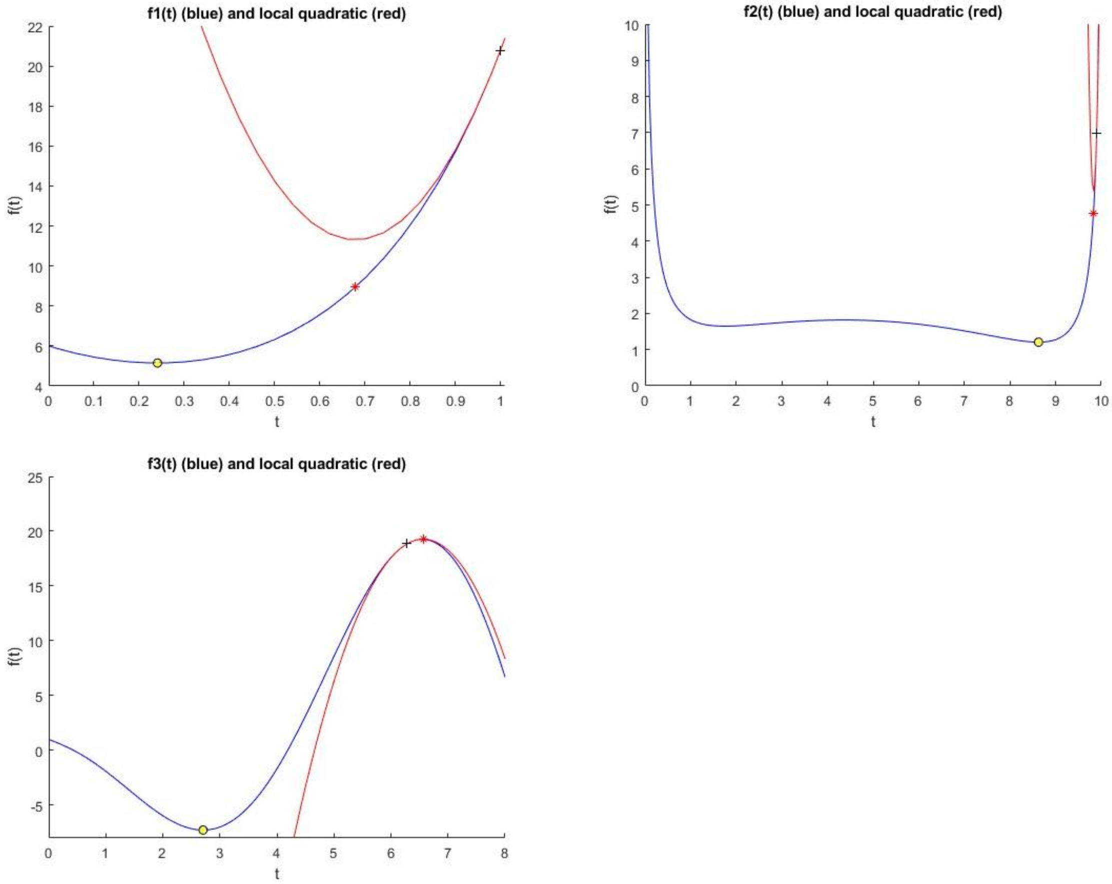

Figure 1: Graphical display of the functions of interest, in blue, along with the initial estimate as a black “+”. A quadratic approximation is shown as a red curve and a single step of quadratic approximation is shown as a red asterisk. The green and yellow asterisk represent the estimation after fixed step approximation and backtracking line search respectively. The black “O” indicates the actual local minimizer for each function.

DISCUSSION

By observing **Figure 1**, it can be seen that quadratic approximation has the highest level of deviation from the actual minimizer of each function. Quadratic approximation's best result was for function one, where the difference between the estimation and the actual minimizer was 0.438, whereas the difference was 1.206 and 3.860 for functions two and three respectively **(Figure 1)**. Both fixed step search and backtracking line search were able to accurately estimate the correct local minimizer for each function. Between the two, fixed step search took 43, 500, and 38 iterations whereas backtracking line search took 13, 42, and 6 for functions one, two, and three respectively **(Table 1).**

The increased accuracy of quadratic approximation on function one could be attributed to the fact that quadratic approximation is useful for minimizing functions that are convex [1]. Of the functions evaluated, only function one is a convex function, so it is logical that it was quadratic approximation’s most accurate result. For all three functions, quadratic approximation did not accurately estimate the minimizer compared to fixed stepsize search and backtracking line search. **Table 1** shows that both fixed stepsize search and backtracking line search achieved identical estimations, but backtracking line search achieved it in less iterations. It is possible that because the stepsize is fixed in fixed stepsize search, oscillation occurred over the local minimizer causing the iteration count to increase. Using fixed stepsize search on function two displays this behavior as the maximum number of iterations, 500, was reached (**Table 1).** This was not the case with backtracking line search as it was able to determine the estimated minimizer in 42 iterations and achieve an identical result of 8.624 **(Table 1)**.

After interpreting the results it can be concluded that the most effective optimization technique for estimation of a local minimizer is backtracking search, as it was both the most accurate and most efficient method of the three. Fixed step size search was equally as accurate, but required more iterations compared to backtracking line search, resulting in lower efficiency. Quadratic approximation was not capable of accurately estimating the functions' minimizers after one iteration.

REFERENCES

[1] Cp, Lee, and S.J., Wright. "Inexact Successive quadratic approximation for regularized optimization." *Computational Optimization and Applications,* vol. 72, pp. 641–674. Jan 25 2019.