Mark Paluta
AA222 Multidisciplinary Optimization

Project 1: Unconstrained Optimization

1. Methodology

Newton's method was implemented as the optimization algorithm of choice. This is a second-order method, using information from both the gradient and the Hessian matrix. The way this method works is by fitting a quadratic to the initial test point, finding the minimum value, and calculating the input vector that gives that minimum value. Then this process is iterated using this new vector from the quadratic minimum as the test point.

This methodology required two additional methods written from scratch, since a calculus package is not to be used. These two methods are the gradient and Hessian computations. The gradient can be calculated with a simple approximation from calculus concepts, with a function evaluation on either side of the test point to estimate the slope. The Hessian calculation is slightly more complicated, so the Wikipedia article on "Finite Difference" was referenced to obtain the correct formulae. It too involves incremental small steps in several directions from the test point.

Both the gradient and Hessian require several evaluations of the test function. These evaluations are kept track of in a global value passed between the functions, so that the Newton's method algorithm can halt evaluation once a specified maximum number of evaluations has been exceeded.

2. Algorithm Path for Example Function

To provide a visual example of the algorithm execution, a plot of contours of Rosenbrock's function is shown below in Figure 1 (the function is explicitly written out in section 3). Superimposed on the contours are three paths of convergence, starting at three unique test points.

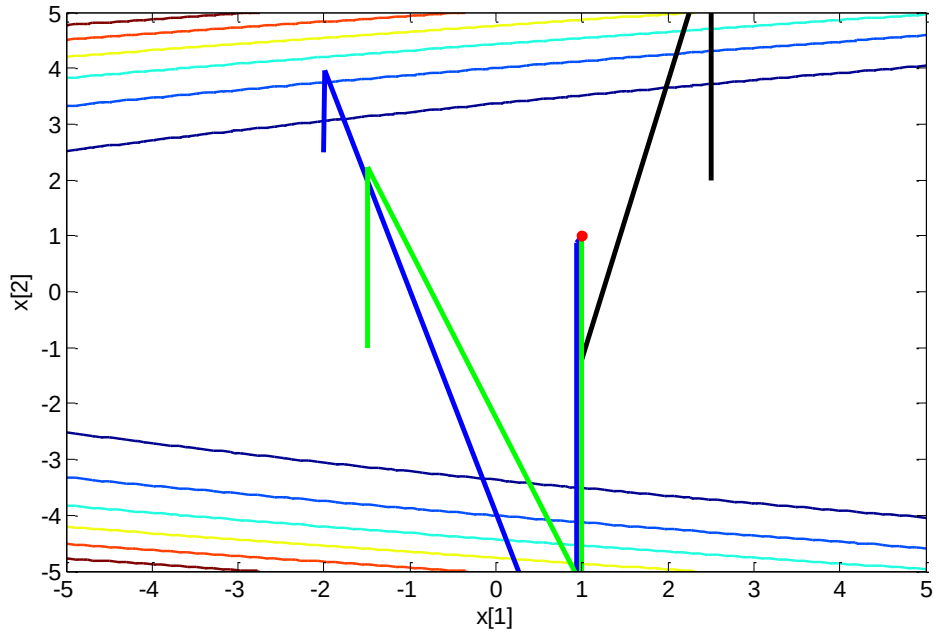

Figure 1. Rosenbrock's function and three paths of convergence.

The three points chosen for the demonstration are (2.5, 2), (-2, 2.5), and (-1.5, -1). With 6 iterations or less, each has converged to the solution of f(1,1) = 0 with an error less than 10^{-5}. This corresponds to less than 100 function evaluations.

3. Convergence Data for Example Functions

This algorithm is tested on three sample functions, including Rosenbrock's function above. These functions are explicitly stated below.

Rosenbrock's function:

$$f = 100(x_2 - x_1^2)^2 + (1 - x_1)^2$$

$$x^* = (1, 1)^T, f^* = 0$$

Wood's function:

$$f = 100(x_2 - x_1^2)^2 + (1 - x_1)^2 + 90(x_4 - x_3^2)^2 + (1 - x_3)^2 + 10.1[(x_2 - 1)^2 + (x_4 - 1)^2] + 19.8(x_2 - 1)(x_4 - 1)$$

$$x^* = (1, 1, 1, 1)^T, f^* = 0$$

Powell's function:

$$f = (x_1 + 10x_2)^2 + 5(x_3 - x_4)^2 + (x_2 - 2x_3)^4 + 10(x_1 - x_4)^4$$

$$x^* = (0, 0, 0, 0)^T, f^* = 0$$

For Rosenbrock's function, a starting point of (2, 2) was used. The convergence is shown in Figure 2 below.

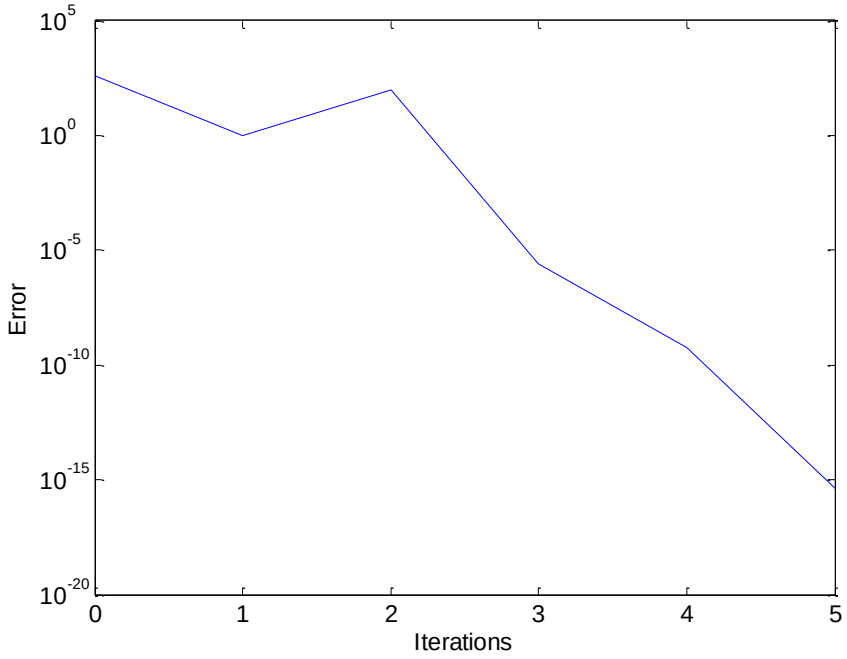

Figure 2. Rosenbrock's function convergence.

It is interesting to note that this algorithm increases its error on its second iteration. After five iterations, it has an error of less than 10^{-15}. This example used less than 100 function evaluations.

The second function is Wood's function, and a test point of (2, 2, 2, 2) was used. The algorithm is less efficient at converging here compared to the first example, as can be seen in the figure below.

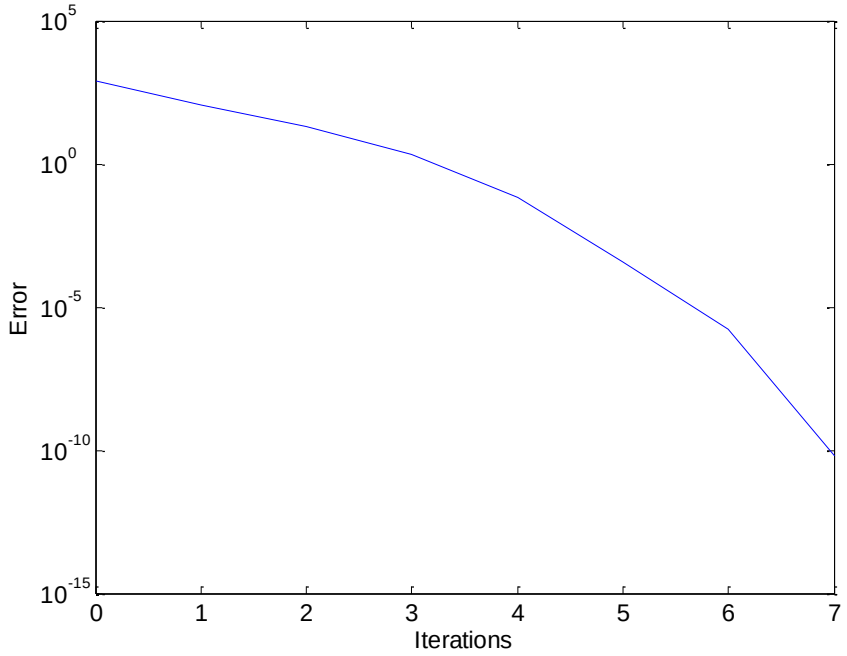

Figure 3. Wood's function convergence.

Seven iterations, utilizing less than 500 function evaluations, brought the solution within 10^{-10} of the global minimum. The output was exclusively decreasing in this example, unlike the first.

The final function evaluated was Powell's function. Again, (2, 2, 2, 2) was used as the test point. This function sported the lowest rate of convergence of the three with this algorithm. The plot below shows this data.

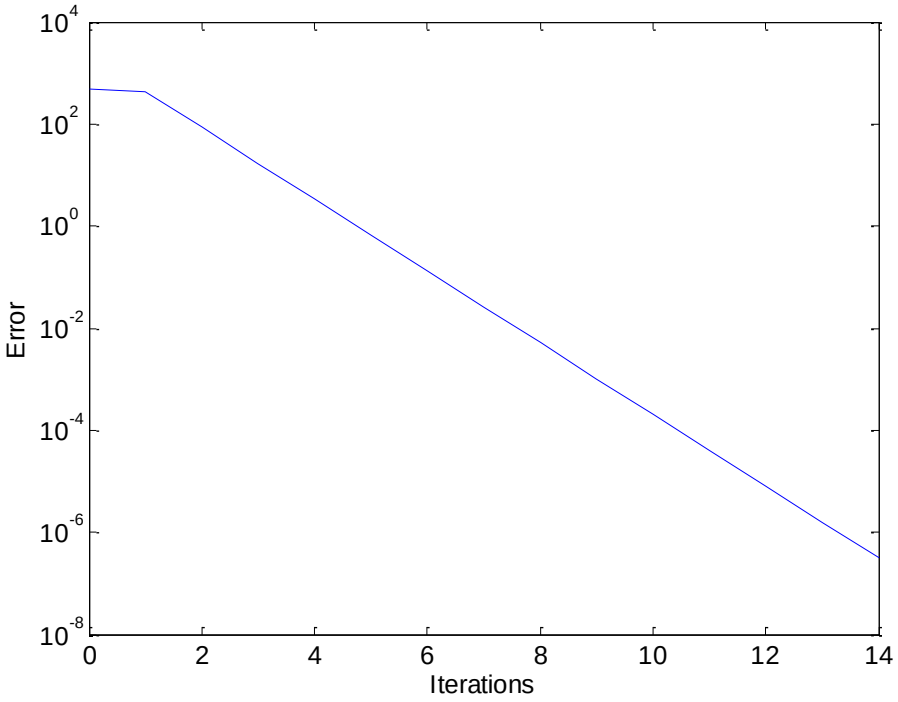

Figure 4. Powell's function convergence.

This function is interesting in that the convergence becomes linear on a log scale after the first iteration. 14 iterations, corresponding to less than 1000 function evaluations, was sufficient to generate a solution within 10^{-6} of the global minimum.

4. Conclusion

Newton's method proved a useful solution for convergence for test points on the same order of magnitude as the location of the actual global minimum. Its utility dropped when the dimension of the function was increased, but for the example functions and test points chosen, it always converged within 10^{-6} or closer to the global minimum in 1000 functional evaluations or fewer—in many cases much closer and with much fewer evaluations.