Homework 6

Ben Larson

Oct 20, 2016

1 Nelder-Mead Method

Nelder-Mead is a derivative free algorithm that works by minimizing a function based on a simplex model. f(x) is found for 3 arbitrary points, then based on the results the algorithm will experiment on a point estimated from the best, worst, and second best value of the simplex. We discard the worst point and re-run the algorithm. The transformations that the simplex can have is a reflection, expansion and contraction. These transformations use the geometry, ordering, and centroid to calculate/estimate new values. These are derivative free, but still have a similar process as gradient based optimization.
For this problem we solved the function:

$$f(x) = x_1^2 + x_2^2 + x_1x_2 + x_1x_3 + x_3x^2$$

I implemented the Nelder-Mead algorithm as found in the website using matlab: The results were as follows:

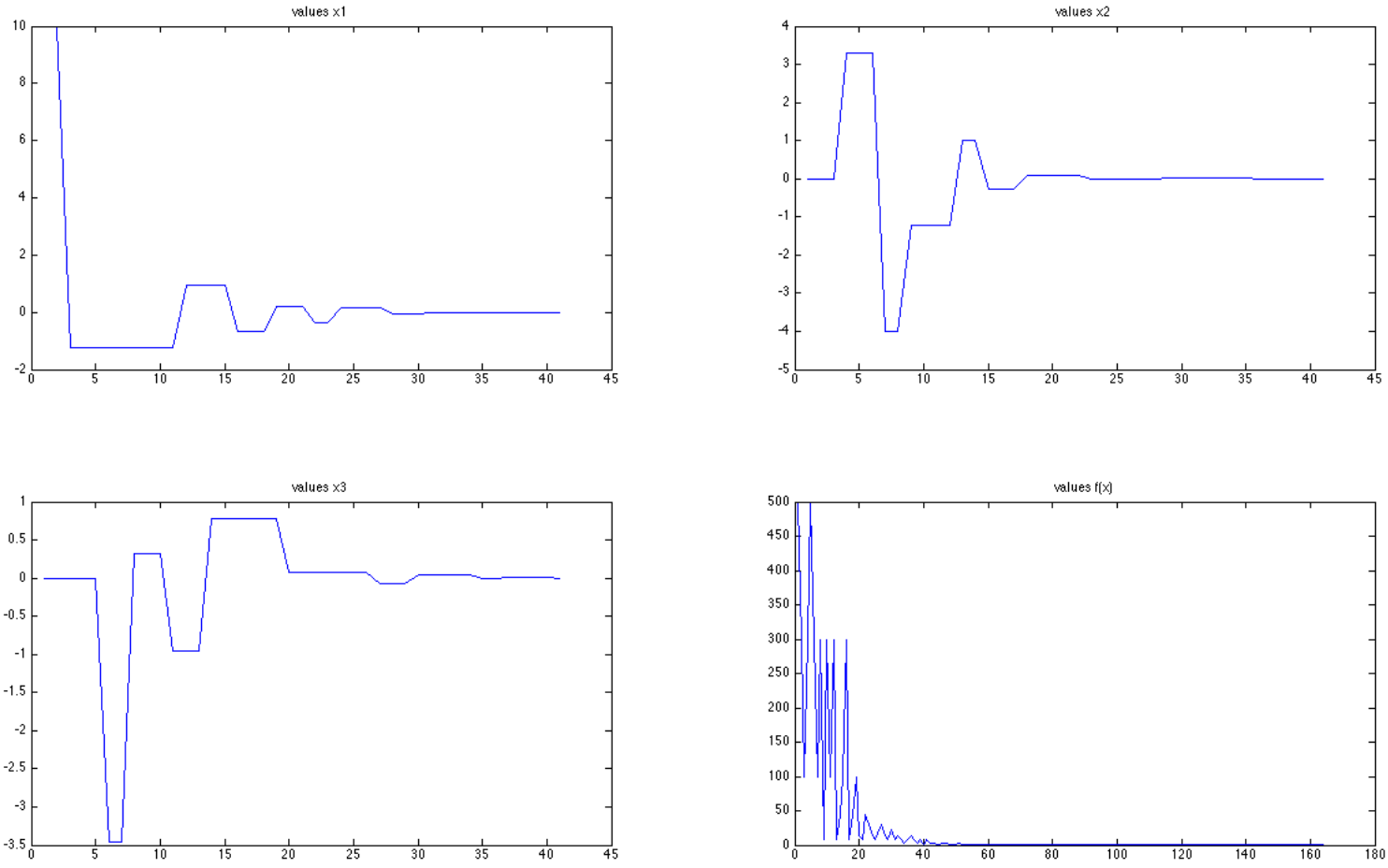

Figure 1: We can see that the x1,x2,x3 all come closer to zero as the algorithm progresses. At zero they kinda bounce around until the termination conditions are met. The bottom right plot is the function evaluated at the newly minimized (x1,x2,x3). Notice it approaches zero as the algorithm progresses.

2 Nelder-Mead, quasi-Newton, and conjugate gradient

For this section we compare the implemented algorithms. For the Bleale function I show a contour and the march of the algorithm towards the minimum. However, with the higher dimensions of the Rosenbrock function I only show the values evaluated at the function f(x).

2.1 Beale's Function

Of all the algorithms Conjugate gradient minimizes in the fewest steps.

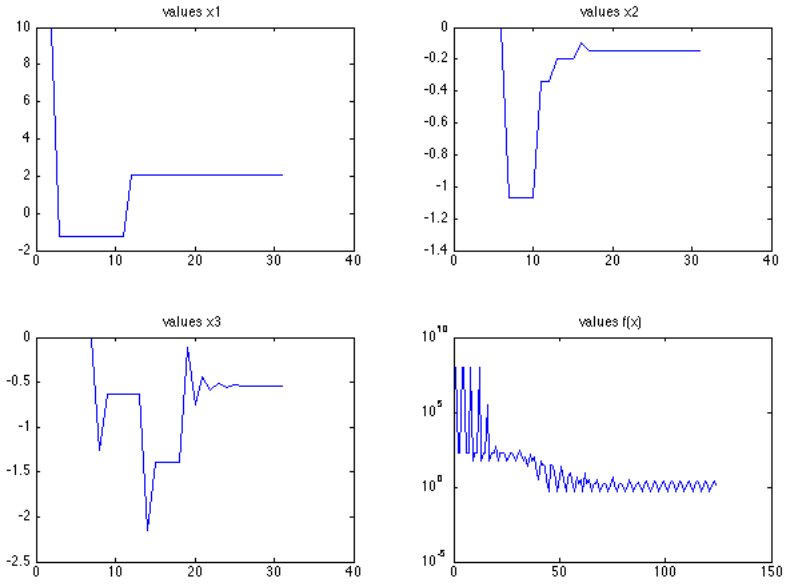

Figure 2: Nelder-Mead optimizing the Bleale function.

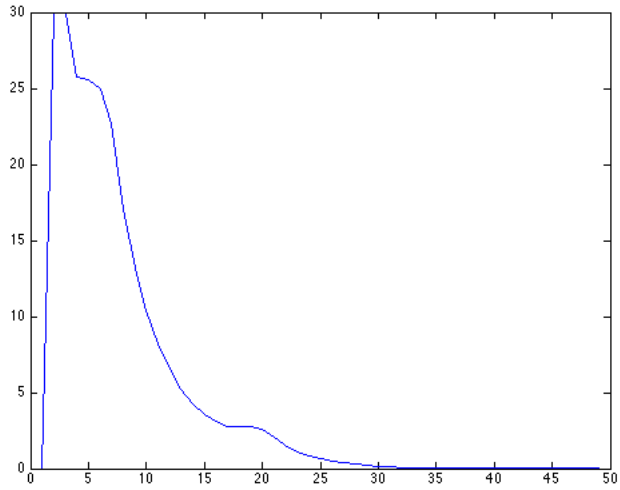

Figure 3: Quasi-Newton BFGS optimizing the Bleale function.

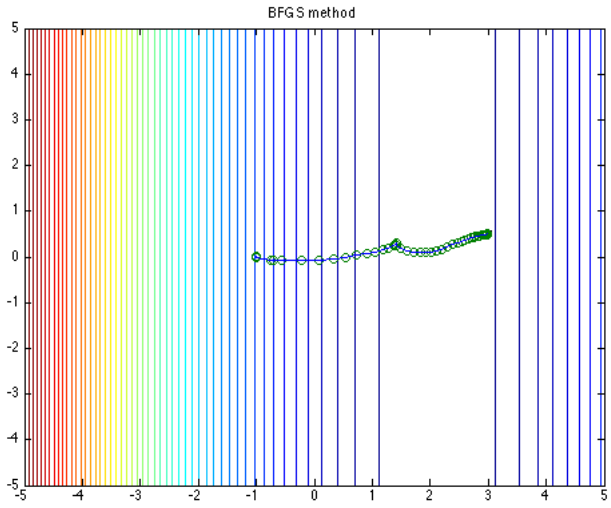

Figure 4: Contour plot of the BFGS method on the Bleale optimization.

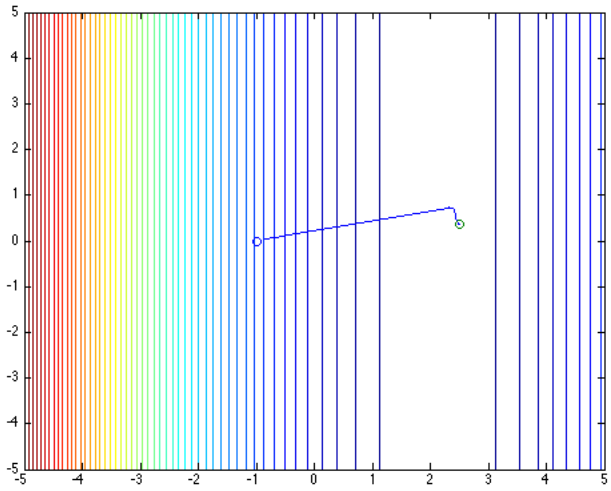

Figure 5: Conjugate Gradient optimizing the Bleale function.

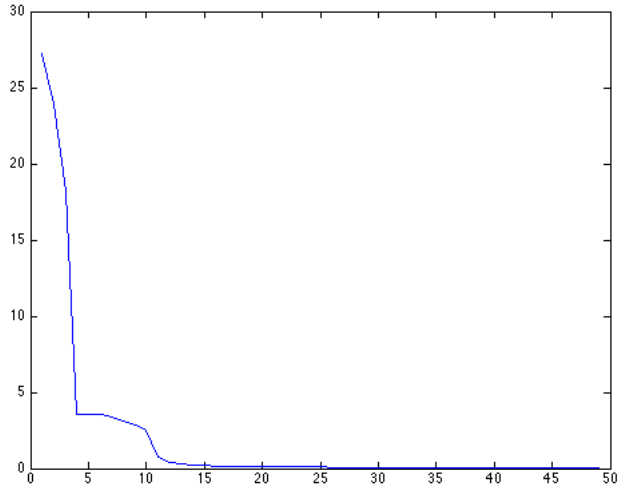

Figure 6: Function values of the optimization using conjugate gradient.

2.2 Rosenbrock Function

This function was difficult to visualize as it contained 4 variables. I plotted the values f(x) at each iteration. We can see that all the functions were minimized to a minimum. BFGS method was the fastest to converge for the experiments that I performed.

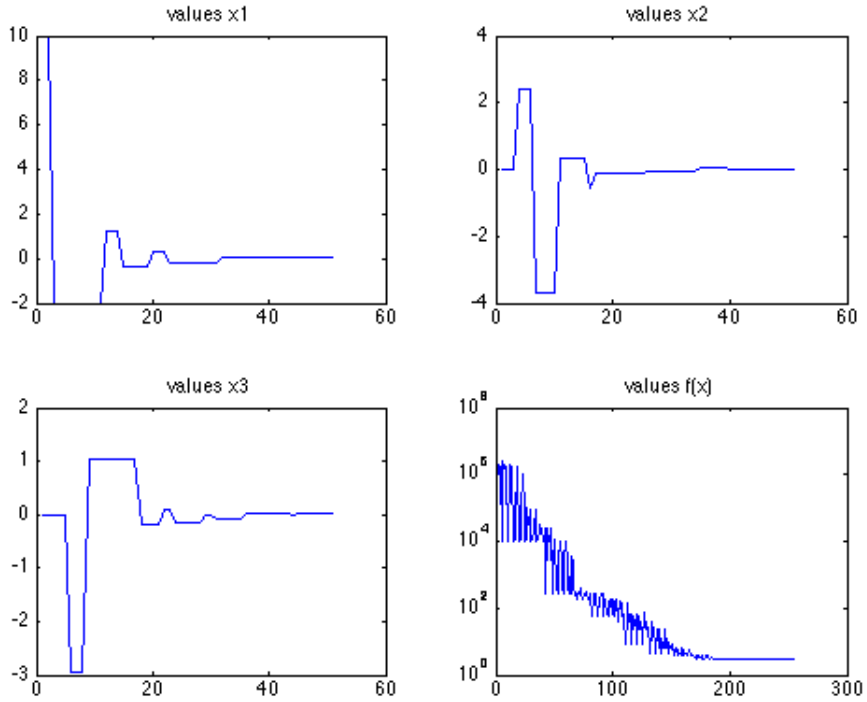

Figure 7: log(f(x)) values for the Nelder-Mead algorithm. I plotted using log so that a mor informative display of the values in relation to the iterations could be seen. We can see the simplex algorithm in action by considering the views of the x1,x2,x3 values. x4 not included.

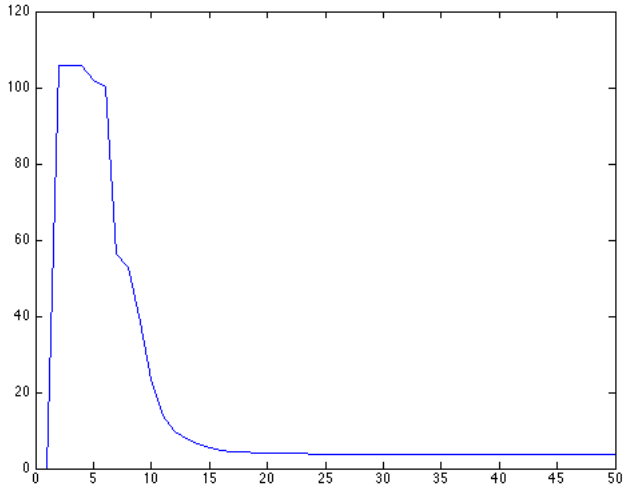

Figure 8: BFGS optimization for the rosen function.

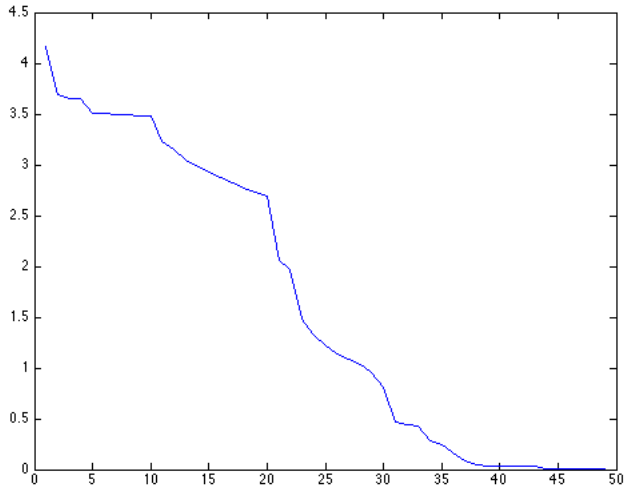

Figure 9: non-linear Conjugate gradient method. We see a slower convergence, which is not what we expect.

3 Lagrange Multipliers

$$min \sum_{i=1}^{n} a_i^2 x_i \quad \texttt{subject to} \quad \sum_{i=1}^{n} \frac{1}{x_i - c} - \frac{1}{b}$$

Following the Lagrangian equation:

$$L(x_i, \lambda = f(x_i) - \lambda(g(x_i) - c)$$

We get:

$$L = \sum_{i=1}^{n} a_i^2 x_i - \lambda \sum_{i=1}^{n} \frac{1}{x_i - c} - \frac{1}{b}$$

$$\nabla L = 0$$

$$\nabla L = \nabla \sum_{i=1}^{n} a_i^2 x_i - \nabla\lambda \sum_{i=1}^{n} \frac{1}{x_i - c} - \frac{1}{b}$$

$$= \sum_{i=1}^{n} a_i^2 \nabla x_i - \lambda \sum_{i=1}^{n} \nabla \frac{1}{x_i - c} - \frac{1}{b}$$

$$0 = \sum_{i=1}^{n} a_i^2 - \lambda \sum_{i=1}^{n} \frac{1}{-(x_i - c)^2}$$

$$\lambda = -\sum_{i=1}^{n} a_i^2 (x_i - c)^2$$