ASSIGNMENT 2

AER1415: Computational Optimization

Atilla Saadat Dehghan
atilla.saadat@mail.utoronto.ca
March 31st 2021

1.1 Question 1 – Newtons Method

Newton's method is a computer optimization method which utilizes the gradient vector and Hessian matrix of an objective function to ideally find the global minima. For this problem, the Rosenbrock objective function is examined and given by:

$$f(x) = \sum_{i=1}^{n-1} 100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2$$

Calculating the gradient vector of the Rosenbrock function for n=2 yields:

$$g(x) = \nabla f(x) = \begin{bmatrix} \frac{\partial f(x)}{\partial x_1} \\ \vdots \\ \frac{\partial f(x)}{\partial x_n} \end{bmatrix} \in \mathbb{R}^n = \begin{bmatrix} 2x_1 - 400x_1(x_2 - x_1^2) - 2 \\ 200(x_2 - x_1^2) \end{bmatrix}$$

Calculating the Hessian matrix of the Rosenbrock function for n=2 yields:

$$H(x) = \nabla^2 f(x) = \begin{bmatrix} \frac{\partial^2 f(x)}{\partial x_1^2} & \cdots & \frac{\partial^2 f(x)}{\partial x_n \partial x_1} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 f(x)}{\partial x_1 \partial x_n} & \cdots & \frac{\partial^2 f(x)}{\partial x_n^2} \end{bmatrix} \in \mathbb{R}^{n \times n} = \begin{bmatrix} 1200x_1^2 - 400x_1 + 2 & -400x_1 \\ -400_{x1} & 200 \end{bmatrix}$$

The newtons method utilizes the gradient vector and Hessian matrix to calculate the search direction for the optimization algorithm and yields the newly calculated x_{k+1} in each iteration:

$$p_k = -H_k^{-1} g(x_k)$$

$$x_{k+1} = x_k + p_k$$

Applying the Newton's method on Rosenbrock function for $n = 2$ and $x_0 = [-1, -2]^T$ yields the logarithmically scaled contour plot in **Figure 1: Newtons Method on Rosenbrock Test Function n=2 - First 5 iterates**Figure 1. The first five iterates are plotted in the green line, showing that the solution converges to the known Rosenbrock global minima of $f(x^*) = [1,1]$ and value of $f(x^*) = 0.0000$.

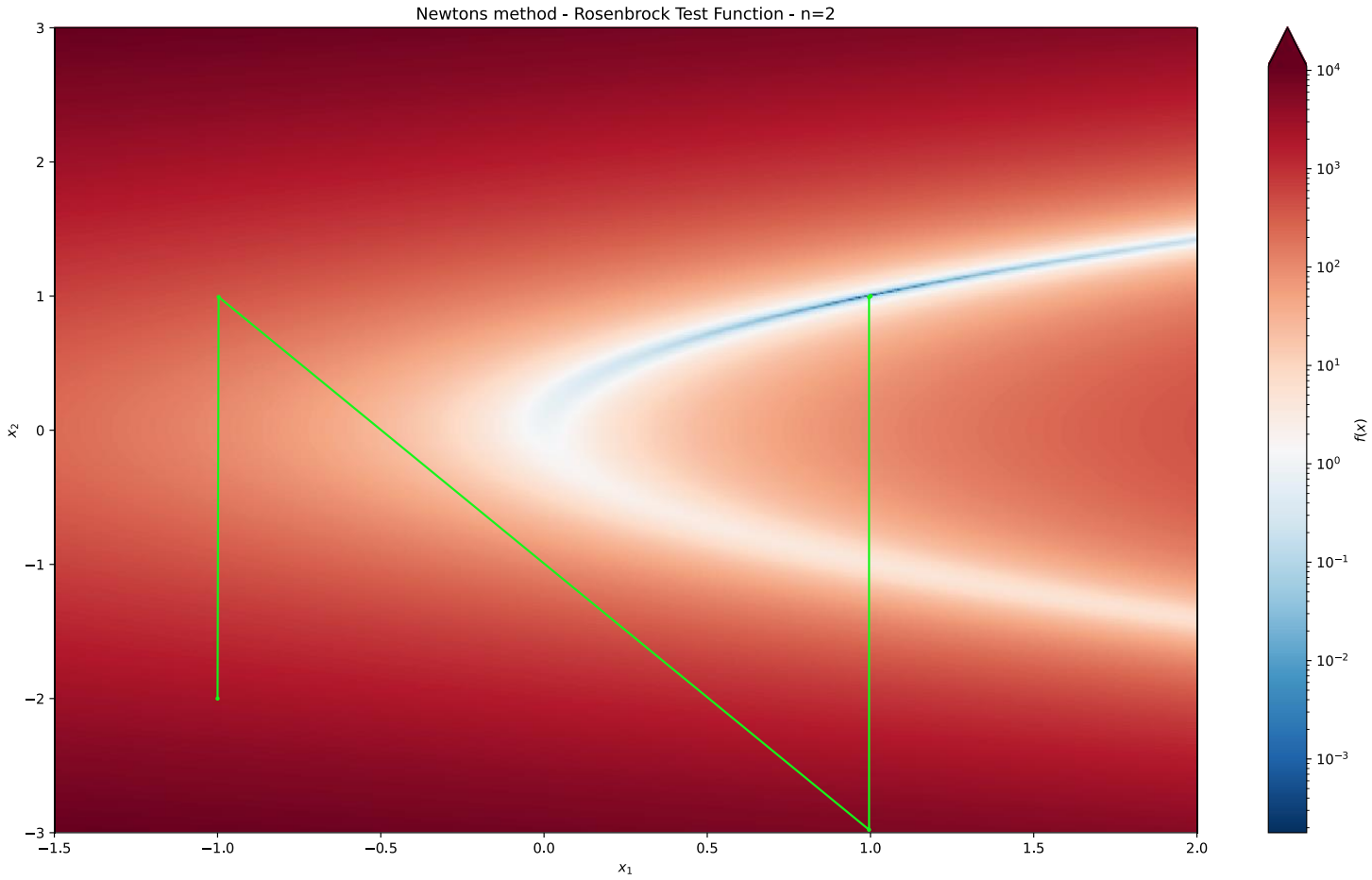

Figure 1: Newtons Method on Rosenbrock Test Function n=2 - First 5 iterates

1.2 Question 2 – Quasi-Newton's Method:

The Quasi-Newton method utilizes only first order information and an approximate Hessian based on the function values and gradients from previous iterations. The outline used for this Quasi-Newton method on the Rosenbrock test function ($n = 2$ and $n = 5$) is as follows:

1. Initialize $x_0 = UNIFORM(-5,5,n)$, $\epsilon_g = 10^{-6}, \epsilon_a = 10^{-10}, \epsilon_r = 10^{-7}$, and $B_0 = I$
2. Compute $g_k = \nabla f(x_k)$. If $\|g_k\|_2 \leq \epsilon_g$ then stop, otherwise continue
3. Compute search direction, $p_k = -B_k^{-1} g_k$
4. Line search: find positive step-length α_k using Armijo's sufficient decrease condition and backtracking
   4.1. Chose starting step length, $\alpha_0 = 0.95$
   4.2. If $f(x_k + \alpha p_k) \leq f(x_k) + \mu_1 \alpha g_k^T p_k$ (where $\mu_1 = 10^{-4}$), then set $a_k = \alpha$ and stop
   4.3. $\alpha = \rho\alpha$, where $\rho = 0.5$ and repeat
5. Update position, $x_{k+1} = x_k + a_k p_k$
6. Evaluate $f(x_{k+1})$. If $|f(x_{k+1}) - f(x_k)| \leq \epsilon_a + \epsilon_r |f(x_k)|$ is satisfies for two successive iterations then stop

7. Update the approximation to the inverse of the Hessian B_{k+1}^{-1} using Broyden-Fletcher-Goldfarb-Shanno (BFGS) update method:
   7.1. Set $y_k = g_{k+1} - g_k$ and $s_k = x_{k+1} - x_k$

   7.2. $B_{k+1}^{-1} = \left[I - \frac{s_k y_k^T}{s_k^T y_k}\right] B_k^{-1} \left[I - \frac{y_k s_k^T}{s_k^T y_k}\right] + \frac{s_k s_k^T}{s_k^T y_k}$

8. Repeat steps 2-8 until any criteria is met: $k = k + 1$

Calculating the gradient vector of the Rosenbrock function for n=5 yields:

$$g(x) = \begin{bmatrix} 2(x_1 - 1) - 400x_1(x_2 - x_1^2) \\ 2(x_2 - 1) - 400x_2(x_3 - x_2^2) + 200(x_2 - x_1^2) \\ 2(x_3 - 1) - 400x_3(x_4 - x_3^2) + 200(x_3 - x_2^2) \\ 2(x_4 - 1) - 400x_4(x_5 - x_4^2) + 200(x_4 - x_3^2) \\ 200(x_5 - x_4^2) \end{bmatrix}$$

Running the Quasi-Newton method on the Rosenbrock test function (n=2) yields the contour and convergence plot (function value and ℓ_2 norm of the gradient) shown in Figure 2 and Figure 3, respectively. Figure 3 also demonstrates the performance of the Quasi-Newton method relative to a Steepest Descent method with backtracking. The Quasi-Newton method clearly out preforms the conventional steepest descent, as it quickly converges to the Rosenbrock global minima with very little iterations required. Figure 4 also shows this similar performance with the Rosenbrock test function for $\mathrm{n} = 5$.

To test the repeatability performance of the two methods, 20 runs were evaluated and the mean values for each method are shown for $n = 2$ and $n = 5$ in Table 1 and Table 2, respectively. It is important to note that for both methods, not every evaluation guarantees that the solution will converge to the true global minima, and thus requires revaluation and comparison to validate the results and chose an acceptable final solution.

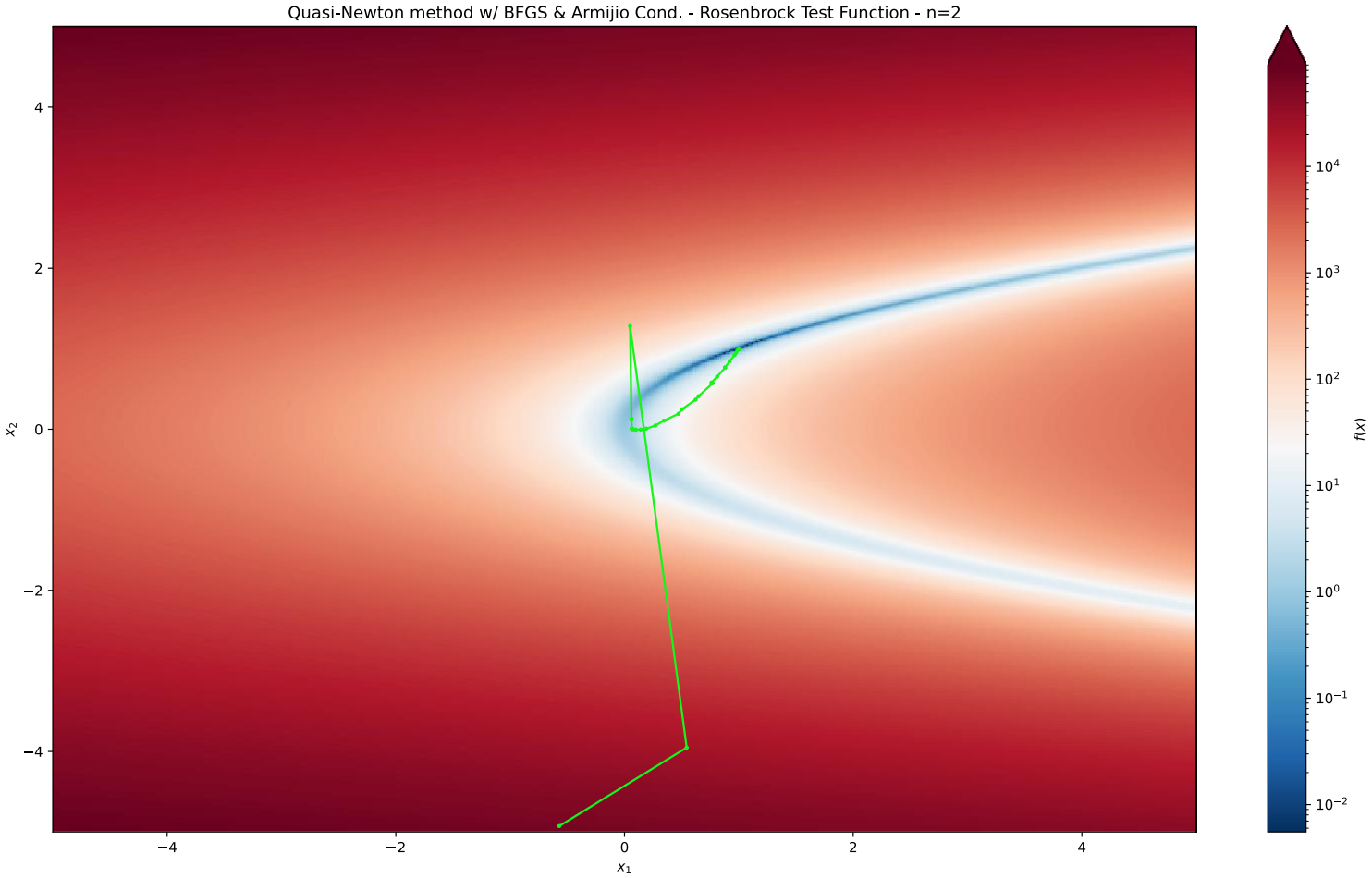

Figure 2: Sample run - Quasi-Newton method w/ BFGS & Armijio Cond. Contour Plot- Rosenbrock Test Function - n=2

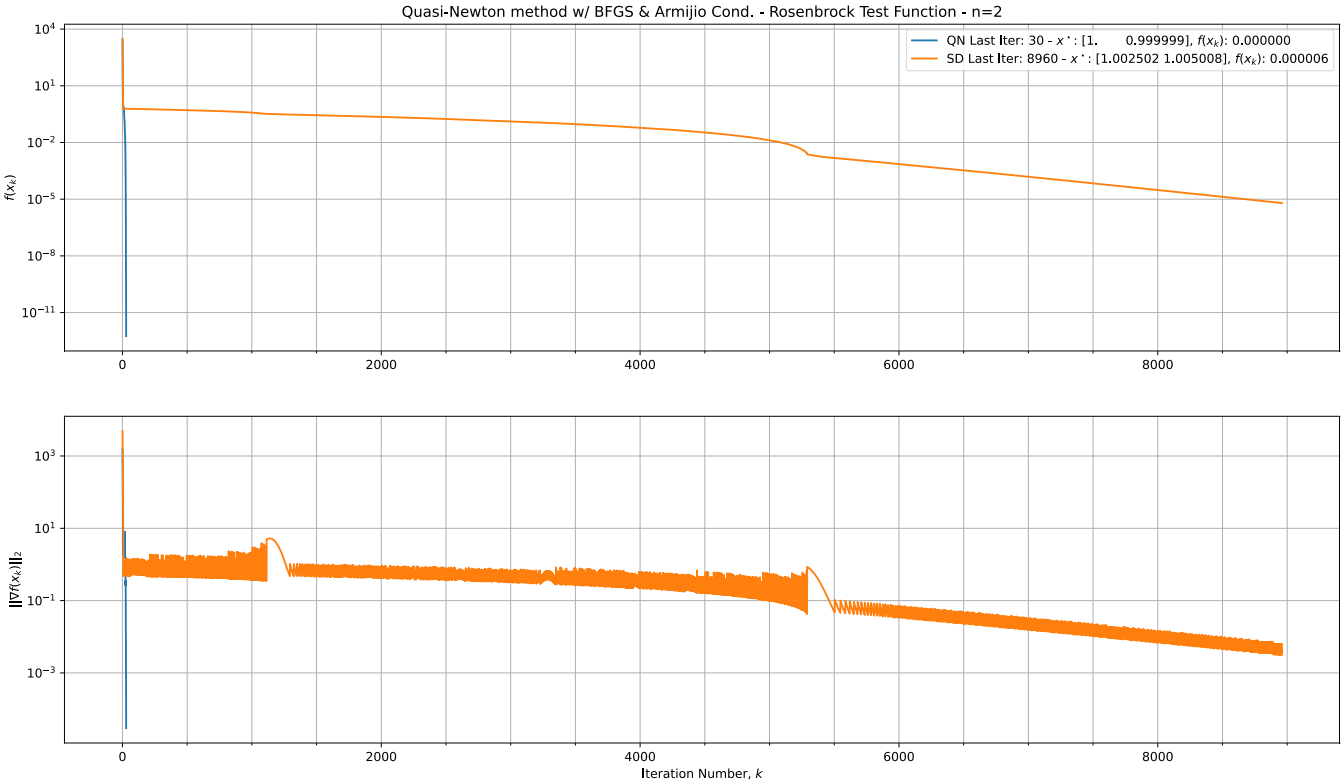

Figure 3: Sample run - Quasi-Newton method w/ BFGS & Armijio Cond. Convergence Plot- Rosenbrock Test Function - n=2

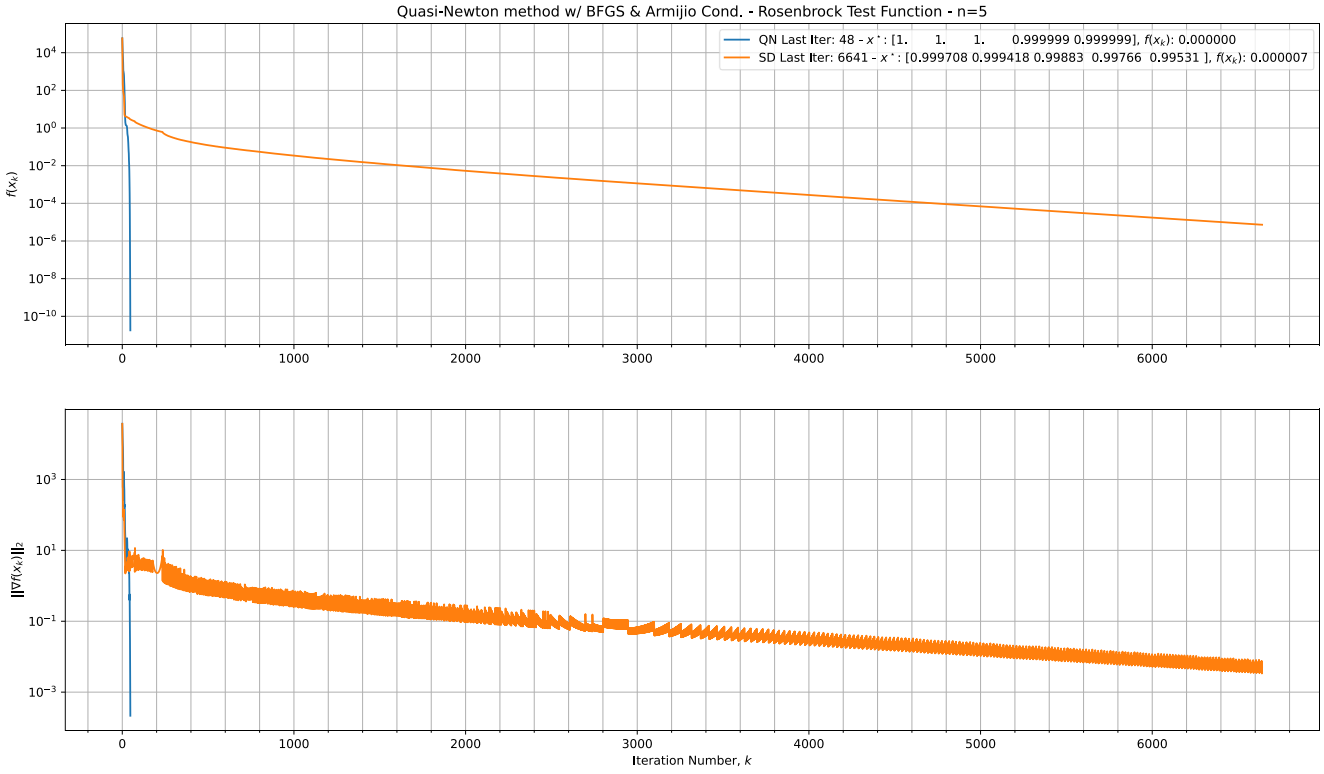

Figure 4: Sample run - Quasi-Newton method w/ BFGS & Armijio Cond. Convergence Plot- Rosenbrock Test Function - n=5

Table 1: Quasi-Newtons and Steepest Descent Mean Values for 20 runs, n=2

	x^*	$f(x^*)$	Iterations to reach x^*
Quasi-Newton	[0.97980972, 0.96851699]	0.009261673330379838	39
Steepest Descent	[0.85451783, 3.74249363]	3.0337720390590723	6101

Table 2: Quasi-Newtons and Steepest Descent Mean Values for 20 runs, n=5

	x^*	$f(x^*)$	Iterations to reach x^*
Quasi-Newton	[0.70569261, 0.99036003, 0.98210549, 0.96667801, 0.94075843]	0.5896259169463631	65
Steepest Descent	[0.50699289, 0.98874422, 0.97954007, 0.96234133, 0.93429586]	0.9885878412185477	8860

1.3 Question 3 & 4 – Barzilai and Borwein (BB) method:

The Barzilai and Borwein (BB)method is similar to the steepest descent method in that the iterates are computed as $x_{k+1} = x_k - a_k \nabla f(x_k)$. However, the BB method is like the Quasi-Newton method in that the hessian is not used and approximates the step length using the previous iteration's function value and gradient.

To evaluate the performance of the BB method, a dataset for a Convex Quadratic function will be used, which contains the A matrix and b vector. The convex quadratic function has the form:

$$f(x) = \frac{1}{2} x^T A x - x^T b$$

where $A \in \mathbb{R}^{n \times n}$ is a SPD sparse 341×341 matrix, $b \in \mathbb{R}^n$, and $n = 341$.

Since the problem involves a quadratic objective function and the BB method requires information from the previous iteration, the first iteration may use exact line search, given by:

$$a_k = \frac{g_k^T g_k}{g_k^T A g_k}$$

For every other iteration, the step length can be approximated the secant equation for the BB algorithm:

$$a_k = \frac{s_k^T s_k}{s_k^T y_k}$$

where $s_k = x_k - x_{k-1}$ and $y_k = \nabla f(x_k) - \nabla f(x_{k-1})$.

The BB method outline is as follows:

9. Initialize $x_0 = UNIFORM(-1,1,n)$, $\epsilon_g = 10^{-6}, \epsilon_a = 10^{-10}, \epsilon_r = 10^{-7}$
10. Compute $g_k = \nabla f(x_k)$. If $\|g_k\|_2 \leq \epsilon_g$ then stop, otherwise continue
11. Compute search direction, $p_k = -g_k$
12. Line search: find positive step-length α_k using exact line search (for $k = 1$) or secant equation ($k > 1$)
13. Update position, $x_{k+1} = x_k + a_k p_k$
14. Evaluate $f(x_{k+1})$. If $|f(x_{k+1}) - f(x_k)| \leq \epsilon_a + \epsilon_r |f(x_k)|$ is satisfies for two successive iterations then stop

Evaluating through the outlined steps for 20 runs using the BB and steepest descent method yields the mean results shown in Table 3. Both methods converge to the same function global minimum with comparable x^* values. The x^* values are stored in the csv file names *Q34_x.csv* in the directory alongside the source code.

Figure 5 shows a sample run of both the BB and Steepest Descent method for the convex quadratic function. The label shows that both methods reach approximately the same value, however the BB method reaches that value significantly faster than the alternative method, specifically 76 vs 432 iterations. Therefore, the BB method is a computationally less intensive method which reaches the minimum value faster (independent of processor speed).

Studying the convergence and gradient plots of the BB method reveals some interesting behaviours. The steepest descent method seems to asymptotically reach the global minimum value, given that both the function value and gradient value gradually decrease and converge after many successive iterations. Unlike this, the BB method seems to drastically spike in function and gradient values in early iterations (relative to steepest descent), however it also generally follows a R-linearly convergence.

Table 3: BB and Steepest Descent Mean Values for 20 runs - Convex Quadratic

	$f(x^*)$	**Iterations to reach x^***
Quasi-Newton	-9.45202717442865	76
Steepest Descent	-9.44622718042555	355

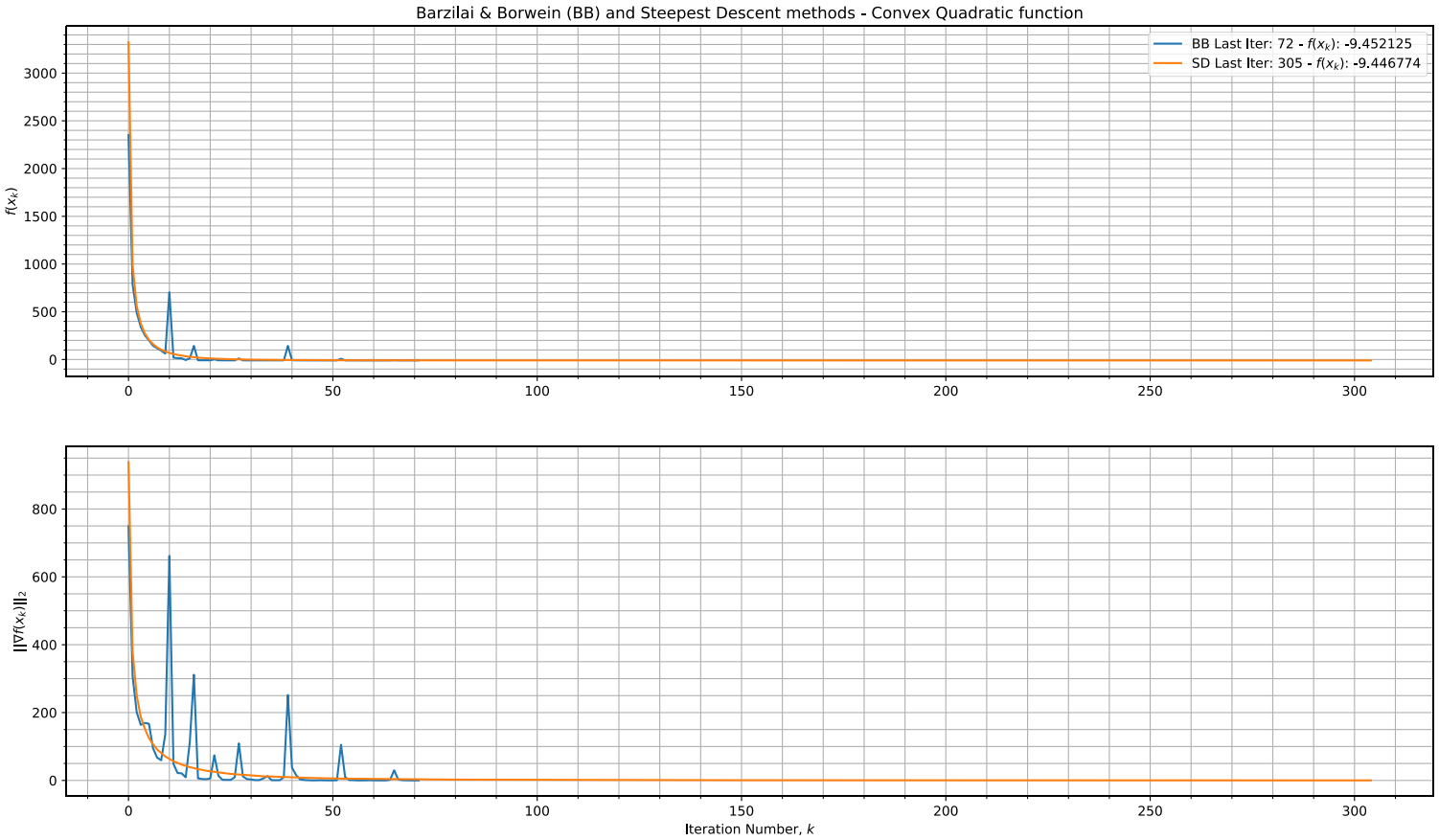

Figure 5: Sample Run - BB vs Steepest Descent Method - Convex Quadratic

1.4 Question 5:

a) To approach an estimate for the parameters α_k, β_k, and γ_k so that the search direction p_k is close to the Newton search direction, the two equations for the search directions must be approximately equivalent:

$$p_k = -H_k^{-1} g_k \approx -\alpha_k g_k + \beta_k p_{k-1} + \gamma_k p_{k-2}$$

Rearranging the Hessian matrix so that the inverse hessian computation is not conducted:

$$-g_k = H_k(-\alpha_k g_k + \beta_k p_{k-1} + \gamma_k p_{k-2})$$

Expanding out the hessian so that the coefficients are parameterized:

$$-g_k = [\alpha_k \quad \beta_k \quad \gamma_k] \begin{bmatrix} -H_k g_k \\ H_k p_{k-1} \\ H_k p_{k-2} \end{bmatrix}$$

Since the problem states that the coefficient must provide an estimate close to an appropriate norm of the Newton search direction, the row-wise ℓ_2-norm can be taken from the left side after an element-wise division

$$[\alpha_k \quad \beta_k \quad \gamma_k] = \left\| -g_k \oslash \begin{bmatrix} -H_k g_k \\ H_k p_{k-1} \\ H_k p_{k-2} \end{bmatrix} \right\|_2$$

Since there is no matrix inversion, this operation to estimate the coefficients no greater than $O(n^2)$.

b) Similar to the first part, the parameters α_k, β_k, and γ_k can be estimated without the Hessian matrix and instead using only first-order derivatives. One way to achieve this is with the BFGS method coupled with the estimation equations above. (Starting with $B_0 = I$)

$$y = x_{k+1} - x_k$$

$$s_k = g_{k+1} - g_k$$

$$[\alpha_k \quad \beta_k \quad \gamma_k] = \left\| -g_k \oslash \begin{bmatrix} -B_k g_k \\ B_k p_{k-1} \\ B_k p_{k-2} \end{bmatrix} \right\|_2$$

$$B_{k+1} = B_k + \frac{(y_k - B_k s_k)(y_k - B_k s_k)^T}{(y_k - B_k s_k)^T s_k}$$