Numerical Optimization
Re-exam Handin 3

Dmitry Serykh (qwl888)

May 23, 2020

1 The Setup

I implemented the stepest descent and Newton's algorithms using the backtracking line search in order to find the step length (α).

1.1 Parameters

I have empirically found that for my implementation, different parameters must be used for the two methods. Following values were used:

- **Initial step length.** For the Netwon's method, I have used the natural value of ($\bar{\alpha} = 1$). For the stepest descent, I empirically found the value of $\bar{\alpha} = 0.1$ to perform best
- **Factor** ρ. For the stepest descent, I used the value of $\rho = 0.2$. For the Newton's method, I have used the value of $\rho = 0.5$.
- **Constant** c. For both methods, I have used to value of $c = 10^{-5}$.
- **Stopping criteria**. For the stepest descent, I have used $\epsilon = \mathbf{d}^T\mathbf{d} = 10^{-7}$. For the Newton's method, I have used $\epsilon = \mathbf{d}^T\mathbf{H}\mathbf{d} = 10^{-7}$. Furthermore, I have terminated the algorithm at 20000 iterations. However this was only relevant to the stepest descent.

2 Testing protocol

In order to test the effectiveness of my implementation, I came up with a testing protocol, where I used following metrics:

- The convergence plots with the number of iteration on the x-scale and the log of the Euclidean distance between the current value of x and the optimum. The results can be seen on Figure 1. I applied the low shelf to the results, so anything would stop at 10^{-6}.
- The convergence plots with the gradient magnitude. The results can be seen on Figure 2.
- **Efficiency**. The number of steps until the gradient magnitude reaches 10^{-7}. The results can be seen on Table 2.
- **Accuracy**. The Euclidean distance to the optimum at the termination point. The results can be seen on Table 1.

I used a random starting point taken from the uniform distribution in the interval between -10 to 10 and repeated each optimization 100 times for both metrics and took the average. This was done to remove the influence of the starting position from the results of the optimization. I used the mean to minimize the effect of the outliers, which is important due to the stochastic nature of the algorithm.

	f_1	f_2	f_3	f_4	f_5
stepest descent	$1.437 \cdot 10^{-4}$	$7.59 \cdot 10^{-4}$	$1.44 \cdot 10^{-10}$	$6.51 \cdot 10^{-7}$	$1.13 \cdot 10^{-4}$
newton	0.0	$8.15 \cdot 10^{-8}$	$4.59 \cdot 10^{-16}$	$6.4 \cdot 10^{-7}$	$4.56 \cdot 10^{-7}$

Table 1: Average value of distance to the optimum at algorithm termination for 100 random starting points in the interval $[-10, 10]$

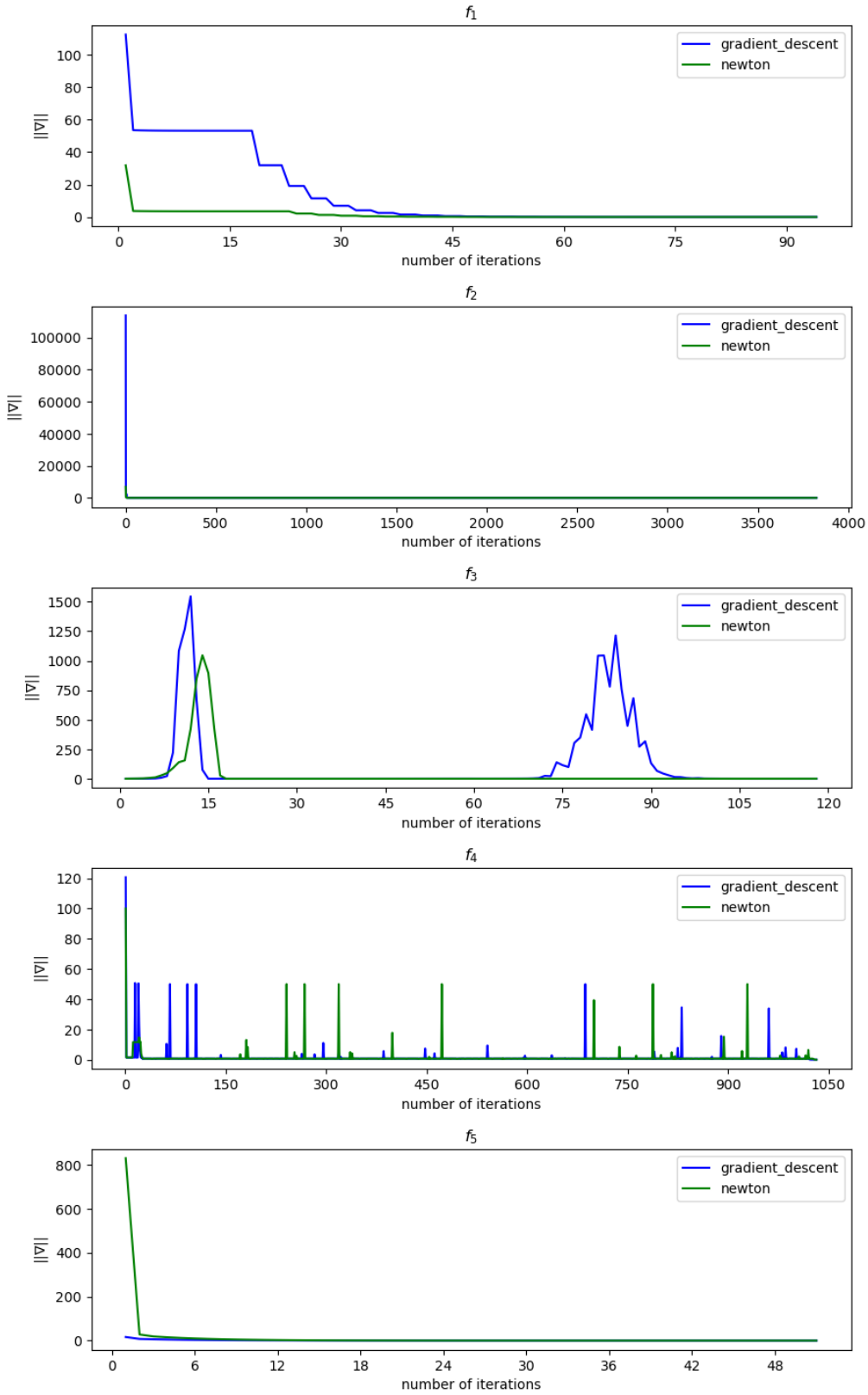

Figure 1: Convergence plot with norm of the gradient on y-axis

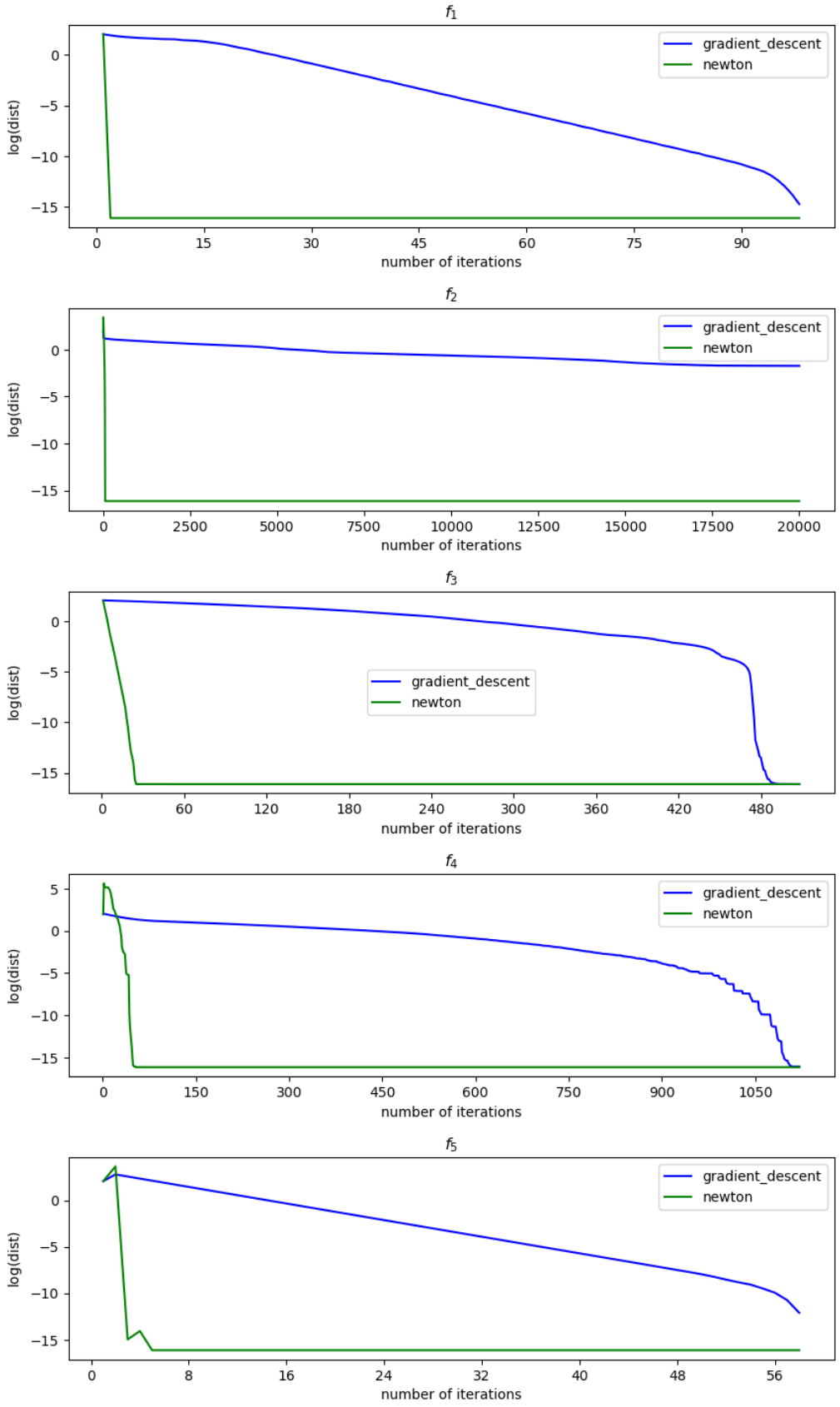

Figure 2: Convergence plot with log of euclidean distance on the y-axis

	f_1	f_2	f_3	f_4	f_5
stepest descent	3392	6991	5546	488	52
newton	1	30	16	29	2

Table 2: Average number of iterations until algorithm termination for 100 random starting points in the interval $[-10, 10]$

3 Theoretical Part

3.1 Exercise 3.7

By the definition of the Steepest Descent method, and subtracting the x^* on both sides, we have:

$$x_{k+1} - x^* = x_k - x^* - \alpha \nabla f(x_k)$$

Then, by the definition of the Q-weighted norm, we have:

$$\begin{aligned}
\|x_{k+1} - x^*\|_Q^2 &= (x_k - x^* - \alpha \nabla f(x_k))^T Q (x_k - x^* - \alpha \nabla f(x_k)) \\
&= (x_k - x^*)^T Q (x_k - x^*) - 2\alpha \nabla f(x_k)^T Q (x_k - x^*) + \alpha^2 \nabla f(x_k)^T Q \nabla f(x_k) \\
&= \|x_k - x^*\|_Q^2 - 2\alpha \nabla f(x_k)^T Q (x_k - x^*) + \alpha^2 (x_k) \nabla f(x_k)^T Q \nabla f(x_k)
\end{aligned}$$

We know that $\nabla f = Q(x_k - x^*)$ and $\alpha = \nabla f^T \nabla f / \left(\nabla f^T Q \nabla f\right)$, therefore:

$$\begin{aligned}
\|x_{k+1} - x^*\|_Q^2 &= \|x_k - x^*\|_Q^2 - 2\alpha \nabla f(x_k)^T \nabla f(x_k) + \alpha^2 \nabla f(x_k)^T Q \nabla f(x_k) \\
&= \|x_k - x^*\|_Q^2 - 2 \left(\nabla f(x_k)^T \nabla f(x_k)\right)^2 / \left(\nabla f(x_k)^T Q \nabla f(x_k)\right) + \left(\nabla f(x_k)^T \nabla f(x_k)\right)^2 / \left(\nabla f(x_k)^T Q \nabla f(x_
\\
&= \|x_k - x^*\|_Q^2 - \left(\nabla f(x_k)^T \nabla f(x_k)\right)^2 / \left(\nabla f(x_k)^T Q \nabla f(x_k)\right) \\
&= \|x_k - x^*\|_Q^2 \left[1 - \frac{\left(\nabla f(x_k)^T \nabla f(x_k)\right)^2}{\left(\nabla f(x_k)^T Q \nabla f(x_k)\right) \|x_k - x^*\|_Q^2}\right]
\end{aligned}$$

And since $\|x_k - x^*\|_Q^2 = \nabla f(x_k)^T Q^{-1} \nabla f(x_k)$:

$$\|x_{k+1} - x^*\|_Q^2 = \|x_k - x^*\|_Q^2 \left[1 - \frac{\left(\nabla f(x_k)^T \nabla f(x_k)\right)^2}{\left(\nabla f(x_k)^T Q \nabla f(x_k)\right) \left(\nabla f(x_k)^T Q^{-1} \nabla f(x_k)\right)}\right]$$

3.2 Application to the Ellipsoid Function

Let $g(k) = \frac{||f_{k+1}(x) - f(x^*)||_Q^2}{||f_k(x) - f(x^*)||_Q^2}$. Then, by Eq. 3.27 and 3.29:

$$g(k) \leq \left(\frac{\lambda_n - \lambda_1}{\lambda_n + \lambda_1}\right)^2$$

where $0 < \lambda_1 \leq \lambda_2 \leq ... \leq \lambda_n$ are the eigenvalues of Q. The Ellipsoid function is given by:

$$f_1(x) = \sum_{i=1}^{d} \alpha^{\frac{i-1}{d-1}} x_i^2$$

In matrix form, it can be represented as $f_1(x) = \mathbf{x}^T \mathbf{Q} \mathbf{x}$, where

$$\mathbf{Q} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & \alpha^{\frac{1}{4}} & 0 & 0 & 0 \\ 0 & 0 & \alpha^{\frac{2}{4}} & 0 & 0 \\ 0 & 0 & 0 & \alpha^{\frac{3}{4}} & 0 \\ 0 & 0 & 0 & 0 & \alpha \end{pmatrix}$$

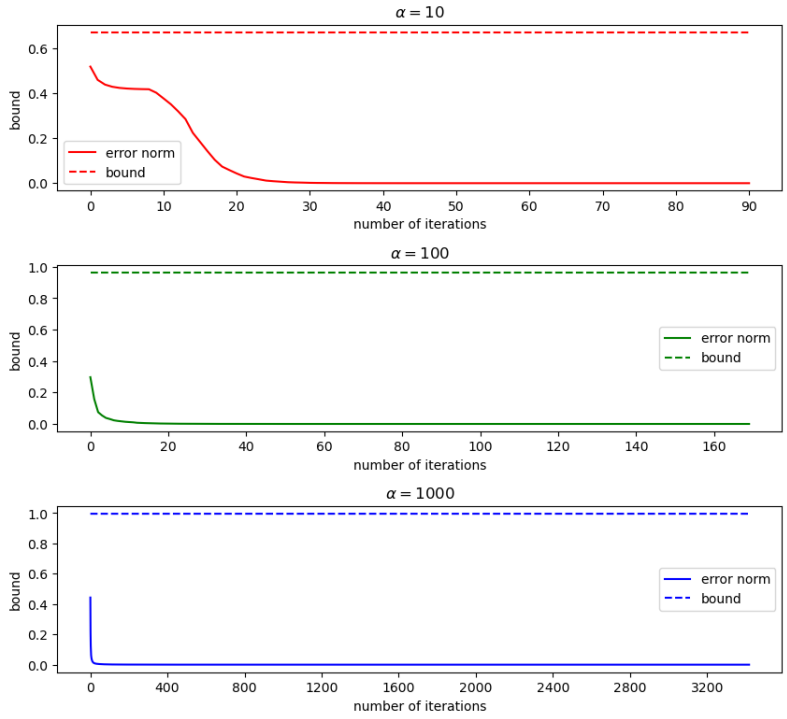

Figure 3: Plot of $g(k)$ and bound 3.29 for the Ellipsoid function with $d = 5$ different values of α

for $d = 5$. Therefore $\lambda_1 = 1$ and $\lambda_n = \alpha$, and bound would equal:

$$g(k) \leq \left(\frac{\alpha - 1}{\alpha + 1}\right)^2$$

I validate this bound empirically for different values of α. The plot of the results can be seen on Figure 3. It can be seen that the bound holds and becomes more and more lose for higher values of α.

4 Conclusion

I have implemented both the steepest descent and Newton's method using the backtracking line search in order to find the step size. The results show that the Newton's method outperforms Gradient descent for all case functions, and there is an asymptotic difference between the convergence rates of the Steepest Descent and Newton's method. The gradient descent especially struggles with the first three functions. However, both methods manage to find the optima of all functions.