Optreg Assignment 9 Eirik Falck

Problem 1

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

a) We can see that the steepest descent algorithm uses close to 10 000 iterations before the gradient termination criterion is met. We also see the function value decreases very slowly.

Conversely, Newtons descent uses considerably fewer iterations as well as the function value becoming smaller.

The condition in algorithm 3.1 ensures that the algorithm terminates at a step length α.

b) The BFGS algorithm uses more iterations than exact Newton, but the resulting function value is smaller. This is the case with both initial points.

c) First, consider $x_0 = [1.2 \;\; 1.2]^T$.

For Newton, the step length α is 1 most of the iterations, except for one. With BFGS, the step length varies more throughout the iterations.
The last iteration in BFGS uses $\alpha = 0.9$, but $\alpha = 1$ the iteration before. For Newton, $\alpha = 1$ the last few iterations.

For $x_0 = [-1.2 \;\; 1]^T$, Newtons α swings between 1 and 0.6 - 0.8 a period, then settles to $\alpha = 1$ towards the end.
For BFGS, the story is similar, but the swing amplitude is smaller and continues for more iterations.

The convergence theory for Newton fits better than for BFGS, since $\alpha = 1$ more in Newton.

NB! $c_1 = 0.5$ have been used in the entire exercise in the line search algorithm.

Problem 2

It's desirable to change the Hessian if it's not positive definite.

Problem 3

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

a) $\nabla f = \begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \frac{\partial f}{\partial x_2} \end{bmatrix} \approx (*)$

$$\frac{\partial f}{\partial x_1} \approx \frac{f(x + \epsilon e_1) - f(x)}{\epsilon}$$

$$= \frac{100(x_2 - x_1 - \epsilon)^2 + (1 - x_1 - \epsilon)^2 - 100(x_2 - x_1)^2 - (1 - x_1)^2}{\epsilon}$$

$$= \frac{1}{\cancel{\epsilon}}\left[101\epsilon^{\cancel{2}} + 202\cancel{\epsilon} x_1 - 200\cancel{\epsilon} x_2 - 2\cancel{\epsilon}\right]$$

$$= 202x_1 - 200x_2 - 2 + 101\epsilon$$

$$= -200(x_2 - x_1) - 2(1 - x_1) + 101\epsilon$$

$$\frac{\partial f}{\partial x_2} \approx \frac{f(x + \epsilon e_2) - f(x)}{\epsilon}$$

$$= \frac{100(x_2 + \epsilon - x_1)^2 + \cancel{(1 - x_1)^2} - 100(x_2 - x_1)^2 - \cancel{(1 - x_1)^2}}{\epsilon}$$

$$= \frac{1}{\cancel{\epsilon}}\left[100\epsilon^{\cancel{2}} - 200\cancel{\epsilon} x_1 + 200\cancel{\epsilon} x_2\right]$$

$$= 200(x_2 - x_1) + 100\epsilon$$

$(*)$
$$\Rightarrow \quad \nabla f \approx \begin{bmatrix} -200(x_2 - x_1) - 2(1 - x_1) + 101\epsilon \\ 200(x_2 - x_1) + 100\epsilon \end{bmatrix} = Gf$$

b) $\nabla f = \begin{bmatrix} -200(x_2 - x_1) - 2(1 - x_1) \\ 200(x_2 - x_1) \end{bmatrix}$

We see that for large values of ϵ the difference is large (of the order of 1), but the difference steadily falls towards 0 (10^{-14}) when ϵ becomes smaller.

Since the analytic derivative evaluates $(*)$ as $\epsilon \to 0$, this result is in line with the theory on derivatives and that taking the difference over a smaller step gives a better approximation.

c) We can see that not choosing ϵ small enough can give very poor approximation of the derivative.

Problem 4

a) The set $\{x_1, \ldots, x_{n+1}\}$ describing the simplex and its associated matrix $[x_2 - x_1, x_3 - x_1, \ldots, x_{n+1} - x_1]$ must be nonsingular, meaning they must not be coplanar/in the same plane. This is important so that the algorithm can move in any direction.

c) When starting in $[1.2 \quad 1.2]^T$ the average function value quickly goes to close to 0, but uses a long time to terminate. It has presumably fallen into the banana valley and slowly creeps towards the optimum since the function is quite flat there.

When starting in $[-1.2 \quad 1]^T$ the average function value quickly falls towards 4 then goes slowly towards 0. This is presumably also because it entered banana valley. When in the valley, it goes a lot back and fourth showing how the flatness of the Rosenbrock function plays a huge role. The algorithm also used twice as many iterations as in the last one.

d) Compared to steepest descent, Nelder-Mead uses fewer iterations and reaches a smaller function value in both starting points.

Both Newton and BFGS beat Nelder-Mead in terms of iterations and final function value. They also beat Nelder-Mead in execution time. But, maybe the story would be different if Newton and BFGS had to numerically calculate the derivatives and hessians?.