Optimization, Quasi-Newton methods
FMA051

Jonathan Andersson (mat11ja1@student.lu.se) - 890812 ****
Adam Jalkemo (adam@jalkemo.se) - 910504 ****
Emil Westenius

January 22, 2017

1 Code discussion

1.1 Line search

In order to avoid having to use any derivatives we decided to use the Golden section line search method. First problem is to be able to decide a good interval that is big enough to include the minimum but small enough to not give any numerical problems. To do this we started at two initial points [-10 10] and then checked that these will result in an acceptable value for each of the directions. If it is not acceptable for some direction we decrease the interval in that direction by a factor of 10 and if it is acceptable we check if increasing the interval by factor 2 in a direction will give a larger or smaller functional value, if it is smaller we keep increasing the interval until the functional value gets larger and the end point is acceptable. We found this method to be successful for all our test. Another problem that would arise was when the function was not unimodal, the interval could sometimes skip past the minimum to converge to a local minimum. To keep this from happening we added a condition to check if it would be better to not take a step at all. With this condition we managed to find the minimum for all our tests. To begin with we ran the method until our interval became smaller than the tolerance, however this gave us some numerical problems for test functions where the minimum was on really large or small points. So instead we looked at the difference of the functional values of the end points for the interval as well as the functional value in the middle of the interval, to make sure we do not get convergence on two different points that just have the same functional value. With these modifications and the criterion we managed to get convergence for all the test functions that we ran. For the test function we got the minimal value -1 found in 0.0025s for [0,1] and -1 found in 0.001s for [1,0].

1.2 Stop conditions

1.3 Line search

In the line search we chose the stopping criterion which had two aspects. Firstly, the difference between the functional values on the endpoints of the interval need to be within the tolerance range. Secondly, the difference between the center of the interval and the average value of the endpoints needs to be within the interval. See below for the stopping function

$$while\ abs(f(a) - f(b)) > tol\ \&\&\ abs(f((a + b)/2) - (f(a) + f(b))/2) > tol$$

1.4 Penalty function

The stopping criterion used in the outer loop is instead to stop when either the difference between the found points are within the tolerance region or the difference between the functional values for subsequent points is small enough. We chose to do use either of the two differences since we mostly will stop when the difference between the functional values is small and only rarely need to look at the difference between the points. The difference between the points will be lower than the tolerance before the functional value only if we are trying to find a very narrow and steep peak. For the problems we have implemented we can only see a huge difference for the problem where we minimize $\min e^{x_1x_2x_3x_4}$, where there is a factor of 10 times more iterations if both conditions needs to be met. For the other problems there is at most a difference of a couple iterations.

2 Problems

2.1 Rosenbrock

When minimizing the rosenbrock problem using the tolerance $tol = 10^{-6}$, $[200, 200]$ as a starting point and the DFP algorithm we need 51 outer iterations to find the minimum. The BFGS algorithm requires the same amount of iteration but yields a slightly lower functional value. Doing the minimization from the other side ($[-200, -200]$) yields a solution much faster, 14 iterations. Selecting initial point closer to the optimal point yields a solution faster, as expected.
When selecting $x_1 = 1$ and x_2 to be a negative value larger than 20 we see that after one iteration the algorithm stops regardless of chosen tolerance. We think this is due to the rosenbrock function and not that the algorithm misbehaves.

2.2 Problem from project description

The problem is formulated as

$$\min e^{x_1x_2x_3x_4}$$

subject to

$$x_1^2+x_2^2+x_3^2+x_4^2+x_5^2=10,\ x_2x_3=5x_4x_5,\ x_1^3+x_3^3=-1$$

This yields the penalty function

$$\alpha(\mathbf{x})=(x_1^2+x_2^2+x_3^2+x_4^2+x_5^2-10)^2+(x_2x_3-5x_4x_5)^2+(x_1^3+x_3^3+1)^2$$

and the auxiliary problem

$$\min q(\mathbf{x},\mu)=e^{x_1x_2x_3x_4}+\mu\alpha(\mathbf{x})$$

When searching for starting points with different outcomes we can find that by selecting

$$x_1=\begin{bmatrix}-2\\2\\2\\-1\\-1\end{bmatrix}\ x_2=\begin{bmatrix}-2\\2\\-2\\-1\\1\end{bmatrix}$$

we get the optimal points and values being

$$x_1^*=\begin{bmatrix}-1.7172\\1.8272\\1.5957\\-0.7637\\-0.7637\end{bmatrix},f(x_1^*)=0.0539,\ x_2^*=\begin{bmatrix}-0.6996\\2.7896\\-0.8703\\-0.6969\\0.6969\end{bmatrix},f(x_2^*)=0.4383$$

Both of the optimizations yield the same result with DFP and BFGS update schemes using tolerance 10^{-6}. Since we can force the optimization to yield the same result when manipulating the $\mu-$vector we can assume that the optimization gets stuck on a local minima for x_2 with the given $\mu-$vector. The DFP and BFGS update schemes behave very similar for almost all different starting points. There are a few different ways of finding starting point which will work for the implementation of this optimization algorithm. By using the given starting point x_1 as base one can look at the constraints and make sure that the signs change equally for the second and third constraint. Some of these points gives the same optimal point and some does not. Most points are x_1^* or x_2^*.
When lowering the tolerance one yields a similar result as when one extended the $\mu-$vector. This can be done for many different starting point in order to get the algorithm to converge.

2.3 Exercise 9.3

The problem formulation is

$$\begin{aligned}\underset{\mathbf{x}}{\text{minimize}}\quad & e^{x_1}+x_1^2+x_1x_2\\ \text{subject to}\quad & 1/2\cdot x_1+x_2=1.\end{aligned}$$

and a penalty on the form $\alpha(x)=(1/2\cdot x_1+x_2-1)^2$ is chosen so that the function to be minimized is:

$$q(x,\mu)=e^{x_1}+x_1^2+x_1x_2+\mu(1/2\cdot x_1+x_2-1)^2$$

2.3.1 Optimal points and function value

The optimal points found is $\mathbf{x}=(-1.278,1.639)$ using the tolerance 1e-6. with the objective function minimum -0.183. Both DFP and BFGS methods converge to the same points with a difference $<10^{-9}$. The series μ used was $(1,10,100,1000,10000,100000,1000000)$. It normally takes two outer iterations for the algorithm to stop.
The program execution converges to this feasible solution for a wide choice of μ in the penalty function. When μ is chosen small e.g $\mu=0.1$ the solver runs into trouble, this is due to a minimum not existing. For large initial μ one can expect the conditioning of the problem cause problems for the solver, however, this problem is not very sensitive to the selection of μ and for DFP using a $\mu=10^{18}$ and BFGS $\mu=10^{16}$ still manages to converge. Selecting a larger initial μ also affects the number of outer iterations necessary for both methods.

2.3.2 Typical output

Typical outputs for solving using BFGS with only $\mu = 100$

```
Outer-Iteration 1
It  x  s.s  f(x)  |grad|  l.s.i.  lambda
1  0.39  0.89  1.96  2.57  40  3.98E-03
  0.80
It  x  s.s  f(x)  |grad|  l.s.i.  lambda
2  -1.28  1.88  -0.19  0.38  25  7.33E-01
  1.65
...
...
Outer-Iteration 3
It  x  s.s  f(x)  |grad|  l.s.i.  lambda
1  -1.28  0.00  -0.19  0.00  0  0.00E+00
  1.65
It  x  s.s  f(x)  |grad|  l.s.i.  lambda
2  -1.28  0.00  -0.19  0.00  0  0.00E+00
  1.65
Within tolerance, done!
Penalty/Barrier: 1.00E+02, function value: -1.87E-01
Found point y: (-1.283, 1.648)
Minimum value: -0.183
Penalty at y: 3.97E-13
Total iterations 3
```

2.4 Exercise 9.5

For exercise 9.5 in the book the problem is formulated as

$$\begin{aligned} \underset{\mathbf{x}}{\text{Minimize}} \quad & (x_1 - 5)^2 + (x_2 - 3)^2 \\ \text{subject to} \quad & x_1 + x_2 \leq 3 \\ & \text{-}x_1 + 2x_2 \leq 4. \end{aligned}$$

The barrier function is chosen as $\beta = \frac{1}{3 - x_1 - x_2} + \frac{1}{4 + x_1 - 2x_2}$ and to hinder the line search from missing the barrier a term was added that becomes large for points outside of the domain. That gives us the auxiliary problem to be solved as minimizing

$$q_\epsilon = (x_1 - 5)^2 + (x_2 - 3)^2 + \epsilon\left(\frac{1}{3 - x_1 - x_2} + \frac{1}{4 + x_1 - 2x_2}\right) + 10^{10}(max(0, x_1 + x_2 - 3) + max(0, \text{-}x_1 + 2x_2 - 4))$$

.

2.4.1 Optimal points and function value

When using a sequence ϵ= [1, 0.1, 0.01, 0.001, 0.0001, 0.00001, 0.000001], tolerance 1e-6 and starting point [0,0], the solver finds the optimal point [2.500, 0.500] with the functional value 12.502 for both the BFGS and DFP where the difference of the solutions is 4.170e-08 for the point and 7.578e-11 for the functional value, both methods takes 8 iterations. The real minimum lies at [2.5, 0.5] and both methods have an distance of around 3e-4 from this point.
How well the solver works and how many iterations it takes will of course depend on what tolerance and sequence of ϵ that is chosen. Starting with a low epsilon we have the risk of getting an ill-conditioned problem and with a large we may never converge. Both BFGS and DFP manages to get feasible solutions for starting epsilon as low as 1e-8 and as big as 1e5. Overall the result of the solver is satisfactory for both methods and the difference between them is small.

3 Solver evaluation

3.1 Consistency in the program behaviour

The algorithm behaves quite similarly for the different problems we have tested it on. However, as expected, some starting points yields weird results such as the algorithm taking zero-length steps and so on. When selecting too large or small μ/ϵ the algorithm does not converge at all.

3.2 Comparison between DFP and BFGS