

**4-6TE3: Assignment 2.**  
**Christopher Anand, Shefali Kulkarni-Thaker**  
15 October 2011, due 11 November 2011 in class

---

**Submission Instructions:**

- Use **MATLAB** for all questions.
  - Compress all of your code files into a file **StudentID.zip** (which can be uncompressed by WinZip or 7-zip (<http://www.7-zip.org>) software under Windows), and email to [cs4te3@cas.mcmaster.ca](mailto:cs4te3@cas.mcmaster.ca) with the subject "StudentID, Assignment 2" no later than 12 p.m. Nov. 11, 2011.
  - In the compressed file, include instructions in a **readme.txt** file on how to run the code.
  - Hard-copy of the discussions and comparative analysis of your results.
- 

Consider the following functions

- Powell's Quadratic Function ( $n = 4$ ). Minimum at  $f(0, 0, 0, 0) = 0$

$$y = (x_1 + 10x_2)^2 + 5(x_3 - x_4)^2 + (x_2 - 2x_3)^4 + 10(x_1 - x_4)^4 = 0$$

- Fletcher's and Powell's Helical Valley ( $n = 3$ ). Minimum at  $f(1, 0, 0) = f(-1, 0, 0) = 0$

$$y = 100 \left( x_3 - \frac{5}{\pi} \operatorname{atan} \left( \frac{x_2}{x_1} \right) \right)^2 + \left( \sqrt{x_1^2 + x_2^2} - 1 \right)^2 + x_3^2$$

- Beale Function ( $n = 2$ ). Minimum in the region of  $-4.5 \leq x_i \leq 4.5$  is  $f(3.025, 0.474) = 0.038$

$$y = \left( 1.5 - x_1 + x_1x_2 \right)^2 + \left( 2.5 - x_1 + x_1x_2^2 \right)^2 + \left( 2.625 - x_1 + x_1x_2^3 \right)^2$$

---

1. Review and implement BFGS, Trust Region, and classical Newton's (without line search) method in MATLAB for the given functions. Use the line search of your choice (Implemented Wolfe-line search method is given on the course web-page.) and justify its usage. Discuss and compare your implementation on the given test functions. Choose different starting points and create a comparative table describing the number of iterations, function evaluations, norms, and CPU time.

Justify the use of one method over another for each of the function. Is your choice any better than MATLAB's `fminunc` or `fminsearch`

Develop an approach to unit-testing of your implementation, and describe your reasons for using this particular unit testing.

Include a (small) test suite, with a description of the design of your test suite. What kind of functions should you include in your test suite. How can you leverage other tools, other optimization methods, analytical solutions?

## Programming Instructions:

Test the given 3 functions with atleast 2 initial points. Your program must be well-structured and the screen output must contain initialization, algorithm and statistics sections. In the initialization section the program should provide the values of variables. In the algorithm section your output information on each iteration. Statistics section gives information on the point found, number of iterations and function evaluations, last values for argument, function, gradient precision, etc. **Failure to meet the specifications will result in reduced mark.** Sample printouts are given for the Rosenbrock's banana function. Your program must always terminate (no infinite loops). It should be stopped if any of the following criteria is satisfied:

- Maximum number of iterations is exceeded
- Maximum number of function evaluations is exceeded
- Argument and function value do not change more than `epsx` and `epsfun`, and gradient norm is less than `epsgrad`
- Hessian becomes singular (for Newton method only).
- $Hessian + \alpha * I$  becomes singular (for Trust-Region method only)

Definition of `function`

```
function [minx, minf, iter, funev] = newton( ffunc, gfunc, hfunc, n, x0, epsfun,
    epsgrad, epsshess, epsx, maxiter, maxfun)
```

Output

```
>> newton('fbanana', 'gbanana', 'hbanana', 2, [5 2]')
```

Newton Method

---

1.Initialization

```
epsfun (default = 0.000001): 1.000000e-006
epsgrad (default = 0.000001): 1.000000e-006
epshess (default = 0.000001): 1.000000e-006f
epsx (default = 0.000001): 1.000000e-006
maxiter (default = 200): 200
maxfun (default = 100000): 100000
Iteration 1: banana(5.000000e+000,2.000000e+000) = 5.291600e+004
```

---

2.Algorithm

```
Iteration 1: banana(4.999131e+000,2.499131e+001) = 1.599305e+001
Iteration 2: banana(1.000604e+000,-1.498700e+001) = 2.556229e+004
Iteration 3: banana(1.000604e+000,1.001209e+000) = 3.651063e-007
Iteration 4: banana(1.000000e+000,9.999996e-001) = 1.333026e-011
Iteration 5: banana(1.000000e+000,1.000000e+000) = 8.077936e-028
Terminate: Desired Precision Reached.
```

---

3.Statistics

```
Minimum Found
Minimum Function Value: 0.000000
2-norm of xk-xk1: 0.000000
|fk-fk1|/(1+|fk1|): 0.000000
2-norm of gradient: 0.000000
```

```

Minimum eigenvalue of Hessian: 0.399390
CPU Time: 0.008777
Total Iterations: 5
Total Function Evaluations: 16

```

```
ans =
```

```

1.0000
1.0000

```

	<b>Newton's</b>	<b>Trust Region</b>	<b>BFGS</b>
<b>Precision</b>	1.00E-06	1.00E-06	1.00E-06
<b>f(x*)</b>	0	8.71E-20	8.05E-19
<b>x*</b>	(1,1)	(1,1)	(1,1)
<b>Iterations</b>	5	43	38
<b>Function Evals</b>	16	130	78
<b>CPU time</b>	0.001014	0.006721	0.013086

Table 1: Sample Comparative Table for Rosenbrock Function  $x_0 = [5,2]$