Name:Chao Jiang

ISE 5643 — Spring 2015 **HW 5** Due: Mar. 25

Problem 1

Consider the minimization of the following function

$$f(x) = (x_1^2 + x_2 - 11)^2 + (x_1 + x_2^2 - 7)^2$$

Perform two iterations to illustrate the following methods:**(20 pts)**

(a) Cauchy

(b) Newton

(c) Modified Newton

(d) Marquardt ($\lambda^0 = 100$)

Use $x^{(0)} = [1, 1]^T$ as a starting point. Compare the final objectives and variable values after two iterations.

(a) Cauchy

To begin, calculate the

$$\bigtriangledown f(x) = \left(4x_1^3 + 4x_1x_2 + 2x_2^2 - 42x_1 - 14, 4x_2^3 + 4x_1x_2 + 2x_1^2 - 26x_2 - 22\right)$$

Then, $x^{(k+1)} = x^{(k)} - \alpha^{(k)} \bigtriangledown f(x^{(k)})$. So we need to calculate the α firstly, $\phi(\alpha) = f(x^k - \alpha^{(k)} \bigtriangledown f(x^k))$ is minimum along the $f(x^k - \alpha \bigtriangledown f(x^k))$.

$$\phi'(\alpha) = - \bigtriangledown f(x^0 - \alpha \bigtriangledown f(x^0)) \cdot \bigtriangledown f(x^0) = 0$$

Solving the above equation, we may get more than one α; so after verifying the $f(x^k)$ we choose the optimal α.

Cauchy Method

k	$x_1^{(k)}$	$x_2^{(k)}$	$f(x^{(k)})$	step
1	2.7121	2.41435	3.88848	0.0372196
2	3.32252	2.14061	5.56999	0.0520706

(b) Newton

The iteration equation is $x^{(k+1)} = x^{(k)} - H^{(-1)} * \bigtriangledown f(x^{(k)})$

Newton Method

k	$x_1^{(k)}$	$x_2^{(k)}$	$f(x^{(k)})$
1	-2.89796	-5.91837	704.066
2	-5.80609	-4.58747	396.328

(c) Modified Newton

The iteration equation is $x^{(k+1)} = x^{(k)} - \alpha^{(k)} H^{-1} \bigtriangledown f(x^{(k)})$. The difference is that we should computing the α which related with both $\bigtriangledown f(x^{(k)})$ and H^{-1}.

Modified Newton Method

k	$x_1^{(k)}$	$x_2^{(k)}$	$f(x^{(k)})$	step
1	1.9686	2.71914	24.9887	-0.24849
2	2.96589	2.38226	2.72502	0.301103

(d) Marquardt ($\lambda^0 = 100$)

The iteration equation is $x^{(k+1)} = x^{(k)} - [H^{(k)} + \lambda^{(k)} I]^{-1} \bigtriangledown f(x^{(k)})$

Marquardt Method

k	$x_1^{(k)}$	$x_2^{(k)}$	$f(x^{(k)})$
1	1.58156	1.37053	63.3424
2	2.67443	1.76319	5.82444

Problem 2

Consider the minimization of the following functions:

$$\begin{aligned} f(x) &= (x_1^2 + x_2 - 11)^2 + (x_1 + x_2^2 - 7)^2 \text{ and the starting point } x^{(0)} = [0,0]^T \\ g(x) &= 100(x_2 - x_1^2)^2 + (1 - x_1)^2 \text{ and the starting point } x^{(0)} = [-1.2, 0]^T \\ h(x) &= 2x_1^3 + 4x_1 x_2^3 - 10x_1 x_2 + x_2^2 \text{ and the starting point } x^{(0)} = [5,2]^T \end{aligned}$$

Write MATLAB codes to find a minimum to each of the above functions by using the following methods: **(50 pts)**

(a) Cauchy

(b) Newton

(c) Modified Newton

(d) Marquardt ($\lambda^0 = 100$)

For stopping criteria use the $|f'(x_i)| < 0.0001$. Use Golden section search for line search with stopping criteria $|b - a| < 0.0001$.

(a) For function $f(x)$:

By running the code, set up the following table:

Cauchy Method

k	$x_1^{(k)}$	$x_2^{(k)}$	$f(x^{(k)})$	step
1	1.78284	2.80161	32.1257	0.127346
2	3.00126	2.02534	0.011745	0.0399026
3	2.99206	2.01154	0.00276918	0.0152977
4	3.00028	2.00606	0.000662725	0.0231743
5	2.9981	2.00279	0.000160301	0.0153888
6	3.00007	2.00147	3.88923e-05	0.0231743
7	2.99954	2.00068	9.46897e-06	0.0154236
8	3.00002	2.00036	2.30999e-06	0.023118
9	2.99989	2.00017	5.64903e-07	0.0154799
10	3	2.00009	1.38901e-07	0.0229706
11	2.99997	2.00004	3.41799e-08	0.0155362
...	...	...	...	...
14	3	2.00001	5.13471e-10	0.0229358
15	3	2	1.26575e-10	0.0155362

Newton Method

k	$x_1^{(k)}$	$x_2^{(k)}$	$f(x^{(k)})$
1	-0.333333	-0.846154	181.501
2	-0.270761	-0.919151	181.616
3	-0.270846	-0.923028	181.617

Modified Newton Method

k	$x_1^{(k)}$	$x_2^{(k)}$	$f(x^{(k)})$	step
1	1.11629	2.83366	52.4946	-3.34887
2	2.8614	2.62674	7.6585	-1.15335
3	3.7967	-1.70316	3.02161	9.60025
4	3.59173	-1.87358	0.011134	1.05854
5	3.58436	-1.84819	3.40879e-07	1.01802
6	3.58443	-1.84813	4.40801e-16	1.00001

Marquardt Method

k	$x_1^{(k)}$	$x_2^{(k)}$	$f(x^{(k)})$
1	0.241379	0.297297	157.796
2	2.42456	1.23931	24.3102
3	3.04524	1.74563	0.824015
4	3.02173	1.93711	0.0556356
5	3.00446	1.98948	0.00167223
6	3.00047	1.99891	1.79916e-05
7	3.00003	1.99994	5.78119e-08
8	3	2	5.0124e-11

(b) For function $g(x)$:

Cauchy Method

k	$x_1^{(k)}$	$x_2^{(k)}$	$f(x^{(k)})$	step
1	0.993336	0.986649	4.48599e-05	0.0100092
2	0.993321	0.986664	4.4658e-05	0.0011011
3	0.993381	0.986731	4.43783e-05	0.0144136
4	0.993363	0.986748	4.41013e-05	0.0011011
5	0.993408	0.986792	4.39025e-05	0.010044
...	...	...	...	...
2051	0.999893	0.999785	1.1489e-08	0.0011011
2052	0.999894	0.999786	1.14406e-08	0.00941973
2053	0.999893	0.999786	1.13926e-08	0.0011011

Newton Method

k	$x_1^{(k)}$	$x_2^{(k)}$	$f(x^{(k)})$
1	-1.19239	1.42173	4.80656
2	0.974882	-3.74666	2206.23
3	0.974908	0.950446	0.000629594
4	1	0.99937	3.96389e-05
5	1	1	1.6032e-19

Modified Newton Method

k	$x_1^{(k)}$	$x_2^{(k)}$	$f(x^{(k)})$	step
1	-1.19239	1.42187	4.80656	1.0001
2	-0.977638	0.909734	4.12305	0.0963514
3	-0.70521	0.44181	3.21588	1.40628
...	...	...	...	...
11	0.999945	0.999893	3.92149e-09	0.981405
12	1	1	1.31214e-16	0.999325

Marquardt Method

k	$x_1^{(k)}$	$x_2^{(k)}$	$f(x^{(k)})$
1	-0.978908	0.606252	16.307
...	...	...	...
19	0.994799	0.989587	2.7193e-05
20	0.99981	0.999594	1.02248e-07
21	0.999996	0.999992	1.58327e-11

(c) For function $h(x)$:

Cauchy Method

k	$x_1^{(k)}$	$x_2^{(k)}$	$f(x^{(k)})$	step
1	0.731023	-3.11223	-54.9284	0.0263517
2	1725.76	-1431.78	-2.0251e+13	20
3	2.34452e+11	-8.49065e+11	-5.74032e+47	20
4	4.89679e+37	-4.05645e+37	-1.3074e+151	20
5	5.33982e+114	-1.93381e+115	NaN	20
6	Inf	-Inf	NaN	20

Newton Method

k	$x_1^{(k)}$	$x_2^{(k)}$	$f(x^{(k)})$
1	2.56562	1.58061	36.2467
2	1.49956	1.24713	1.23284
3	1.13858	0.983523	-2.94596
4	1.02534	0.858311	-3.31463
5	1.00243	0.834343	-3.32408
6	1.00156	0.833452	-3.32409

Modified Newton Method

k	$x_1^{(k)}$	$x_2^{(k)}$	$f(x^{(k)})$	step
1	-43.6876	-6.38787	-123965	20
2	-472.41	-38.2873	-1.04977e+08	-20
3	-3534.71	-171.873	-1.65467e+10	-13.4643
4	-11282.3	-382.696	-3.42895e+11	-4.64516
5	-29940.3	-739.631	-5.22071e+12	-3.51912
6	-74656.3	-1364.19	-7.40683e+13	-3.18234
...	...	...	...	...
262	-1.22972e+102	-8.84905e+67	-3.10767e+305	-2.97958
263	-2.94792e+102	-1.58502e+68	-4.28115e+306	-2.97958
264	-7.06682e+102	-2.83906e+68	NaN	-2.97958

Marquardt Method

k	$x_1^{(k)}$	$x_2^{(k)}$	$f(x^{(k)})$
1	4.09843	1.53292	136.26
2	3.14614	1.23546	48.671
3	2.30023	1.05589	11.9997
...	...	...	...
8	1.00223	0.833612	-3.32409
9	1.00158	0.833456	-3.32409

Problem 3

Write a computational analysis report for question number 2 discussing the performance of each method on the three functions and relative comparisons of directions used and step sizes. Comment on worst/best performance of each method in general. **(15 pts)**

By creating the table below, the first value of each item is the iteration numbers for each method, the "a¡b¡c¡d" stand for the final minimal value of each method, if one method get an "a" in this function, which means this method get a more small final value compared with other methods. From the performance of iteration numbers and

Performance of Four Methods

	Cauchy	Newton	Modified Newton	Marquardt ($\lambda^0 = 100$)
$f(x)$	15/b	3/c	6/a	8/b
$g(x)$	2053/a	5/a	12/a	21/a
$h(x)$	6/a	6/b	264/a	9/b

accuracy, we know that Cauchy method sometimes need many iterations to converge. and Modified Newton Method will always get a good result. So the best performance of these methods is Modified Newton Method, and the worst method is Newton Method considering it sometimes will get a local minimal.

Problem 4

Use your codes to solve $f(x)$ problem number 2 with starting points $x^{(0)} = [6, -6]^T, x^{(0)} = [-5, -4]^T, x^{(0)} = [-4, 6]^T$ for Cauchy and Newton methods. What are your optimal solutions? Are they different with that in number 2? **(15 pts)**

(a) For Cauchy Method:

Cauchy Method

$x^{(0)}$	x_1	x_2	$f(x^{(k)})$
$[6, -6]^T$	3.58443	-1.84813	1.37512e-10
$[-5, -4]^T$	3.58443	-1.84813	1.17468e-11
$[-4, 6]^T$	-2.80512	3.13131	1.38171e-12

(b) For Newton Method:

Newton Method

$x^{(0)}$	x_1	x_2	$f(x^{(k)})$
$[6, -6]^T$	3.58443	-1.84813	4.85478e-18
$[-5, -4]^T$	-3.77931	-3.28319	2.69059e-12
$[-4, 6]^T$	-2.80512	3.13131	2.37243e-11

My optimal solutions is Newton Method given start point $[6, -6]^T$, the final result is $4.85478e-18$. They are almost the same as the result of Modified Newton Method given start point $[0, 0]^T$.

Matlab Code

All related code files can be download at my GitHub[1].

Cauchy:

```
% Cauchy Method x^(k+1)=x^(k)-alpha^(k)*gradient(x^(k))
% -gradient(x^(k)-alpha*gradient(x^(k)))*gradient(x^(k))
% (i,j) start point
i=6;
j=-6;
syms a b;
grad(a,b)=gradient(f(a,b));
epsilon=0.0001;
while (norm(grad(i,j))>=epsilon)
     step = Golden(i,j);
    %fprintf(1,'step=%g\n', double(step));
    X = double([i j]'-step*grad(i,j));
     fprintf(1,'%g %g %g %g\n',double(X(1)),double(X(2)),double(f(X(1),X(2))),double(step));
     i=X(1);
     j=X(2);
end

function step = Golden(i,j)
% initial parameters

syms s Y a b;
grad(a,b)=gradient(f(a,b));
Y=[i j]' - s*grad(i,j);
y(s) = f(Y(1),Y(2));

a = -20;                                          % start of interval
b = 20;                                           % end of interval
epsilon = 0.0001;                               % accuracy value
```

[1] https://github.com/chao92/Engineering-Optimization

```
ratio = double((sqrt(5)-1)/2);                    % golden proportion coefficient, aroud 0.618
%k = 0;                                            % number of iterations
%iter = 500;                                       % maximun number of iterations

len = b-a;
x1 = b - ratio * len;                             % computing x1;
x2 = a + ratio * len;                             % computing x2;
f1=y(x1);
%f1=f(i-x1*[1 0]*grad(i,j),j-x1*[0 1]*grad(i,j));
f2=y(x2);
%f2=f(i-x2*[1 0]*grad(i,j),j-x2*[0 1]*grad(i,j));

% search
while(((b-a) > epsilon))

    % evaluate f(x1) and f(x2)
    if(f1<f2)                                     % if f(x1)<f(x2), then drop interval (x2,b);
            b = x2;                               % repalce b with x2;
            len = x2-a;                           % length becomes b-a=x2-a;
            x2 = x1;                              % replace the next iteration x2 with x1;
            x1 = b - ratio * len;                 % computing x1 using the equation of computing x1;
            f2=f1;
            f1=y(x1);
            %f1=f(i-x1*[1 0]*grad(i,j),j-x1*[0 1]*grad(i,j));
    else

            a = x1;
            len = b -a;
            x1 = x2;
            x2 = a + ratio * len;
            f1=f2;
            f2=y(x2);
            %f2=f(i-x2*[1 0]*grad(i,j),j-x2*[0 1]*grad(i,j));
```

```
    end

end
 step = (a+b)/2;
end
```

Newton:

```
% Newton's Method x^(k+1)=x^(k)-H^(-1)*gradient(x^(k))
% (i,j) start point
i=5;
j=2;
syms a b;
epsilon=0.0001;
grad(a,b)=gradient(f(a,b));
hess(a,b)=hessian(f(a,b));
while (abs(grad(i,j))>=epsilon)
X= [i j]'-inv(hess(i,j))*grad(i,j);
fprintf(1,'%g %g %g\n',double(X(1)),double(X(2)),double(f(X(1)
    ,X(2))));
i=X(1);
j=X(2);
end
```

Modified Newton:

```
% (i,j) start point
i=-1.2;
j=0;
syms a b;
epsilon=0.0001;
grad(a,b)=gradient(f(a,b));
hess(a,b)=hessian(f(a,b));
while (norm(grad(i,j))>=epsilon)
     step = Golden_M(i,j);
    %fprintf(1,'step=%g\n', double(step));
    X = double([i j]'-step*inv(hess(i,j))*grad(i,j));
     i=X(1);
     j=X(2);
     fprintf(1,'%g %g %g %g\n',double(i),double(j),double(f(i,j
        )),double(step));
end
```

Marquardt:

```
% (i,j) start point
```

```
i=5;
j=2;
iter=1000; % maximum number of iteratios allowed
lamda = 100;
syms a b X Y Z;
epsilon=0.0001;
grad(a,b)=gradient(f(a,b));
hess(a,b)=hessian(f(a,b));
I = [1 0; 0 1];
x(1)=i;
x(2)=j;
k=0;
while ((abs(grad(i,j))>=epsilon))
X = -inv(hess(i,j)+lamda*I)*grad(i,j);
tem=[i j]'-inv(hess(i,j)+lamda*I)*grad(i,j);
tem = double(tem);
    if(f(tem(1),tem(2))<f(x(1),x(2)))
        lamda = 0.5 * lamda;
        k = k + 1;
       x(1)=real(tem(1));
        x(2)=real(tem(2));
        i = x(1);
        j = x(2);
        fprintf(1,'%g %g %g\n',double(x(1)),double(x(2)),double(f(x(1),x(2))));
    else
        lamda = 2 * lamda;
    end
end
fprintf(1,'%g %g %g\n',double(x(1)),double(x(2)),double(f(x(1),x(2))));
```