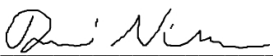


Compulsory Assignment in Optimization Fall 2014

Course quarter: Q1, 2014

Study number: 10783

Name of Student: René Nilsson

Signature: 

Submission date: 21/9-2014

You must confirm to the following regulations

Regulations

Each student must hand in satisfactory solutions to compulsory assignment in order to take the oral exam. . It should be completed individually (in Danish or English) and handed in by email. A solution from a group of students will not be accepted. Feel free to write your assignment by hand, and hand in a scanned version of your work.

Contents

1	Exercise 1	2
2	Exercise 2	4
3	Exercise 3	6
4	Exercise 4	8
5	Exercise 5	12
6	Exercise 6	14

1 Exercise 1

Optimization using the simplex method:

% The augmented matrix:

clc; clear;

```
A = [1  2  0  1  0  0  0  28;
      2  0  4  0  1  0  0  16;
      0  1  1  0  0  1  0  12;
     -2 -5 -3  0  0  0  1  0]
```

A =

1	2	0	1	0	0	0	28
2	0	4	0	1	0	0	16
0	1	1	0	0	1	0	12
-2	-5	-3	0	0	0	1	0

We want to bring x_2 in, since $A(4,2) < A(4,3) < A(4,1)$, and thus increases $A(4,8)$ the most. The pivot row should be $A(3,2)$, since

$$\frac{A(3,8)}{A(3,2)} = 12 < \frac{A(1,8)}{A(1,2)} = 14$$

```
A(1,:) = A(1,:)-2*A(3,:);
```

```
A(4,:) = A(4,:) + 5*A(3,:)
```

A =

1	0	-2	1	0	-2	0	4
2	0	4	0	1	0	0	16
0	1	1	0	0	1	0	12
-2	0	2	0	0	5	1	60

Next, we bring in x_1 , since $A(4,1)$ is the only negative value in $A(4,:)$. $A(1,1)$ is the new pivot, since

$$\frac{A(1,8)}{A(1,1)} = 4 < \frac{A(2,8)}{A(2,1)} = 8$$

```
A(2,:) = A(2,:) -2*A(1,:);
```

```
A(4,:) = A(4,:) + 2*A(1,:)
```

A =

1	0	-2	1	0	-2	0	4
0	0	8	-2	1	4	0	8
0	1	1	0	0	1	0	12
0	0	-2	2	0	1	1	68

Last, we bring in x_3 , with pivot $A(2, 3)$, since $A(1, 3)$ is negative and

$$\frac{A(2, 8)}{A(2, 1)} = 1 < \frac{A(3, 8)}{A(3, 1)} = 12$$

```
A(2,:) = A(2, :)/8;
```

```
A(1,:) = A(1, :) + 2*A(2, :);
```

```
A(3,:) = A(3, :) - A(2, :);
```

```
A(4,:) = A(4, :) + 2*A(2, :)
```

```
A =
```

```
Columns 1 through 7
```

1.0000	0	0	0.5000	0.2500	-1.0000	0
0	0	1.0000	-0.2500	0.1250	0.5000	0
0	1.0000	0	0.2500	-0.1250	0.5000	0
0	0	0	1.5000	0.2500	2.0000	1.0000

```
Column 8
```

```
6.0000
1.0000
11.0000
70.0000
```

Thus the maximum of the problem is 70, which is achieved when $x_1 = 6$, $x_2 = 1$ and $x_3 = 11$.

2 Exercise 2

Minimizing the function $f(x) = x_1^2 + 3x_2^2 - 2x_1x_2 + 3x_2$. 1. Write f on the form: $\frac{1}{2}x^T Q x - x^T b$:

```
Q = [2 -2;-2 6]
b = [0;-3]
```

Q =

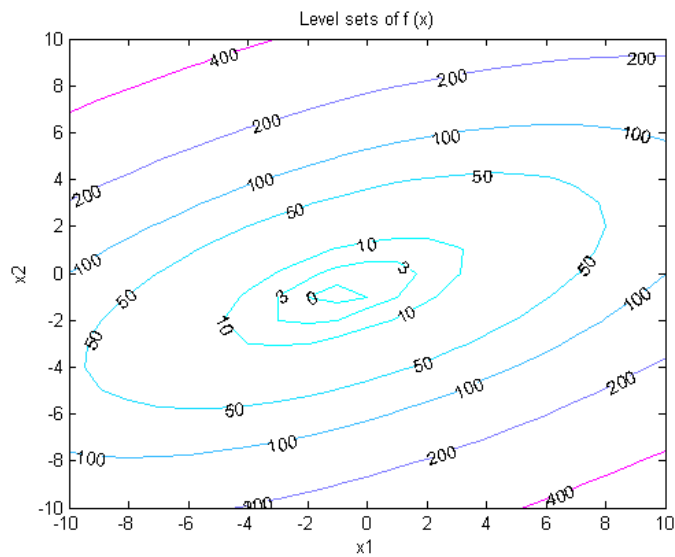
```
2    -2
-2     6
```

b =

```
0
-3
```

2. Sketch the levels set for f and the gradient of f in the point $(1, 1)^T$ in a x_1, x_2 -coordinate system.

```
[x1, x2] = meshgrid(-10:1:10,-10:1:10);
x = [x1;x2];
f1 = x1.^2+3*x2.^2-2*x1.*x2+3*x2;
[C,h1] = contour(x1,x2,f1,[0 3 10 50 100 200 400]);
set(h1,'ShowText','on','TextStep',get(h1,'LevelStep')*2)
colormap cool
title('Level sets of f (x)')
xlabel('x1')
ylabel('x2')
```



Gradient of f in the point $(1, 1)^T$:

```
g1 = Q*[1;1]-b
```

g1 =

0
7

3. Find all point satisfying the FONC. Do these points satisfy the SONC?

The gradient of f is found to be:

$$f'(x) = Q * x - b$$

Finding the point satisfying the FONC equals solving the following:

$$Q * x - b = 0$$

This gives the following point:

`x = inv(Q)*b`

x =

-0.7500
-0.7500

The Hessian is found to be $F(x) = Q$. Thus, the SONC is to test whether $Q > 0$:

`format short`
`eigs(Q)`

ans =

6.8284
1.1716

Since all eigenvalues of Q is positive, $Q > 0$ then the SONC is satisfied in all points satisfying the FONC.

4. Find the minimum of f over R^2

Since both the FONC and SOSC is satisfied by the previous piont, this is the global minimum:

x

x =

-0.7500
-0.7500

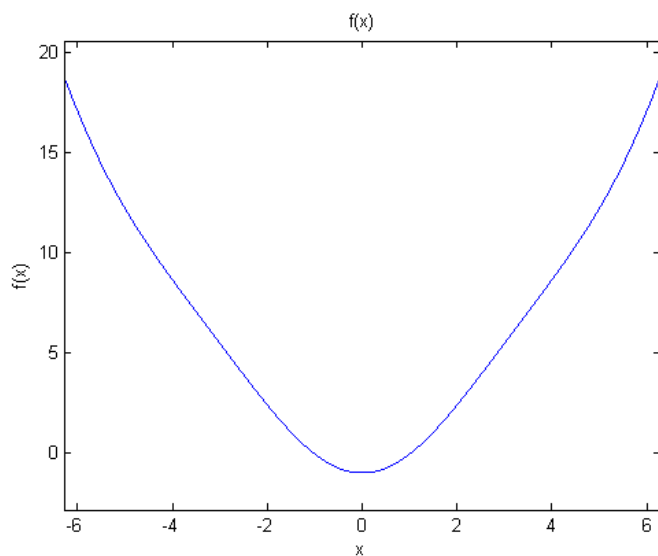
3 Exercise 3

Newtons method

Given the function $f(x) = \frac{1}{2} * x^2 - \cos(x)$.

1. Sketch the graph of f and find the 1. and 2. derivative of f :

```
syms x;  
f2(x) = 0.5*x.^2-cos(x);  
ezplot(f2)  
title('f(x)');  
ylabel('f(x)');  
xlabel('x');
```



First derivative:

```
g2(x) = diff(f2)
```

```
g2(x) =
```

```
x + sin(x)
```

Second derivative:

```
F2(x) = diff(g2)
```

```
F2(x) =
```

```
cos(x) + 1
```

2. Use Newtons method to find the minimizer of f , where:

```
x0 = 0.25;
```

Newtons method uses the formular

$$x^{(k+1)} = x^k - \frac{f'(x^k)}{f''(x^k)}$$

when both the first and second derivatives are known.

```
x1 = eval(x0 - g2(x0)/F2(x0))
x2 = eval(x1 - g2(x1)/F2(x1))
x3 = eval(x2 - g2(x2)/F2(x2))
```

```
x1 =
-0.0026
```

```
x2 =
3.0277e-09
```

```
x3 =
0
```

Note that the accuracy

```
er = abs(x3-x2)
```

```
er =
3.0277e-09
```

satisfy the needed accuracy of $e < 10^{(-5)}$. Thus, the needed number of iterations is 3.

4 Exercise 4

Steepest descent algorithm: Given the quadratic function from exercise 2:

```
syms xx1 xx2;  
x = [xx1;xx2];  
Q = [2 -2;-2 6];  
b = [0;-3];  
f3(x) = 0.5*x.'*Q*x-x.'*b;
```

The initial starting point is:

```
x0 = [1;1];
```

The gradient of f is:

```
g3(x) = Q*x-b;
```

To compute x^1 we need

$$\alpha_0 = \arg \min_{\alpha \geq 0} f(x^0 - \alpha \nabla f(x^0)) = \Phi(\alpha)$$

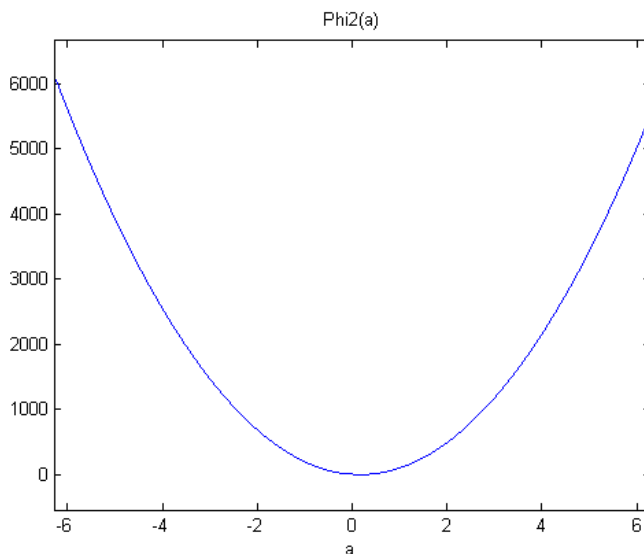
```
syms a;  
arg = x0-a*g3(x0(1),x0(2));  
Phi0(a) = f3(arg(1),arg(2));
```

To find α_0 we use the FONC and plot the graph of $\Phi(\alpha)$, to ensure it is a minimum

```
a0 = solve(diff(Phi0(a)) == 0)  
ezplot(Phi0(a))  
title('Phi2(a)');
```

```
a0 =
```

```
1/6
```



Now we find x^1 :


```
x1 = x0-a0*g3(x0(1),x0(2))
```

```
x1 =
```

```
1  
-1/6
```

Now we use same approach to find x^2 and x^3 :

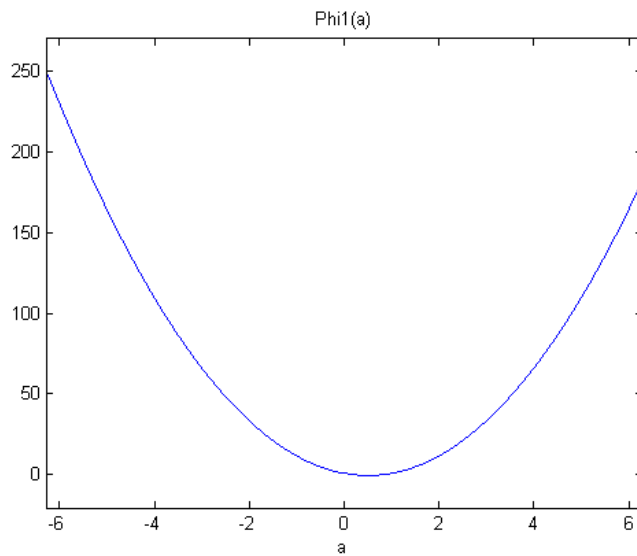
```
arg = x1-a*g3(x1(1),x1(2));  
Phi1(a) = f3(arg(1),arg(2));  
a1 = solve(diff(Phi1(a)) == 0)  
ezplot(Phi1(a))  
title('Phi1(a)');  
x2 = x1-a1*g3(x1(1),x1(2))
```

```
a1 =
```

```
1/2
```

```
x2 =
```

```
-1/6  
-1/6
```



```
arg = x2-a*g3(x2(1),x2(2));  
Phi2(a) = f3(arg(1),arg(2));  
a2 = solve(diff(Phi2(a)) == 0)  
ezplot(Phi2(a))  
title('Phi2(a)');  
x3 = x2-a2*g3(x2(1),x2(2))
```

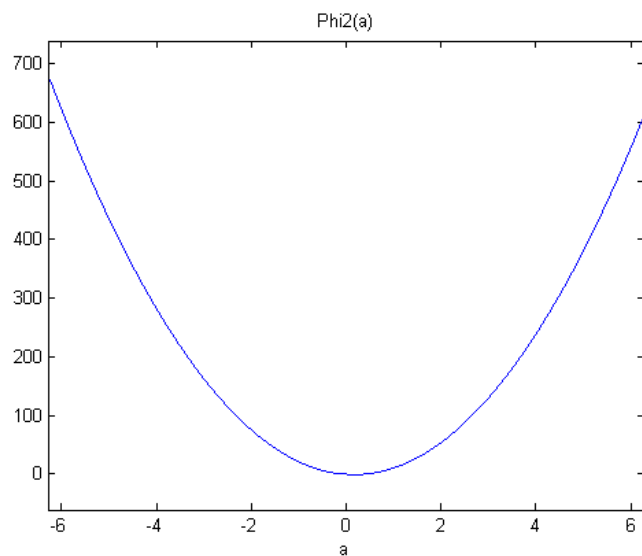
a2 =

1/6

x3 =

-1/6

-5/9

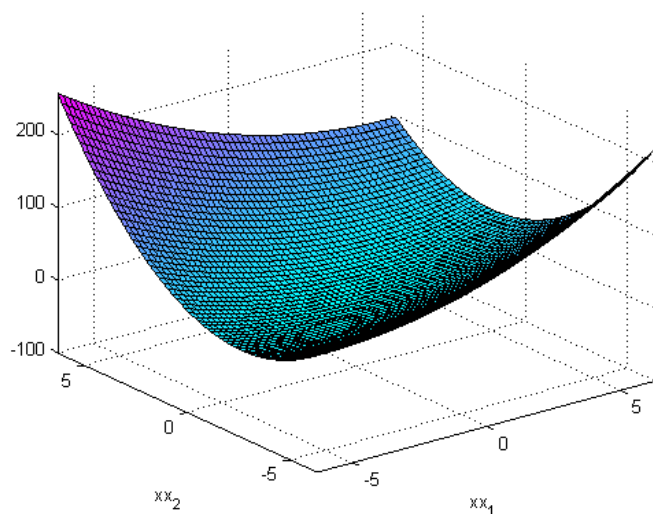


2. Does the steepest descent algorithm converge?

From Theorem 8.2 in the Chong book, the steepest descent algorithm will always converge. This is also clearly evident, when looking at the graph of $f(x)$:

`ezsurf(f3(xx1,xx2))`

$$3 \, xx_2 + xx_1 (xx_1 - xx_2) - xx_2 (xx_1 - 3 \, xx_2)$$



3. Use a fixed-step size algorithm, what step size should be used to ensure convergence?

Since Q is symmetric, we know that $f(x)$ converges if the following inequality is satisfied:

$$0 < \alpha < \frac{2}{\lambda_{\max}(Q)}$$

```
lambda_max = max(eigs(Q))
```

```
lambda_max =
```

```
6.8284
```

Thus, as long as $0 < \alpha < 6.8284$ the algorithm will converge.

5 Exercise 5

Data-fitting problem

1. Calculate and write out $J(x)$

Since the number of samples is unknown, we calculate $J(x)$ column-wise, as a function of t_i :

```
syms yi A ti s;
x = [A,s];
Jc1(x) = diff(yi-A*exp(-(ti^2)/(s^2)),A)
Jc2(x) = diff(yi-A*exp(-(ti^2)/(s^2)),s)
```

```
Jc1(A, s) =
```

```
-exp(-ti^2/s^2)
```

```
Jc2(A, s) =
```

```
-(2*A*ti^2*exp(-ti^2/s^2))/s^3
```

2. Fill in the missing code, and plot decision variables A and σ as a function of iteration number.

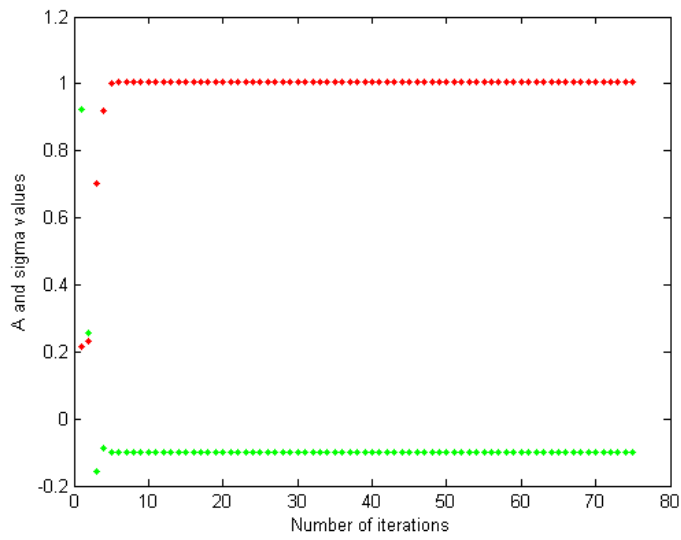
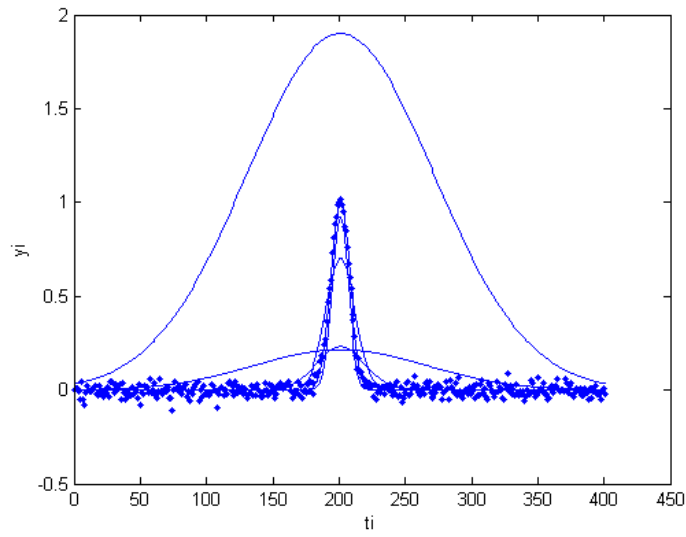
Simulation dataset

```
t=(-2:.01:2);
x_gold=[1;.1];
y=x_gold(1)*exp(-(t).^2/x_gold(2)^2);
y=y+.03*randn(size(y));
figure(1)
plot(y,'.b'); hold on;
xlabel('ti');
ylabel('yi');
xx=[];
x=x_gold+ [.9;.9];

for k=1:75
    J= [ -exp(-(t).^2/x(2)^2);
        -(2*x(1)*t.^2.*exp(-(t).^2/x(2)^2))/x(2)^3] .';

    f = x(1)*exp(-(t).^2/x(2)^2);
    r=y-f;

    x=x-(J.'*J)^(-1)*(J.'*r. ');
    xx=[xx,x];
    plot(f)
    pause(.1)
end
figure(2)
plot(xx(1,:),'.r');hold on;
plot(xx(2,:),'.g');
xlabel('Number of iterations');
ylabel('A and sigma values');
```



Note that σ is found to be -0.1 , whereas the function is defined with $\sigma = .1$. This is the case, since $\sigma = \pm 0.1$ both solves the datafitting problem, since σ is squared everywhere it is used.

6 Exercise 6

Conjugate gradient algorithm

Given the function f :

```
syms xx1 xx2;  
x = [xx1;xx2];  
Q = [2 -2;-2 6];  
b = [0;-3];  
f4(x) = 0.5*x.'*Q*x-x.'*b;
```

The initial starting point is:

```
x0 = [1;1];
```

Calculate the first 3 steps in the conjugate gradient method.

The gradient of f is:

```
gr(x) = Q*x-b;
```

Which evaluated in x_0 gives:

```
g0 = gr(x0(1),x0(2))
```

```
g0 =
```

```
0  
7
```

Since the gradient is not zero, we continue, by setting $d_0 = -g_0$ and finding α_0 :

```
d0 = -g0;  
a0 = -(g0.'*d0)/(d0.'*Q*d0)
```

```
a0 =
```

```
1/6
```

Next, we use the update function:

```
x1 = x0+a0*d0
```

```
x1 =
```

```
1  
-1/6
```

Reevaluating the gradient:

$$\mathbf{g}_1 = \mathbf{gr}(\mathbf{x}_1(1), \mathbf{x}_1(2))$$

$$\mathbf{g}_1 =$$

$$\begin{pmatrix} 7/3 \\ 0 \end{pmatrix}$$

Next we calculate β_0 :

$$\mathbf{b}_0 = \mathbf{g}_1.' * \mathbf{Q} * \mathbf{d}_0 / (\mathbf{d}_0.' * \mathbf{Q} * \mathbf{d}_0)$$

$$\mathbf{b}_0 =$$

$$1/9$$

Next we find the new direction:

$$\mathbf{d}_1 = -\mathbf{g}_1 + \mathbf{b}_0 * \mathbf{d}_0$$

$$\mathbf{d}_1 =$$

$$\begin{pmatrix} -7/3 \\ -7/9 \end{pmatrix}$$

Now we repeat the algorithm steps by finding α_1 and calculating \mathbf{x}_2 :

$$\mathbf{a}_1 = -(\mathbf{g}_1.' * \mathbf{d}_1) / (\mathbf{d}_1.' * \mathbf{Q} * \mathbf{d}_1)$$

$$\mathbf{x}_2 = \mathbf{x}_1 + \mathbf{a}_1 * \mathbf{d}_1$$

$$\mathbf{a}_1 =$$

$$3/4$$

$$\mathbf{x}_2 =$$

$$\begin{pmatrix} -3/4 \\ -3/4 \end{pmatrix}$$

Thus, again we have found the minimum.