CSI 747 – HW 8 – Resubmission Updates

Jayasurya Kanukurthy – G00863457 – jkanukur@gmu.edu

Only the Augmented Lagrangian methods have been updated so as to converge to the correct solution. The codes are implemented as given below:

Answer 2.a: The Augmented Lagrangian Implementation for Problem 2 (a) in MATLAB is as follows:

```
function [ output_args ] = AL1( x,y,k )

    f = @(x) 2*x(1)-3*x(2);
    g = @(x) x(1)^2+x(2)^2-25;

    output_args = f(x)-y'*g(x)+0.5*k*(norm(g(x))^2);

end


function [ output_args ] = AL1_Gradient( x,y,k )

    grad_f = [2;-3];
    g = @(x) x(1)^2+x(2)^2-25;
    Grad_g = @(x) [2*x(1);2*x(2)];

    output_args =   grad_f - y*Grad_g(x)+k*Grad_g(x)*g(x);

end

function [ output_args ] = AL1_Hessian( x,y,k )
        x1 = x(1);
        x2 = x(2);
        y1 = y(1);
        output_args =[ -2*y1 + k*(x1^2 + x2^2 - 25)*2 + k*4*x1^2,
k*2*(x2)*2*x1;
                        k*2*(x1)*2*x2, -2*y1 + k*(x1^2 + x2^2 - 25)*2 +
k*4*x2^2];
end

clear all; clc;
epsilon = 0.01;
x = [1;7];
y = 0;
k = 15;

g = @(xv) xv(1)^2+xv(2)^2-25;


while norm(g(x)) > epsilon
```

```
    newton_steps = 0;
    eta = 0.1;
    %Implementing Unconstrained Newton's Method
    while norm(AL1_Gradient(x,y,k)) >= max(epsilon,0.2*g(x))
        %Gradient of Augmented Lagrangian
        Gradient = AL1_Gradient(x,y,k);

        %Hessian of Augmented Lagrangian
        Hessian = AL1_Hessian(x,y,k);

        lambda = 0.00001;
        %Regularization: Checking for positive definiteness
        while min(eig( Hessian + lambda*eye(length(Hessian)))) <= 0
            lambda = 10*lambda;
        end
        %Hessian = Hessian + lambda*eye(length(Hessian));
        steps = (Hessian + lambda*eye(length(Hessian)))\-(Gradient);
        alpha = 1;
        %Armijo rule
        while (AL1(x+alpha*steps,y,k)-AL1(x,y,k)) >=
eta*alpha*AL1_Gradient(x,y,k)'*steps
            alpha = alpha/2;
        end

        x = x+alpha*steps;
        newton_steps = newton_steps+1;
    end

    fprintf('x1 = %f; x2 = %f; Newton Steps = %d; Constraint Violation =
%f\n', x(1), x(2), newton_steps, g(x));
    y = y - k *g(x);

end
```

<u>Output</u>:

x1 = -2.774811; x2 = 4.162265; Newton Steps = 19; Constraint Violation = 0.024028
x1 = -2.773491; x2 = 4.160261; Newton Steps = 1; Constraint Violation = 0.000026

Answer 2.b: The Implementation of Augmented Lagrangian Method for Problem 2.b is as follows:

```
function [ output_args ] = AL2( x,y,k )
    x1 = x(1);
    x2 = x(2);
    y1 = y(1);
    output_args = x1^2+2*x1*x2+x2^2-y1*(3*x1^2+x2^2-9)+k/2*norm(3*x1^2+x2^2-
9)^2;
```

```
end

function [ output_args ] = AL2_Gradient( x,y,k )
    x1 = x(1);
    x2 = x(2);
    y1 = y(1);
    output_args=    [2*x1 + 2*x2 - 6*y1*x1 + k*(3*x1^2 + x2^2 - 9)*6*x1;
                    2*x1 + 2*x2 - 2*y1*x2 + k*(3*x1^2 + x2^2 - 9)*2*x2];

end


function [ output_args ] = AL2_Hessian( x,y,k )
        x1 = x(1);
        x2 = x(2);
        y1 = y(1);
        output_args = [ 2 - 6*y1 + k*(3*x1^2 + x2^2 - 9)*6 + k*6*x1*6*x1,2 +
k*(2*x2)*6*x1;
                        2 + k*(6*x1)*2*x2,2 - 2*y1 + k*(3*x1^2 + x2^2 - 9)*2
+ k*(2*x2)*2*x2];
end


clear all; clc;
epsilon = 0.001;

y = 0;
x =[-2.5;2.5];

k = 1;
g = @(x) 3*x(1)^2+x(2)^2-9;

while norm(g(x)) > epsilon
    newton_steps = 0;
    eta = 0.1;
    %Implementing Unconstrained Newton's Method
    while norm(AL2_Gradient(x,y,k)) >= max(epsilon,0.2*g(x))
        %Gradient of Augmented Lagrangian
        Gradient = AL2_Gradient(x,y,k);

        %Hessian of Augmented Lagrangian
        Hessian = AL2_Hessian(x,y,k);

        lambda = 0.00001;
        %Regularization: Checking for positive definiteness
        while min(eig( Hessian + lambda*eye(length(Hessian)))) <= 0
            lambda = 10*lambda;
        end
        %Hessian = Hessian + lambda*eye(length(Hessian));
        steps = (Hessian + lambda*eye(length(Hessian)))\-(Gradient);
        alpha = 1;
        %Armijo rule
```

```
        while (AL2(x+alpha*steps,y,k)-AL1(x,y,k)) >= 
eta*alpha*AL2_Gradient(x,y,k)'*steps
            alpha = alpha/2;
        end

        x = x+alpha*steps;
        newton_steps = newton_steps+1;
    end

    fprintf('x1 = %f; x2 = %f; Newton Steps = %d; Constraint Violation = 
%f\n', x(1), x(2), newton_steps, g(x));
    y = y - k *g(x);
end
```

Output:

x1 = -1.500000; x2 = 1.500000; Newton Steps = 5; Constraint Violation = 0.000000

Answer 2.c:

The Implementation of Augmented Lagrangian Method for Problem 2.c is as follows:

```
function [ output_args ] = AL3( x,y,k )
    x1 = x(1); x2 = x(2); x3 = x(3); x4  = x(4);
    y1 = y(1);y2 = y(2);

    g = [x1^2+x2^2+x3^2+x4^2-4;x1+x2+2*x3+3*x4-1];
    output_args = 3*x1^3+2*x2^3+x3^3+x4^3-[y1 y2]*g+k/2*norm(g)^2;

end


function [ output_args ] = AL3_Gradient(  x,y,k )
    x1 = x(1); x2 = x(2); x3 = x(3); x4  = x(4);
    y1 = y(1);y2 = y(2);
    g1 = x1^2+x2^2+x3^2+x4^2-4;
    g2 = x1+x2+2*x3+3*x4-1;
    output_args=    [9*x1^2-2*x1*y1-y2+2*k*x1*g1+k*g2;
                     6*x2^2-2*x2*y1-y2+2*k*x2*g1+k*g2;
                     3*x3^2-2*x3*y1-2*y2+2*k*x3*g1+2*k*g2;
                     3*x4^2-2*x4*y1-3*y2+2*k*x4*g1+3*k*g2];
end


function [ output_args ] = AL3_Hessian(  x,y,k )
    x1 = x(1); x2 = x(2); x3 = x(3); x4  = x(4);
    y1 = y(1);y2 = y(2);
```

```
    output_args = [ 18*x1 - 2*y1 + k* (x1^2 + x2^2 + x3^2 + x4^2 - 4)*2 +
k*2*x1*2*x1 + k,k*2* (x2)*2*x1 + k,  k*2* (x3)*2*x1 + k*2,k*2* (x4)*2*x1 +
k*3;
k*2* (x1)*2*x2 + k, 12*x2 - 2*y1 + k* (x1^2 + x2^2 + x3^2 + x4^2 - 4)*2 +
k*2*x2*2*x2 + k, k*2* (x3)*2*x2 + k*2, k*2* (x4)*2*x2 + k*3;
k*2* (x1)*2*x3 + k*2,k*2* (x2)*2*x3 + k*2, 6*x3 - 2*y1 + k* (x1^2 + x2^2 +
x3^2 + x4^2 - 4)*2 + k*2*x3*2*x3 + k*2*2,k*2* (x4)*2*x3 + k*2*3;
k*2* (x1)*2*x4 + k*3,k*2* (x2)*2*x4 + k*3,k*2* (x3)*2*x4 + k*3*2,6*x4 - 2*y1
+ k* (x1^2 + x2^2 + x3^2 + x4^2 - 4)*2 + k*2*x4*2*x4 + k*3*3];

end
clear all; clc;
epsilon = 0.01;
y = [0;0];
x = [1;2;3;4];
k = 100;
g = @(x) [x(1)^2+x(2)^2+x(3)^2+x(4)^2-4; x(1)+x(2)+2*x(3)+3*x(4)-1];

while norm(g(x)) > epsilon
    newton_steps = 0;
    eta = 0.1;
    %Implementing Unconstrained Newton's Method
    while norm(AL3_Gradient(x,y,k)) >= max(epsilon,0.2*g(x))
        %Gradient of Augmented Lagrangian
        Gradient = AL3_Gradient(x,y,k);

        %Hessian of Augmented Lagrangian
        Hessian = AL3_Hessian(x,y,k);

        lambda = 0.00001;
        %Regularization: Checking for positive definiteness
        while min(eig( Hessian + lambda*eye(length(Hessian)))) <= 0
            lambda = 10*lambda;
        end
        %Hessian = Hessian + lambda*eye(length(Hessian));
        steps = (Hessian + lambda*eye(length(Hessian)))\-(Gradient);
        alpha = 1;
        %Armijo rule
        while (AL3(x+alpha*steps,y,k)-AL1(x,y,k)) >=
eta*alpha*AL3_Gradient(x,y,k)'*steps
            alpha = alpha/2;
        end

        x = x+alpha*steps;
        newton_steps = newton_steps+1;
    end

    fprintf('x1 = %f; x2 = %f; x3 = %f; x4 = %f; Newton Steps = %d;
Constraint1 = %f;Constraint2 = %f\n', x(1), x(2),x(3),x(4), newton_steps,
g(x));
    y = y - k *g(x);
end
```

Output:

x1 = -1.870848; x2 = 0.207074; x3 = 0.414149; x4 = 0.600589; Newton Steps = 14; Constraint1 = 0.075179;Constraint2 = -0.033708
x1 = -1.849036; x2 = 0.208028; x3 = 0.416056; x4 = 0.603051; Newton Steps = 2; Constraint1 = -0.001018;Constraint2 = 0.000256