AMERICAN UNIVERSITY OF BEIRUT

MAROUN SEMAAN FACULTY OF ENGINEERING & ARCHITECTURE

Department of Industrial Engineering

& Management

ENMG 616 – Advanced Optimization Techniques & Algorithms

Assignment 4 – Constrained Optimization & Duality

By:

Ralph Mouawad – ID 202204667

To:

Dr. Maher Nouiehed

A copy of the MATLAB codes will be provided for each question.

Part 1 - Solving Quadratic Optimization Problem –

Question 1-2:

```
n = 100;
B = randn(n,n);
Q = B*B';
Q = Q +(min(eig(Q))+10)*eye(n);
q = 10*randn(n,1);
L = max(eig(Q));
x0 = ones(n,1)
```

Question 3:

```
x1 = x0;
step1 = 1/L;
grad = Q*x1 + q;
for i = 1:200000
    x1 = x1 - step1*grad; % compute new iteration
    x1(x1 < 1) = 1; % projection to boundary when x < 1
    x1(x1 > 3) =3;
    grad = Q*x1 + q; % compute new gradient
end
x1;
```

After each computation of x1, I'm seeing whether it is in the region or not. If so, I project x1 back to the boundary defined here as x1 in the interval [1,3]. The same will be done when implementing the algorithm with diminishing step-size.

x1* has many terms very close to 1, maybe because of the initial matrices generated with normal elements. Also, it is hard to achieve a gradient very close to zero because of this normalization.

Question 4:

```
i=0;
x2 = x0;
grad = Q*x2 + q;
for j = 1:200000
    i = i+1;
    step2 = 5*log(i)/i*L;
    x2 = x2 - step2*grad;
    x2(x2 < 1) = 1;
    x2(x2 > 3) = 3;
    grad = Q*x2 + q;

end
x2;
```

The solution x2* has many of its elements being 1 & 3. The fact that we're projecting back to the set removes the problem of obtaining NaN solutions (mainly because of the step size).

Question 5: Frank-Wolfe method –

```
x3 = ones(n, 1);
step = 1/L;
errors_3 = zeros(2000,1);
for i = 1:2000
    grad = Q * x3 + q;
    d = zeros(n, 1);
    d(grad >= 0) = 1; % the element in 'd' where the
corresponding one in the gradient is positive should be 1
    d(grad < 0) = 3; % same but here when element of grad is
negative we take 3
    x3 = x3 + step * (d - x3);
end
```

Explanation of the code:

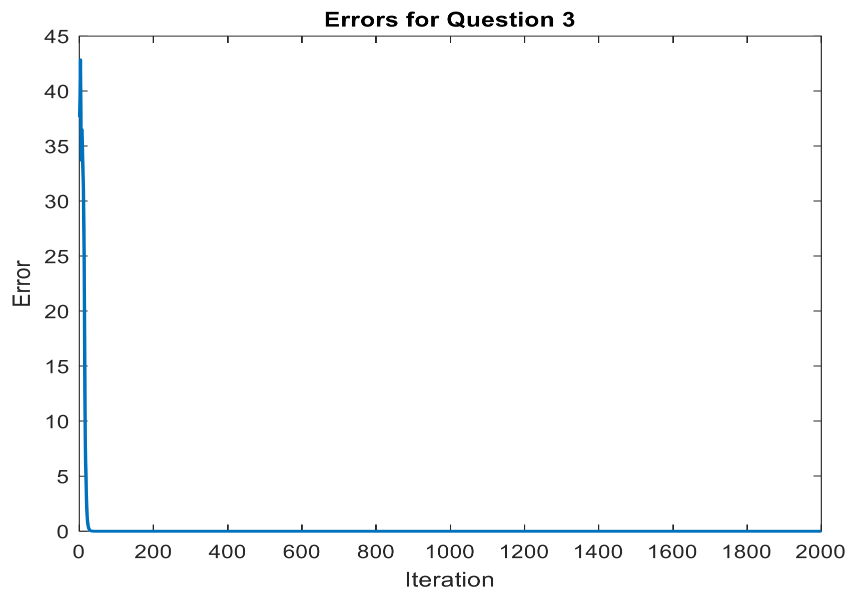
I defined a direction vector that takes the value 1 when the corresponding element in the gradient is positive and 3 when the corresponding element in the gradient is negative. I did this because we want to increase the elements that have a negative coefficient in the gradient and the opposite for those with positive values.

Question 6 –

I modified the previous codes to compute the error each time and these are the codes.

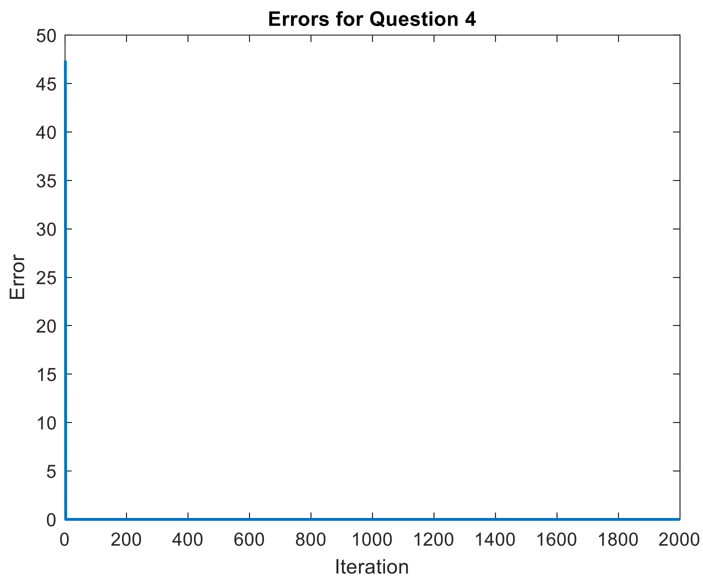
For constant step-size:

```
x1 = x0;
step1 = 1/L;
grad = Q*x1 + q;
errors_1 = zeros(2000, 1);
for i = 1:2000
    x1 = x1 - step1 * grad; % compute new iteration
    x1(x1 < 1) = 1; % projection to boundary when x < 1
    x1(x1 > 3) = 3; % projection to boundary when x > 3
    y1 = x1 - grad;
    y1(y1 > 3) = 3; % I am projecting the second term of
the error onto the feasible set before computing the
total error
    y1(y1 < 1) = 1;
    errors_1(i) = norm(x1 - y1)^2; %error at iteration i
    grad = Q * x1 + q; % compute new gradient
end
errors_1;
% Plot errors for Question 3
figure;
plot(1:2000, errors_1, 'LineWidth', 1.5);
title('Errors for Question 3');
xlabel('Iteration');
ylabel('Error');
```



For diminishing step-size:

```
x2 = x0;
grad = Q*x2 + q;
errors_2 = zeros(2000,1);
for j = 1:2000
    step2 = 5 * log(j) / j * L; % new step size
    x2 = x2 - step2 * grad; % new iteration
    x2(x2 < 1) = 1; % projection
    x2(x2 > 3) = 3; % projection
    y2 = x2 - grad; % second term of the error
    y2(y2 > 3) = 3; % projection of second term of error
    y2(y2 < 1) = 1; % projection of second term of error
    errors_2(j) = norm(x2 - y2)^2; %error at step i
    grad = Q * x2 + q; % compute new gradient
end
errors_2;
% Plot errors for Question 4
figure;
plot(1:2000, errors_2, 'LineWidth', 1.5);
title('Errors for Question 4');
xlabel('Iteration');
ylabel('Error');
```



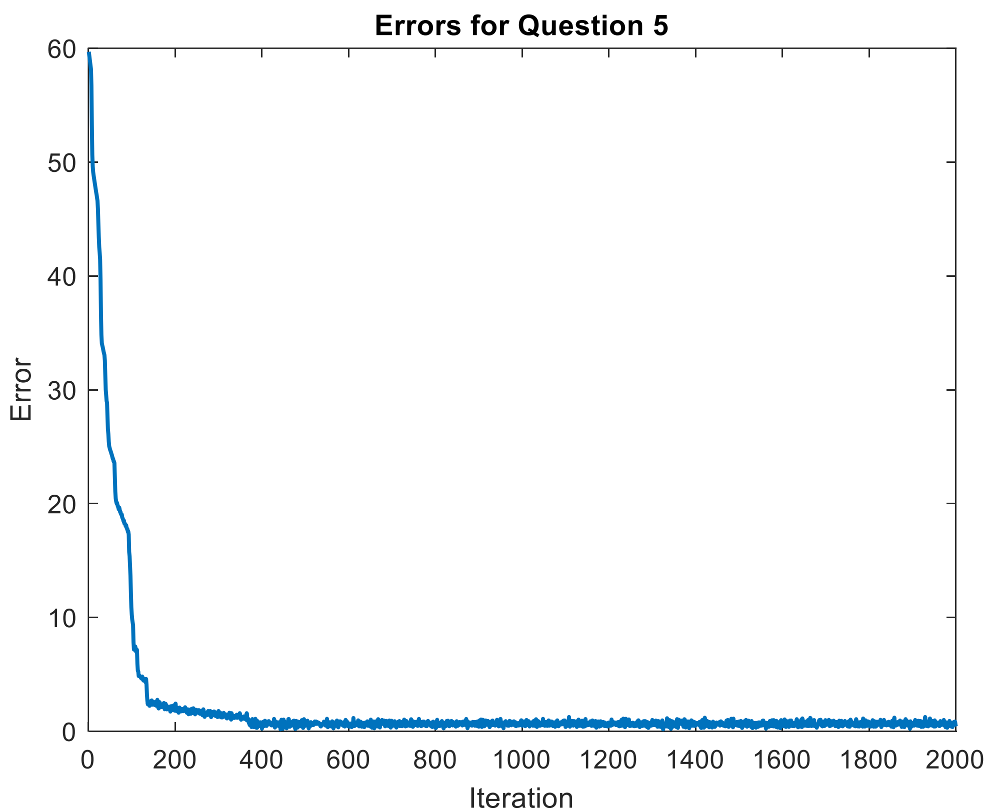
We only have an error at the first iteration.

For the Frank Wolfe Method:

```
x3 = ones(n, 1);
step = 1/L;
errors_3 = zeros(2000,1);
for i = 1:2000
    grad = Q * x3 + q;
    d = zeros(n, 1);
    d(grad >= 0) = 1; % the element in 'd' where the
corresponding one in the gradient is positive should be
1
    d(grad < 0) = 3; % same but here when element of
grad is negative we take 3
    x3 = x3 + step * (d - x3);
    y3 = x3 - grad;
    y3(y3 < 1) = 1;
    y3(y3 > 3) = 3;
    errors_3(i) = norm(x3 - y3)^2;
end
x3;
errors_3;

% Plot errors for Question 5
figure;
```

```
plot(1:2000, errors_3, 'LineWidth', 1.5);
title('Errors for Question 5');
xlabel('Iteration');
ylabel('Error');
```



Part 1.1 - Dual Problem –

Questions 1 & 2–

Primal: $\min_x f_0(x) = \frac{1}{2} x^T Q x + q^T x$

s.t: $x - 3 \leq 0 \dashrightarrow \lambda_1$ } vectors

$-x + 1 \leq 0 \dashrightarrow \lambda_2$

Lagrangian Function

$$L(x, \lambda_1, \lambda_2) = f_0(x) + \sum_{i=1}^{2} \lambda_i f_i(x)$$

$$= \frac{1}{2} x^T Q x + q^T x + \lambda_1^T (x-3) + \lambda_2^T (-x+1)$$

$$\frac{\partial L}{\partial x} = 0 \Rightarrow Qx^* + q + \lambda_1 - \lambda_2 = 0.$$

$$Qx^* = -\lambda_1 + \lambda_2 - q$$

$$x^* = (Q)^{-1} (-\lambda_1 + \lambda_2 - q)$$

$$x^* = -(Q)^{-1} (\lambda_1 - \lambda_2 + q).$$

$$g(\lambda) = \min_x L(x, \lambda_1, \lambda_2)$$

$\lambda = \begin{pmatrix} \lambda_{11} & \lambda_{21} \\ \lambda_{12} & \lambda_{22} \\ \lambda_{13} & \lambda_{23} \\ \vdots & \vdots \end{pmatrix}$

$g(\lambda) = g(\lambda_1, \lambda_2)$

$$= \frac{1}{2} \left[(Q)^{-1} (\lambda_1 - \lambda_2 + q) \right]^T Q \left[(Q^{-1})(\lambda_1 - \lambda_2 + q) \right] - q^T (Q)^{-1} (\lambda_1 - \lambda_2 + q)$$
$$+ \lambda_1^T \left(-(Q)^{-1} (\lambda_1 - \lambda_2 + q) - 3 \right)$$
$$+ \lambda_2^T \left((Q)^{-1} (\lambda_1 - \lambda_2 + q) + 1 \right)$$

$$g(\lambda) = \frac{1}{2} (\lambda_1 - \lambda_2 + q)^T \cancel{Q^{-1}} \cancel{Q} \left[(Q^{-1})(\lambda_1 - \lambda_2 + q) \right] - q^T (Q)^{-1} (\lambda_1 - \lambda_2 + q)$$
$$+ \lambda_1^T \left(-(Q)^{-1} (\lambda_1^T - \lambda_2^T + q) - 3 \right)$$
$$+ \lambda_2^T \left((Q)^{-1} (\lambda_1^T - \lambda_2^T + q) + 1 \right).$$

$$g(\lambda_1, \lambda_2) = \frac{1}{2} (\lambda_1 - \lambda_2 + q)^T (Q)^{-1} (\lambda_1 - \lambda_2 + q) - q^T (Q)^{-1} (\lambda_1 - \lambda_2 + q)$$
$$+ \lambda_1^T \left(-(Q)^{-1} (\lambda_1 - \lambda_2 + q) - 3 \right)$$
$$+ \lambda_2^T \left((Q)^{-1} (\lambda_1 - \lambda_2 + q) + 1 \right).$$

Dual Problem:

$$\max_{\lambda_1, \lambda_2} \frac{1}{2} (\lambda_1 - \lambda_2 + q)^T (Q)^{-1} (\lambda_1 - \lambda_2 + q) - q^T (Q)^{-1} (\lambda_1 - \lambda_2 + q)$$
$$+ \lambda_1^T \left(-(Q)^{-1} (\lambda_1 - \lambda_2 + q) - 3 \right)$$
$$+ \lambda_2^T \left((Q)^{-1} (\lambda_1 - \lambda_2 + q) + 1 \right)$$

s.t:

$$\lambda_1 \geq 0 \Rightarrow -\lambda_1 \leq 0$$
$$\lambda_2 \geq 0 \Rightarrow -\lambda_2 \leq 0$$

Question 3 –

The K.K.T conditions are:

Primal Feasibility:

- $f_i(x^*) <= 0 \rightarrow x^* <= 3$
  $x^* >= 1$

Dual Feasibility:

- $\lambda_i^* >= 0 \rightarrow \lambda_1^* >= 0$
  $\lambda_2^* >= 0$

Complementary Slackness:

- $\lambda_1^*(x^* - 3) = 0$
- $\lambda_2^*(-x^* + 1) = 0$

Stationarity Condition:

$$\nabla L(x, \lambda_1, \lambda_2) = Qx^* + q + \lambda_1^* - \lambda_2^* = 0$$

I substituted sometimes lambda1 by its transpose to correct the size for matrix multiplication.

Question 4 –

I will simplify my dual problem to implement on MATLAB. First we can write it as

Min $-g(\lambda_1, \lambda_2)$

s.t: $\lambda_1, \lambda_2 >= 0$

Now I will simplify a little bit my function to implement it on MATLAB.

$g(\lambda_1, \lambda_2) = \frac{1}{2}(\lambda_1 - \lambda_2 + q)^T Q^{-1}(\lambda_1 - \lambda_2 + q) - 3\lambda_1 + \lambda_2$

I will minimize -g(.,.)

I will implement the dual on MATLAB.

```
dual_function = @(lambda) -((1/2) * (lambda(1) - lambda(2) +
q)' * inv(Q) * (lambda(1) - lambda(2) + q) - 3 * lambda(1) +
lambda(2));

grad_dual = @(lambda) gradient(dual_function);
lambda = zeros(1,2);
L_dual = max(eig(inv(Q)));
for iteration = 1:2000
    % Compute the gradient of the objective function
    grad = grad_dual(lambda);

    % Perform the gradient descent step
    lambda = lambda - 1/L * grad;
    % Project onto the non-negativity constraints
    lambda(lambda < 0) = 0;
end
lambda;
dual_function(lambda)
```

I computed the primal = 4.1143e+03 as well as the dual.

We can say that strong duality holds.

Part 1.2 – Duality

Q = 0.5*(B + B')

I replaced at the beginning Q and these are the results.

Primal = 2.0972e+03.

We can see a huge decrease in the primal value, but the dual didn't run. I'm expecting it to change and break the strong duality. Also, in the Frank Wolfe plot, it took a little more time to reach 0 (error) and the graph wasn't fluctuating too much.

Part 2: Projection to a set of linear equality constraints:

$$\min_{x \in \mathbb{R}^n} \frac{1}{2} \|z - x\|_2^2$$
$$Ax = b.$$

1) Lagrangian Fct:

$$L(x, \lambda) = \frac{1}{2} \|z - x\|_2^2 + \lambda^T (Ax - b).$$

2) $g(\lambda) = \min_x L(x, \lambda) \quad s.t: \lambda \geq 0.$

$$\frac{\partial L}{\partial x} = 0$$

$$\Rightarrow x^* - z + A^T \lambda = 0$$

$$\Rightarrow \boxed{x^* = z - A^T \lambda}$$

$$g(\lambda) = \frac{1}{2} \| \cancel{z} - \cancel{z} + A^T \lambda \|_2^2 + \lambda^T (A(z - A^T \lambda) - b)$$

$$= \frac{1}{2} (A^T \lambda)^T (A^T \lambda) + \lambda^T A z - \lambda^T A A^T \lambda - \lambda^T b.$$

$$= \frac{1}{2} \lambda^T A A^T \lambda + \lambda^T A z - \lambda^T A A^T \lambda - \lambda^T b$$

$$= -\frac{1}{2} \lambda^T A A^T \lambda + \lambda^T A z - \lambda^T b.$$

Dual Prob:

$$\max \ -\frac{1}{2} \lambda^T A A^T \lambda + \lambda^T A z - \lambda^T b$$
$$s.t: \lambda \geq 0.$$

3) KKT conditions:

Primal Feasibility:

$$Ax^* - b = 0$$

Dual Feasibility:

$$\lambda \geqslant 0.$$

Complementary slackness:

$$\sum_i \lambda_i^* g_i(x) = 0 \Rightarrow \lambda^*(Ax^* - b) = 0.$$

Stationarity:

$$\frac{\partial L}{\partial x} = 0 \Rightarrow x^* - z + A^T\lambda^* = 0.$$

4) $x^* = z - A^T\lambda^*$. Let's put this in primal feasib:

$$\left.\begin{aligned} A(z - A^T\lambda^*) - b &= 0. \\ Az - AA^T\lambda^* &= b. \\ -AA^T\lambda^* &= b - Az. \\ AA^T\lambda^* &= Az - b. \\ A^T\lambda^* &= (A)^{-1}(Az - b). \end{aligned}\right\} \Rightarrow \begin{aligned} x^* &= z - A^T\lambda^* \\ x^* &= z - (A)^{-1}(Az - b). \end{aligned}$$

in terms of A, z and b.