HOMEWORK 3
MAE 206- OPTIMIZATION METHODS
INSTRUCTOR: PROF. SOLMAZ S. KIA

Problem 1. *Rate of convergence of the steepest descent algorithm for quadratic costs.*

Consider

$$f(x) = \frac{1}{2}x^\top Q x - \begin{bmatrix}14\\6\end{bmatrix}^\top x + 10, \quad x = \begin{bmatrix}x^1\\x^2\end{bmatrix} \in \mathbb{R}^2.$$

For the numerical values given below for Q, compute the condition number of Q. Use $\nabla f(x) = 0$ to compute $x^\star$ and $f(x^\star)$.

Write a Matlab code to solve this problem using the steepest descent algorithm with exact line search stepsize.

(a) Let $Q = \begin{bmatrix}5 & 2\\2 & 1\end{bmatrix}$. Complete the table below (report your answers with 6 significant digits). Plot also contours of your cost function around $x^\star$.

k	x_k^1	x_k^2	$\|d_k\|$	α_k	$f(x_k)$	$\frac{f(x_k)-f(x^\star)}{f(x_{k-1})-f(x^\star)}$
1	40.000000	-100.000000				$\times\times\times\times\times\times$
2						
3						
4						
5						
6						
7						
8						
9						
10						
20						
30						
40						
50						
60						
70						
80						
90						

Problem 1 continued

(b) Let $Q = \begin{bmatrix} 20 & 5 \\ 5 & 16 \end{bmatrix}$. Complete the table below (report your answers to with 6 significant digits). Plot also contours of your cost function around $x^\star$.

k	x_k^1	x_k^2	$\|d_k\|$	α_k	$f(x_k)$	$\frac{f(x_k)-f(x^\star)}{f(x_{k-1})-f(x^\star)}$
1	40.000000	-100.000000				$\times\times\times\times\times\times$
2						
3						
4						
5						
6						
7						
8						
9						
10						
11						
12						
13						
14						
15						

Choose an admissible fixed stepsize and solve the problem in case (b) using steepest descent algorithm with your stepsize and fill out the table below. Explain how you identify the admissible fixed stepsize.

k	x_k^1	x_k^2	$\|d_k\|$	α	$f(x_k)$	$\frac{f(x_k)-f(x^\star)}{f(x_{k-1})-f(x^\star)}$
1	40.000000	-100.000000				$\times\times\times\times\times\times$
2						
3						
4						
5						
6						
7						
8						
9						
10						
11						
12						
13						
14						
15						

Problem 2. *This problem is a practice on computing rate of convergence.*
Consider the iterative process

$$x_{k+1} = \frac{1}{2}(x_k + \frac{a}{x_k})$$

where $a > 0$. Assuming the process converges, to what does it converge? What is the order of convergence?

Problem 3. Use conjugate gradient method, to solve the two quadratic optimization problems given in Problem 1. How many steps does it take to find the solution?

Problem 4. *Scaling steepest descent method to obtain better convergence (for more details see pages 69-72 of Ref[1] and pages 243-246 of Ref[2].*
Recall that the rate of convergence of steepest descent algorithm for an unconstrained optimization problem with quadratic cost function $f(x) = \frac{1}{2}x^\top Qx - b^\top x$, $x \in \mathbb{R}^{n\times n}$ depends on the conditions number (ratio of largest to smallest eigenvalues of Q). For general nonlinear functions this rate depends on the eigenvalues of $\nabla^2 f(x^\star)$. We can improve the rate of convergence of the steepest descent by using change of variables to express the problem in a new coordinate system. Suppose $\mathbf{T} \in \mathbb{R}^{n\times n}$ is an invertible matrix. Let $x = T\,y$. Then, the problem of finding x to minimize $f(x)$ is equivalent to that of finding y to minimize $h(y) = f(Ty)$. Using y as underlying set of variables, we then have

$$\nabla h = T^\top \nabla f,$$

where ∇f is the gradient of f with respect to x. Thus using the steepest descent, the direction of search is

$$d_y = -T^\top \nabla f,$$

which in the original variables is

$$d_x = -TT^\top \nabla f.$$

Thus we see that the change of variable changes the direction of the search. The rate of convergence of steepest descent with respect to y will be determined by the eigenvalues of the Hessian of the objective, taken with respect to y. That Hessian is

$$\nabla^2 h(y) = H(y) = T^\top \nabla^2 f(Ty)T.$$

Thus, if $x^\star = Ty^\star$ is the solution point, the rate of convergence is governed by the matrix

$$H(y^\star) = T^\top \nabla^2 f(x^\star)T.$$

With the choice of T we can adjust the rate of convergence. Note however that if T is an orthonormal matrix, corresponding to y being defined from x by a simple rotation of coordinate, then $T^\top T = I$, and we see that the direction remains unchanged and the eigenvalues of $H(y^\star)$ are the same as eigenvalues of $\nabla^2 f(x^\star)$.

In case of the quadratic cost $f(x) = \frac{1}{2}x^\top Qx - b^\top x$, using change of variable $x = T\,y$, we can write the cost function as $h(y) = f(Ty) = \frac{1}{2}y^\top T^\top QTy - b^\top Ty$. The rate of convergence of optimization problem

$$y^\star = \underset{y\in\mathbb{R}^n}{\operatorname{argmin}} \frac{1}{2}\, y^\top \underbrace{T^\top QT}_{\bar{Q}}\, y - \underbrace{b^\top T}_{\bar{b}^\top}\, y \tag{1}$$

is defined by condition number of $\bar{Q} = T^\top QT$. With an appropriate choice for T we can improve the rate of convergence.

Consider a quadratic cost function with $Q = \begin{bmatrix} 2 & 0.1 \\ 0.1 & 0.2 \end{bmatrix}$ and $b = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$.

(a) Compute $x^\star$ from $\nabla f(x^\star) = 0$.

(c) Compute the rate of convergence of the steepest descent algorithm for this problem.

(d) Write a Matlab code to solve this problem using steepest descent method (see the sample code below). Starting from $x_0 = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$, how many steps it takes to stop the algorithm with stoping condition $\|\nabla f(x_k)\| \leq 10^{-6}$.

(e) Use the scaling matrix $T = \begin{bmatrix} a & 0 \\ 0 & b \end{bmatrix}$ to place the eigenvalues of $T^\top QT$ at $\{1, 1.4\}$. For the scaled optimization problem compute the rate of convergence of the steepest descent algorithm.

(f) Write a Matlab code to solve for $y^\star$ in the scaled optimization problem (1) using steepest descent method. Starting from $y_0 = T^{-1}x_0 = T^{-1}\begin{bmatrix} 1 \\ 2 \end{bmatrix}$, how many steps does it take to stop the algorithm with stoping condition $\|\nabla f(y_k)\| \leq 10^{-6}$. Compute $x^\star$ from $x^\star = T\,y^\star$.

Note: Online Nonlinear Equation Solver: `https://www.wolframalpha.com/examples/EquationSolving.html`

– – – – – – – – – – – – – – – Sample Code – – – – – – – – – – – – – – –

Find the minimizer of

$$f(x) = \frac{1}{2}x^\top \begin{bmatrix} 70 & 2 \\ 2 & 2 \end{bmatrix} x - \begin{bmatrix} 1 \\ 2 \end{bmatrix} x$$

A simple code for Steepest descent with exact line search:

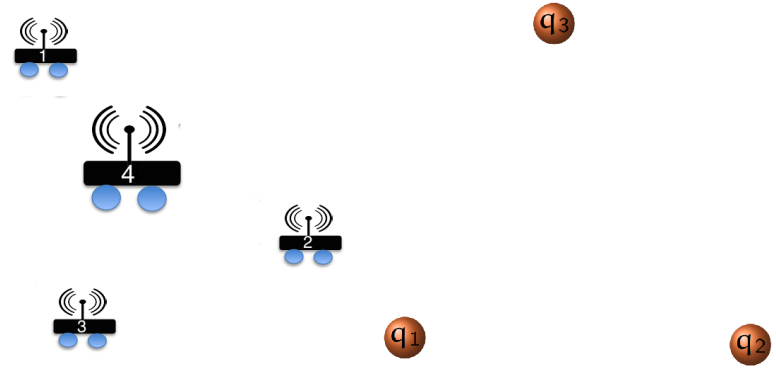
```
close all
Q=[70 2
    2 2];
b=[1 2]';

x_s=inv(Q)*b; %exact value of the minimizer

plot(x_s(1),x_s(2),'g*')
hold on
x1 = -1:0.01:1.2;
x2 = -1:0.01:4;
[X1,X2] = meshgrid(x1,x2);
v=[0.1,0.1,0.5,0.5,1,1,5,5,10,10,15,15,20,20,25,25,30,30,40,40,50,50,60,60];
Z = 0.5*(Q(1,1)*X1.*X1+2*Q(1,2)*X1.*X2+Q(2,2)*X2.*X2)-b(1)*X1-b(2)*X2;
contour(X1,X2,Z,v,'ShowText','on')
x_sol(:,1)=[1 4]';
plot(x_sol(1,1),x_sol(2,1),'r+')
k=1;
g(:,k)=Q*x_sol(:,k)-b;
while norm(g(:,k))>10^(-6)
    alpha(k)=(g(:,k)'*g(:,k))/(g(:,k)'*Q*g(:,k));
    x_sol(:,k+1)=x_sol(:,k)-alpha(k)*g(:,k);
    plot(x_sol(1,k+1),x_sol(2,k+1),'r+')
    k=k+1;
    g(:,k)=Q*x_sol(:,k)-b;
end
plot(x_sol(1,:),x_sol(2,:),'r')
N=k-1  %number of the iteration
```

– – – – – – – – – – – –Fun with quadratic programming (what follows is not part of your HW) – – – – – – – – –

Consider a mobile sensor deployment problem in which 4 robots are deployed to monitor 3 sites with given location $q_1 = (0, 0)$, $q_2 = (6, 0)$ and $q_3 = (3, 3)$. Position of robot $i \in \{1, 2, 3, 4\}$ is given by $p_i = (x_i, y_i)$.



Our objective is to find the location of the robots such that the following objective is satisfied:
objective Robot 1 is the closest robot in the team to location q_1, Robot 2 is the closest robot in the team to locations q_2, Robot 3 is the closest robot in the team to location q_3. Robot 4 is the backup robot which we like to keep it close to all three locations. To maintain the connectivity of the robots, we also want them to stay close to each other.

This deployment problem can be formulated as a quadratic optimization problem below

$$(p_1^\star, p_2^\star, p_3^\star, p_4^\star) = \text{argmin } \frac{1}{2}\Big(\|p_1 - q_1\|^2 + \|p_2 - q_2\|^2 + \|p_3 - q_3\|^2 + \|p_4 - q_1\|^2 + \|p_4 - q_2\|^2 + \|p_4 - q_3\|^2 + \\ \|p_1 - p_2\|^2 + \|p_1 - p_3\|^2 + \|p_1 - p_4\|^2 + \|p_2 - p_3\|^2 + \|p_2 - p_4\|^2 + \|p_3 - p_4\|^2\Big)$$

Since x and y directions are independent from each other, we can formulate the problem above as

$$(x_1^\star, x_2^\star, x_3^\star, x_4^\star) = \text{argmin } \frac{1}{2}\Big((x_1 - 0)^2 + (x_2 - 6)^2 + (x_3 - 3)^2 + (x_4 - 0)^2 + (x_4 - 6)^2 + (x_4 - 3)^2 + \\ (x_1 - x_2)^2 + (x_1 - x_3)^2 + (x_1 - x_4) + (x_2 - x_3)^2 + (x_2 - x_4)^2 + (x_3 - x_4)^2\Big)$$

$$(y_1^\star, y_2^\star, y_3^\star, y_4^\star) = \text{argmin } \frac{1}{2}\Big((y_1 - 0)^2 + (y_2 - 0)^2 + (y_3 - 3)^2 + (y_4 - 0)^2 + (y_4 - 0)^2 + (y_4 - 3)^2 + \\ (y_1 - y_2)^2 + (y_1 - y_3)^2 + (y_1 - y_4) + (y_2 - y_3)^2 + (y_2 - y_4)^2 + (y_3 - y_4)^2\Big)$$

Can you write these problems in the standard form $\frac{1}{2}\, x^\top Q x + b^\top x + c$ and solve them using one of the algorithms you learned so far?