Task (1):

for QR factorization, first of all we need to make a "Vandermonde matrix". Two polynomials are introduced in problem set. as following

$$y\_1 = x(\cos(r + 0.5x^3)) + \sin(0.5x^3) \qquad eq(1)$$

$$y\_2 = 4x^5 - 5x^4 - 20x^3 + 10x^2 + 40x + 10 + r \qquad eq(2)$$

$r = 0$

① in the matlab code, I used linspace command to provide 30 points between -2 and 2 on x axis.

② Then I used these 30 points to make "Vandermonde" matrix:

$$A = \begin{array}{c} \begin{array}{ccccc} \underline{x^0} & \underline{x^1} & \underline{x^2} & & \underline{x^5} \end{array} \\ \begin{bmatrix} 1 & x_1 & x_1^2 & \cdots & x_1^5 \\ 1 & x_2 & x_2^2 & & \vdots \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & x_{30} & x_{30}^2 & & x_{30}^5 \end{bmatrix}_{30\times5} \end{array} \qquad n=30\ ,\ m=5$$

③ I wrote a function for backsubstitution technique

④ I used $[Q, R1] = (A, 0)$ in matlab

which gives none-zero part of R i.e: $R = \begin{bmatrix} R1 \\ 0 \end{bmatrix}$

⑤ Solution description:

$$Ax = y \qquad \text{main equation}$$

$$(Q\,R1)\,x = y \qquad \text{Using } [Q, R1] = (A, 0)$$

$$(Q^T Q)\,R1\,x = Q^T y$$

$$R1\,x = b \qquad \text{Using backsubstitution}$$

$\therefore$ x will be estimated

Note: by increasing m, approximated plot and dataset will have better overlap.

Task(2):

in this task I wrote chelosky factorization function. General procedure is as bellow:

$$Ax = y \xrightarrow[\text{side}]{A^T \text{ both}} A^T A x = A^T y$$

Note: timing by A^T because we need square matrix for chelosky method

- by using chelosky function we have:

$$LDL^T x = A^T y \Rightarrow R \underbrace{R^T x}_{U} = \underbrace{A^T y}_{b}$$

- by using forward substitution function we can calculate U, the we have $R^T x = U$

- finally by using backward substitution method we can calculate x

Details of code is as following:

```
function [] = assignment_1(m)
clc ;
close all;
n     =  30;
start = -2;
stop  =  2;
x     =  linspace(-2, 2, 30);

eps = 1;
rng(1);
r = rand(1,n) * eps;
y_1 = x.*(cos(r+0.5*x.^3)+sin(0.5*x.^3));                      % data set(1): equation (1)
y_2 = 4*x.^5 - 5*x.^4 - 20*x.^3 + 10*x.^2 + 40*x + 10 + r;     % data set(2): equation (2)

% ******** making a column vector from y_2 ****************

y1 = y_1 .';
y2 = y_2 .';

%************************************************************
%      making matrix A which is Vandermonde matrix
%************************************************************

A = ones(n , m);
for j = 2 : m
    for i = 1 : n

        A(i , j) = x (1, i)^(j-1);

    end
end


[Q , R1] = qr(A, 0);  % QR decomposition. Please note that R1 is non-zero part of R such that R =
[R1 , 0]

B = A * A';
[L , D] = cholesky(A' * A);
bb = A' * y2;
R2  = L * D^(1/2);

 c1 = backsubst(R1 , (Q') * y1);
   c2 = backsubst(R1 , (Q') * y2);

    cc1 = backsubst(R2' , fwrdsubst (R2 , A' * y1));
    cc2 = backsubst(R2' , fwrdsubst (R2 , A' * y2));


%**************************************************************
%                        Plotting Task (1)
%**************************************************************
%  equation (1)
   figure
   plot (x, A * c1);
   hold on
   plot (x, y_1 ,'o');
   title ('Task1: Approx of Polynomial #(1) via [QR] method')
   legend ('y1: approx via [QR]', 'y1 :data set')
% equation (2)
   figure
   plot (x, A * c2)
   hold on
   plot (x, y_2,'o');
   title ('Task1: Approx of Polynomial #(2) via [QR] method')
   legend ('y2: approx via [QR]', 'y2 :data set')
%**************************************************************
%                        Plotting Task (2)
%**************************************************************
%  equation (1)
    figure
    plot (x, A * cc1)
    hold on
    plot (x, y_1,'o');
    title ('Task2: Approx of Polynomial #(1) via [QR] method')
    legend ('y2: approx via Chol', 'y2 :data set')
%  equation (2)
    figure
    plot (x, A * cc2)
    hold on
```

```
    plot (x, y_2,'o');
    title ('Task2: Approx of Polynomial #(2) via [QR] method')
    legend ('y2: approx via Chol', 'y2 :data set')
end

%**************************************************************
%                    Cholesky Method:

% B X = A(T) y ----> L * D^(0.5) * D^(0.5) * L(T) * X = A(T) y
%            R1 * R1(T) X = A(T) y ----> R1 * U = A(T) y
%                      R1(T) X = U
%**************************************************************
function [L , D] = cholesky(B)    %R = L * D^(0.5)

    [n , m] = size (B);
    L = zeros (m);
    D = zeros (n);

    Bk = B;

    if (n == m)
        for k = 1 : n
            lk = Bk(: , k) / Bk(k , k);
            D(k , k) = Bk(k , k);
            Bk = Bk - D(k , k) * lk * lk';
            L(: , k) = lk;
        end
    else
        fprint ('Square Matrix is Required for Input of Cholesky Method')
    end

end

%**************************************************************
%                       Back Substitution
%**************************************************************
function x = backsubst(U , b)

% finds x where Ux = b for an upper triangular matrix U

[n , m]  = size (U);


% erz = 1 e-12        % required for checking upper-triangular matrix


 x  = zeros(n , 1);
if (n == m)

    x (n) = b(n) / U(n , m);
end

for k = (n-1) : -1 : 1
    x(k) = (b(k) - U(k , k+1 : n) * x(k+1 : n)) / U(k , k);
end
end

%**************************************************************
%                       Forward Substitution
%**************************************************************
function x = fwrdsubst (U, b)

[n , m]  = size (U);

if (n == m)

    x = zeros (n , 1);

    x(1) = b(1)/ U(1 , 1);

    for k = 2 : n

    %for j = 1 : (k - 1)

    x(k) = (b(k) - U(k , 1 : k) * x(1 : k)) / U(k , k);

    end
end
end
```

Task (3)

Actually condition is independent from type of algorithm which is applied for solving a problem. In the other word, condition tells us how much is perturbation results due to small changes in x. Therefore, condition is a property of problem an it is independent from solution. Apparently for matrix operation, depend on the size of matrix we have a series of linear equation presenting as $Ax=b$

Consequently, condition is characteristic of matrix and it does not depend on applied solution for for solving system of equations.

comparing QR and Cholesky, we cannot judge them via condition factor. Since we know condition factor for an invertible matrix like A is defined as:

$$K = \|A^{-1}\| \cdot \|A\|$$

Consequently condition factor is the same for both