Assignment 3

Numerical Methods, 2024 Spring

111550057 資工15 莊婷馨

1.

(a) Use a function divided_diff() to calculate the value of divided difference table.

```
% calculate the divided difference table
function table = divided_diff(x,y)
    n = length(x);
    table = zeros(n,n);
    table(:,1) = y';    % first column is f(x)
    for j = 2:n
        for i = j:n
            % calculate the value based on the previous column
            table(i,j) = (table(i,j-1) - table(i-1,j-1)) / (x(i) - x(i-j+1));
        end
    end
end
```

The result:

```
divided difference table:
    1.2300         0         0         0         0
    2.3400    2.2200         0         0         0
   -1.0500   -8.4750  -11.8833         0         0
    6.5100   -7.5600   -1.5250 -103.5833         0
   -0.0600  -16.4250   14.7750  -81.5000   73.6111
```

The first column is f[x], second column is f[x(i-1), x(i)], and so on. The last column has value f[x(1),...x(5)].

(b) To interpolate with the first 3 points, according to the divided difference table, the equation will be

$p(x) = 1.23 + 2.22 \times (x + 0.2) + (-11.8833) \times (x + 0.2) \times (x - 0.3)$

The result:

```
p(0.4) = 1.849
```

(c) The "best set" of 3 points should be the three points that has closest x value to 0.4. Get the points by sorting them according to distance to 0.4.

```
abs_diff = abs(x - 0.4);  % calculate the distance between x and 0.4
[~, sorted_indices] = sort(abs_diff);    % sort based on the distance
x = x(sorted_indices);
fx = fx(sorted_indices);
```

The three points are (0.3, 2.34), (0.7, -1.05) and (0.1, -0.06). Then use the function in part (a) to get the divided difference table

```
divided difference table:
    2.3400         0         0         0         0
   -1.0500   -8.4750         0         0         0
   -0.0600   -1.6500  -34.1250         0         0
    1.2300   -4.3000    2.9444  -74.1389         0
    6.5100  -52.8000  121.2500 -118.3056   73.6111
```

The equation interpolating first 3 points is

$p(x) = 2.34 + (-8.475) \times (x - 0.3) + (-34.125) \times (x - 0.3) \times (x - 0.7)$

The result:

```
p(0.4) = 2.5162
```

2. Choose 5 evenly spaced points

```
x = [-1, -0.5, 0, 0.5, 1];
y = [0, 0, 1, 0, 0];
```

Use the method mentioned in class, $HS = Y$, to calculate the coefficients of

$$g_i(x) = a_i(x - x_i)^3 + b_i(x - x_i)^2 + c_i(x - x_i) + d_i$$

Implementation:

```
x = [-1, -0.5, 0, 0.5, 1];
y = [0, 0, 1, 0, 0];

h = [0.5, 0.5, 0.5, 0.5];

H = [h(1), 2*(h(1)+h(2)), h(2),0, 0 ;
     0, h(2), 2*(h(2)+h(3)), h(3),0 ;
     0, 0, h(3), 2*(h(3)+h(4)), h(4)];

Y = [( (y(3)-y(2)) / h(2) ) - ( (y(2)-y(1)) / h(1) ) ;
     ( (y(4)-y(3)) / h(3) ) - ( (y(3)-y(2)) / h(2) ) ;
     ( (y(5)-y(4)) / h(4) ) - ( (y(4)-y(3)) / h(3) ) ];
Y = 6 .* Y;
```

- End Condition 3

```
syms k;
H = [ (3*h(1)+2*h(2)), h(2),0 ;
      h(2), 2*(h(2)+h(3)), h(3);
      0, h(3), (2*h(3)+3*h(4))];
s = H\Y;
S = [ s(1); s(1); s(2); s(3); s(3)];
a = zeros(1,4);
b = zeros(1,4);
c = zeros(1,4);
for i=1:4
    a(i) = (S(i+1)-S(i))/(6*h(i));
    b(i) = S(i)/2;
    c(i) = ((y(i+1)-y(i))/h(i)) - (2*h(i)*S(i) + h(i)*S(i+1)) / 6;
end
g1 = a(1)*(k-x(1))^3 + b(1)*(k-x(1))^2 + c(1)*(k-x(1)) + y(1);
g2 = a(2)*(k-x(2))^3 + b(2)*(k-x(2))^2 + c(2)*(k-x(2)) + y(2);
g3 = a(3)*(k-x(3))^3 + b(3)*(k-x(3))^2 + c(3)*(k-x(3)) + y(3);
g4 = a(4)*(k-x(4))^3 + b(4)*(k-x(4))^2 + c(4)*(k-x(4)) + y(4);
```

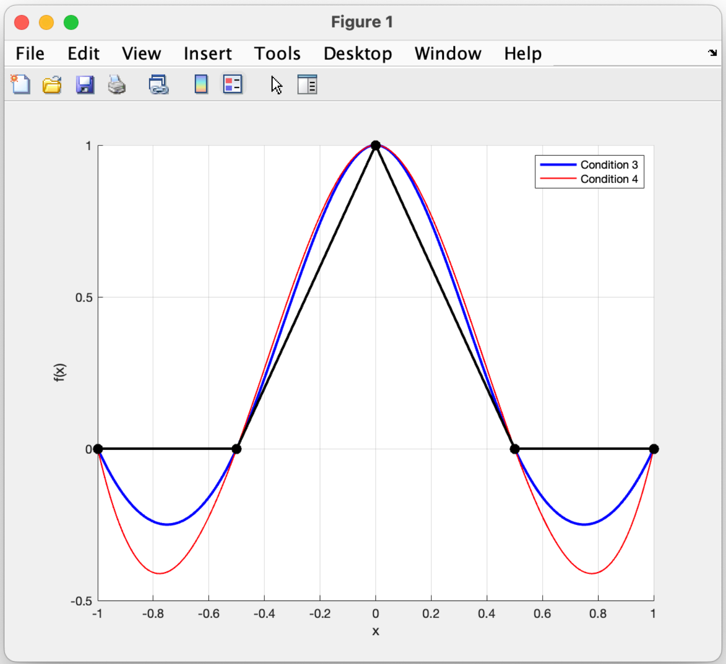
Set S(1) = S(2) and S(5) = S(4) according to the condition.
Then calculate the coefficients and form splines.

- End Condition 4

```
H = [ (h(1)+h(2))*(h(1)+2*h(2))/h(2) , (h(2)^2-h(1)^2)/h(2), 0 ;
      h(2), 2*(h(2)+h(3)), h(3);
      0,  (h(3)^2-h(4)^2)/h(3), (h(4)+h(3))*(h(4)+2*h(3))/h(3)];
s = H\Y;
S = [ ((h(1)+h(2))*s(1) - h(1)*s(2)) / h(2) ;
      s(1);
      s(2);
      s(3);
      ((h(3)+h(4))*s(3) - h(4)*s(2)) / h(3) ];
```

The calculation process of coefficients is the same as condition 3.

Then we are able to plot the entire figure, including two splines:



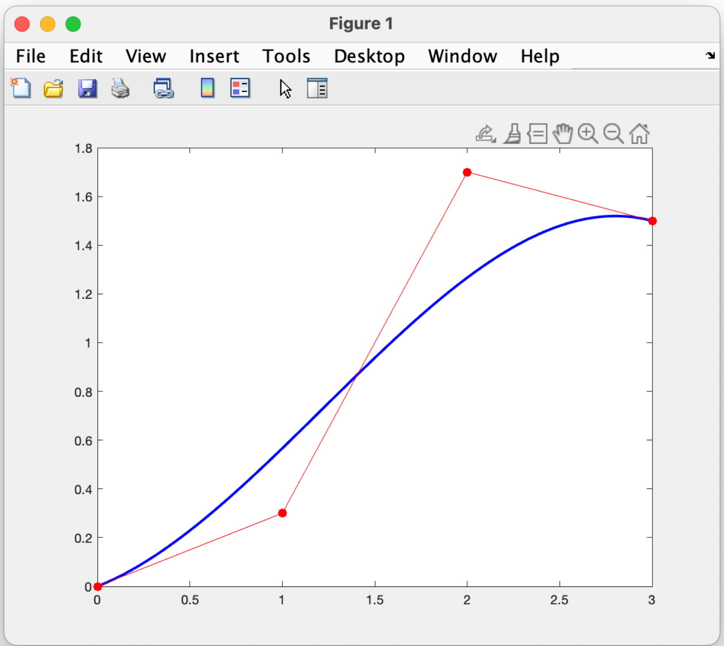
According to the graph, it is obvious that end condition 3 gives a better fit to the function.

3.

(a) Find the cubic Bezier curve by formula mentioned in class

```
u = linspace(0,1,100);
bezier_curve = kron((1-u).^3,pt1) + kron((3*u.*(1-u).^2),pt2) + kron((3*(u.^2).*(1-u)), pt3) + kron((u.^3),pt4);
```

Then plot the diagram:



(b) In order to let the curve pass through (1, 0.3) and (2, 1.7) , let the new control points locate at (1, 0.3+a) and (2, 1.7+b) .

Assuming the original points pass through the new curve, set two equations to calculate a and b.

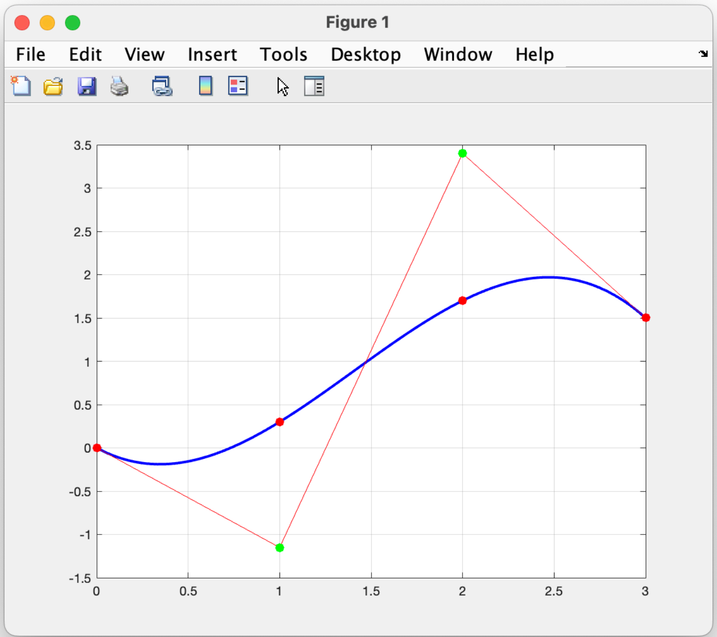
```
% u=1/3 -> x=1, y=0.3
% u=2/3 -> x=2, y=1.7
% Define equations for points pt2 and pt3
syms a b;
eqns = [(1 - (1/3).^3) * pt1(2) + 3 * (1/3) * (1 - 1/3).^2 * (0.3 + a) + 3 * ((1/3)^2) * (1 - 1/3) * (1.7 + b) + ((1/3)^3) * pt4(2) - 0.3 == 0,
        (1 - (2/3).^3) * pt1(2) + 3 * (2/3) * (1 - 2/3).^2 * (0.3 + a) + 3 * ((2/3)^2) * (1 - 2/3) * (1.7 + b) + ((2/3)^3) * pt4(2) - 1.7 == 0 ];
```

Solve the equations and get the new control points:

```
new point 2:
    1.0000   -1.1500

new point 3:
    2.0000    3.4000
```

The plot of the new curve:



4. The points nearest to (2.8, 0.54) are

$x\backslash y$	0.2	0.4	0.5	0.7
1.3	2.521	2.792	2.949	3.314
2.5	3.721	3.992	4.149	4.514
3.1	4.321	4.592	4.749	5.114
4.7	5.921	6.192	6.349	6.714

Plot the surface by the formula:

```
X = U * M * x * M' * V';
Y = U * M * y * M' * V';
Z = U * M * z * M' * V';
```

Where the variables are

```
y = [0.2, 0.4, 0.5, 0.7; 0.2, 0.4, 0.5, 0.7; 0.2, 0.4, 0.5, 0.7; 0.2, 0.4, 0.5, 0.7];
x = [1.3, 1.3, 1.3, 1.3; 2.5, 2.5, 2.5, 2.5; 3.1, 3.1, 3.1, 3.1; 4.7, 4.7, 4.7, 4.7];
z = [2.521, 2.792, 2.949, 3.314; 3.721, 3.992, 4.149, 4.514; 4.321, 4.592, 4.749, 5.114; 5.921, 6.192, 6.349, 6.714];

M = [-1, 3, -3, 1;
     3, -6, 3, 0;
     -3, 0, 3, 0;
     1, 4, 1, 0];
M = (1/6) .* M;

syms u v;
U = [u.^3; u.^2; u; ones(size(u))];
V = [v.^3; v.^2; v; ones(size(v))];
U = U';
V = V';
```

Calculate the value of u and v at (2.8, 0.54)

```
x_i = 2.8;
y_i = 0.54;

u_val = (x_i-1.3)/3.4;
v_val = (y_i-0.2)/0.5;
```

The result:

```
z(2.8, 0.54) = 4.3706
```

5.

(a) Calculate the normal equations by first recursively adding the square difference of each dimension. Then calculate the partial derivatives of a, b and c.

```
syms a b c;
R = 0;
for i=1:7
    R = R + (z(i) - (x(i)*a + y(i)*b + c))^2;
end
Ra = diff(R, a) == 0;
Rb = diff(R, b) == 0;
Rc = diff(R, c) == 0;
```

The result:

```
(20079*a)/50 + (21349*b)/50 + (313*c)/5 - 443587/1250 == 0

(21349*a)/50 + (11471*b)/25 + 70*c - 1873327/5000 == 0

(313*a)/5 + 70*b + 14*c - 26919/500 == 0
```

(b) Solve the normal equations to get the a, b and c.

```
eq = [Ra, Rb, Rc];
sol = solve(eq,[a, b, c]);
```

The result:

```
z = a*x + b*y + c
    a: 700991077/439192100
    b: -308480343/439192100
    c: 242286941/1097980250
```

(c) Fit a, b, and c into the first equation to get the sum of the squares of the deviations of the points.

```
a_value = sol.a;
b_value = sol.b;
c_value = sol.c;
R_value = subs(R, [a, b, c], [a_value, b_value, c_value]);
```

The result:

```
701379088827/2195960500000
```

6.

① Maclaurin Series of $\cos(x)$:

$$\sum_{n=0}^{\infty} \frac{(-1)^n \cdot (x)^{2n}}{(2n)!} = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} \cdots$$

② Chebyshev Series $\quad T_{n+1}(x) = 2x\,T_n(x) - T_{n-1}(x)$

$T_0(x) = 1$

$T_1(x) = x$

$T_2(x) = 2x^2 - 1$

$T_3(x) = 4x^3 - 3x$

$T_4(x) = 8x^4 - 8x^2 + 1 \quad \ldots\ldots$

$$\cos(x) \approx T_0(x) - \frac{1}{2!}\left(\frac{1}{2}T_2(x) - \frac{1}{2}T_0(x)\right)$$

$$+ \frac{1}{4!}\left(\frac{1}{8}T_4(x) + \frac{1}{2}T_2(x) + \frac{3}{8}T_0(x)\right)$$

$$- \frac{1}{6!}\left(\frac{1}{32}T_6(x) + \frac{3}{16}T_4(x) + \frac{15}{32}T_2(x) + \frac{5}{16}T_0(x)\right)$$

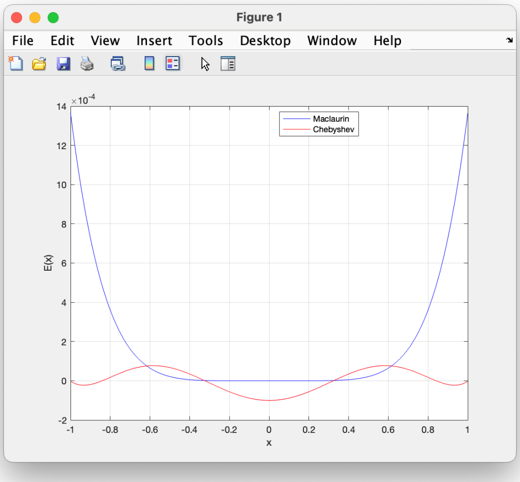
$$\Rightarrow \cos(x) = 0.7652\,T_0(x) - 0.2298\,T_2(x) + 0.0049\,T_4(x)$$

$$- 0.0000434\,T_6(x)$$

$$\approx 0.9999 - 0.4988x^2 + 0.0392x^4$$

Comparison:

The Malclaurin series has smaller error near 0. However, the further x is from 0, the error of Malclaurin series increase drastically.
The error of Chebyshev series are bigger near 0 but acceptable, and does not increase much when near x-1 or x=-1.



7. The Fourier coefficients are calculated as follows

$$f(x) = \frac{A_0}{2} + \sum_{n=1}^{\infty} \left(A_n \cos\left(\frac{n\pi x}{L}\right) + B_n \sin\left(\frac{n\pi x}{L}\right) \right)$$

$$2L = 3, \quad L = \frac{3}{2}$$

$$A_0 = \frac{1}{L} \int_{-1}^{2} f(x)\, dx$$

$$= \frac{2}{3} \int_{-1}^{2} (x^2 - 1)\, dx$$

$$= \frac{2}{3} \left(\frac{1}{3}x^3 - x \right) \Bigg|_{x=-1}^{x=2}$$

$$= \frac{2}{3} \left(\frac{8}{3} - 2 + \frac{1}{3} - 1 \right) = 0$$

$$A_n = \frac{2}{3} \int_{-1}^{2} (x^2 - 1) \cos\left(\frac{2n\pi x}{3}\right) dx$$

$$B_n = \frac{2}{3} \int_{-1}^{2} (x^2 - 1) \cdot \sin\left(\frac{2n\pi x}{3}\right) dx$$

Use matlab to calculate An and Bn.

```
n_terms = 5; % Number of Fourier coefficients to compute

an = zeros(1, n_terms);
bn = zeros(1, n_terms);

% Fourier coefficients
for n = 1:n_terms
    an(n) = (2 / T) * integral(@(x) f(x) .* cos(2*pi*n*x / T), -1, 2);
    bn(n) = (2 / T) * integral(@(x) f(x) .* sin(2*pi*n*x / T), -1, 2);
end
```

The results (can be adjusted to show more coefficients):

```
a1 = -1.2829, b1 = -0.3123
a2 = 0.2995, b2 = 0.4362
a3 = 0.1013, b3 = -0.3183
a4 = -0.2352, b4 = 0.0700
a5 = 0.1472, b5 = 0.1271
```