Chapter 3

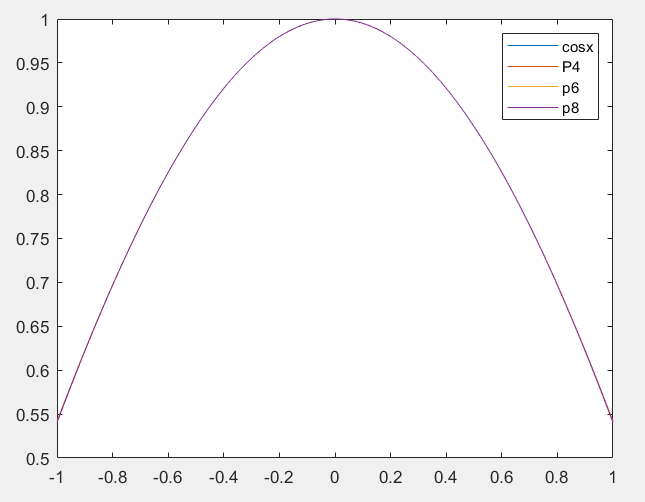
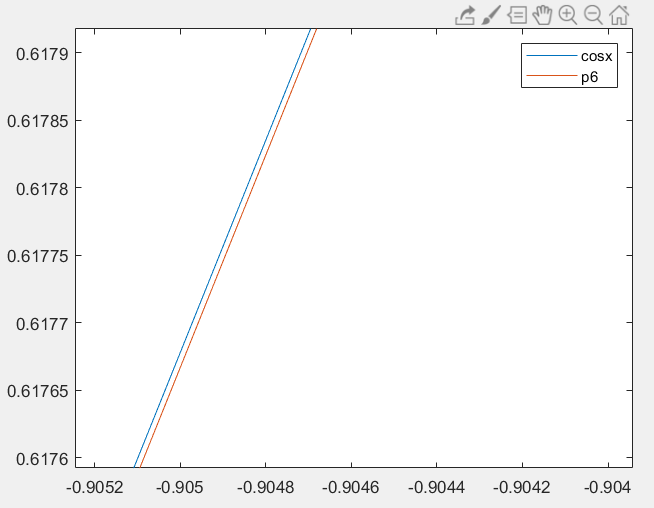
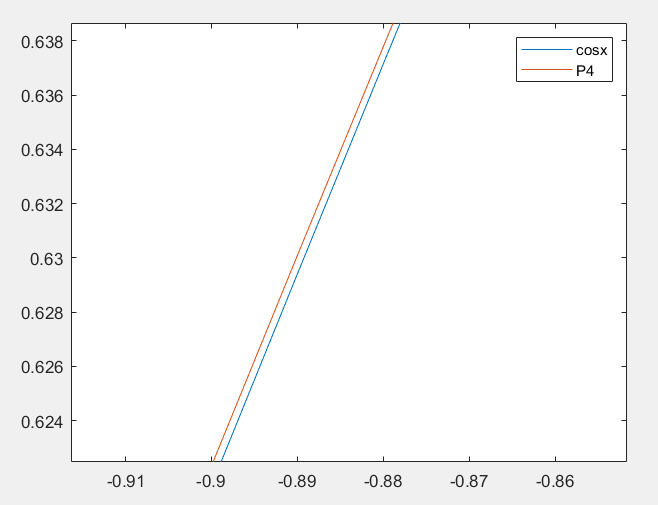
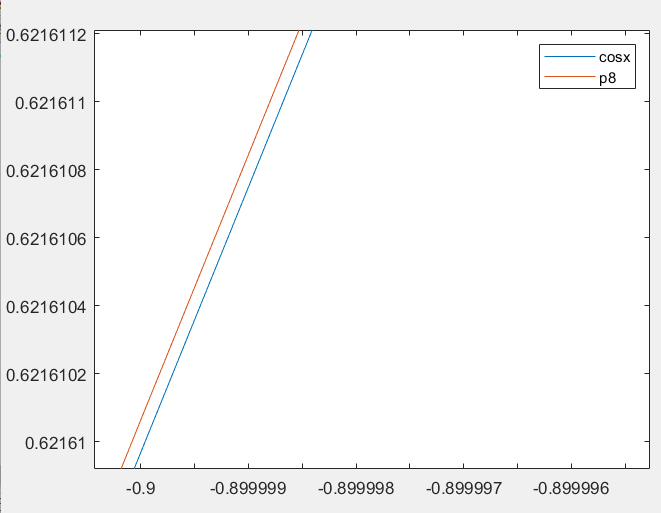
##### P102 2

(a) Use the plot command to plot , from Exercise 2 on the same graph using the interval   
(b) Create a table with columns that consist of evaluated at 19 equally spaced values of from the interval .

Taylor Polynomial Approximation:  
，  
let ，  
,   
,   
.

x=-1:0.01:1;  
plot(x,cos(x))  
hold on  
plot(x,polyval([1/prod(1:4),0,-1/2,0,1],x))  
hold on  
plot(x,polyval([-1/prod(1:6),0,1/prod(1:4),0,-1/2,0,1],x))  
hold on  
plot(x,polyval([1/prod(1:8),0,-1/prod(1:6),0,1/prod(1:4),0,-1/2,0,1],x))  
legend('cosx','P4','p6','p8')

Draw the graph, and compare the fitting degree of different order approximation curve near the end point, it can be seen that the polynomial of order 8 has a better fitting degree at the end point, so it has a better fitting degree for the function curve.

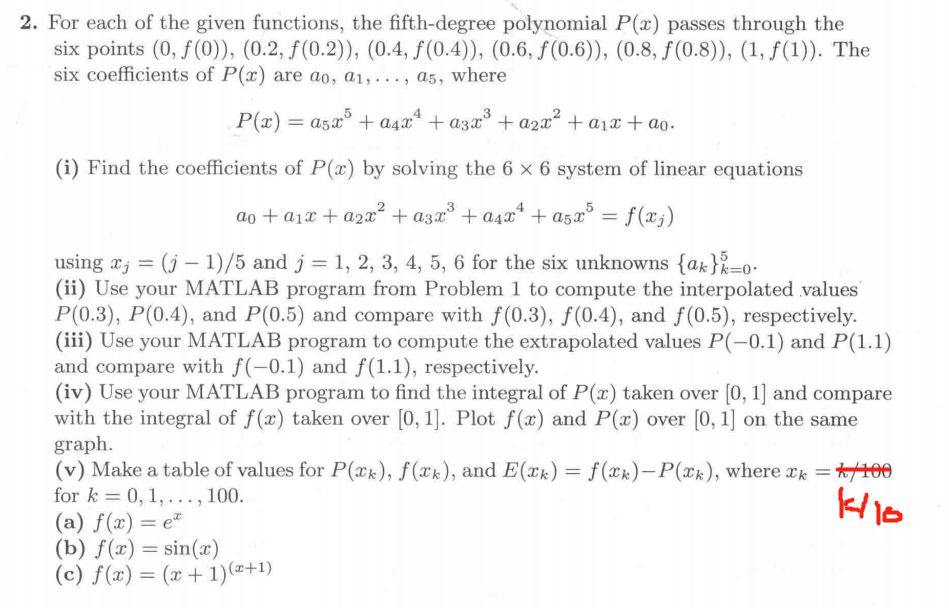
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
| -0.9 | 0.622337500000000 | 0.621599387500000 | 0.621610063770089 | 0.621609968270664 |
| -0.8 | 0.697066666666667 | 0.696702577777778 | 0.696706738793651 | 0.696706709347165 |
| -0.7 | 0.765004166666667 | 0.764840765277778 | 0.764842195039931 | 0.764842187284488 |
| -0.6 | 0.825400000000000 | 0.825335200000000 | 0.825335616571429 | 0.825335614909678 |
| -0.5 | 0.877604166666667 | 0.877582465277778 | 0.877582562158978 | 0.877582561890373 |
| -0.4 | 0.921066666666667 | 0.921060977777778 | 0.921060994031746 | 0.921060994002885 |
| -0.3 | 0.955337500000000 | 0.955336487500000 | 0.955336489127232 | 0.955336489125606 |
| -0.2 | 0.980066666666667 | 0.980066577777778 | 0.980066577841270 | 0.980066577841242 |
| -0.1 | 0.995004166666667 | 0.995004165277778 | 0.995004165278026 | 0.995004165278026 |
| 0 | 1.000000000000000 | 1.000000000000000 | 1.000000000000000 | 1.000000000000000 |
| 0.1 | 0.995004166666667 | 0.995004165277778 | 0.995004165278026 | 0.995004165278026 |
| 0.2 | 0.980066666666667 | 0.980066577777778 | 0.980066577841270 | 0.980066577841242 |
| 0.3 | 0.955337500000000 | 0.955336487500000 | 0.955336489127232 | 0.955336489125606 |
| 0.4 | 0.921066666666667 | 0.921060977777778 | 0.921060994031746 | 0.921060994002885 |
| 0.5 | 0.877604166666667 | 0.877582465277778 | 0.877582562158978 | 0.877582561890373 |
| 0.6 | 0.825400000000000 | 0.825335200000000 | 0.825335616571429 | 0.825335614909678 |
| 0.7 | 0.765004166666667 | 0.764840765277778 | 0.764842195039931 | 0.764842187284488 |
| 0.8 | 0.697066666666667 | 0.696702577777778 | 0.696706738793651 | 0.696706709347165 |
| 0.9 | 0.622337500000000 | 0.621599387500000 | 0.621610063770089 | 0.621609968270664 |

P109 1, 2  
1.Write a program in MATLAB that will implement Algorithm 3.1. The program should accept the coefficients of the polynomial as an matrix: .  
，  
，  
，let ，thus .

%Horner1.m  
function [v] = Horner1(A,X)  
%Input - A: Coefficients of P(x)  
% - X: Independent variable  
%Output - the values of P(x)  
A=flip(A);  
N=length(A)-1;  
B=zeros(N+1,1);  
B(N+1)=A(N+1);  
for k=N:-1:1  
 B(k)=A(k)+B(k+1)\*X;  
end  
v=B(1);  
end  
  
%test  
>> A=[3,2,1];  
>> X=1;  
>> [v] = Horner1(A,X)  
v =  
 6

%Horner2.m  
function [v] = Horner2(A,X)  
%input - N: Degree of P(x)  
% - A: Coefficients of P(x)  
% - C: Constant of integration  
% - X: Independent variable  
%Output - the values of P'(x)  
A=flip(A);  
N=length(A)-1;  
D=zeros(N,1);  
D(N)=N\*A(N+1);  
for k=N:-1:2  
 D(k-1)=(k-1)\*A(k)+D(k)\*X;  
end  
v=D(1);  
end  
  
%test  
>> A=[3,2,1];  
>> X=1;  
>> [v] = Horner2(A,X)  
v =  
 8

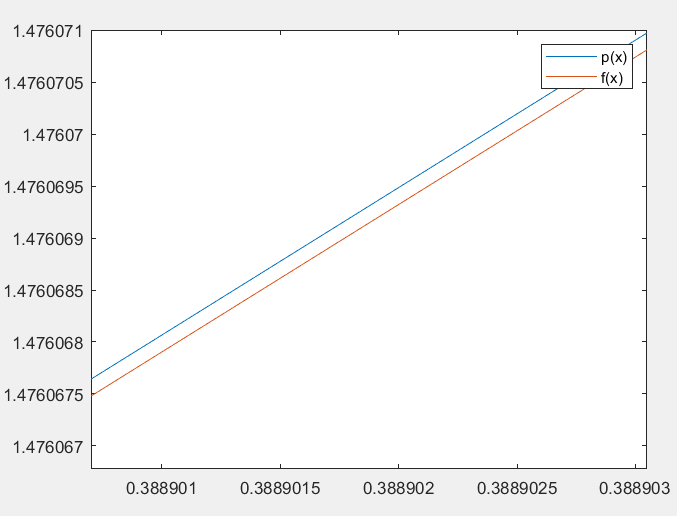
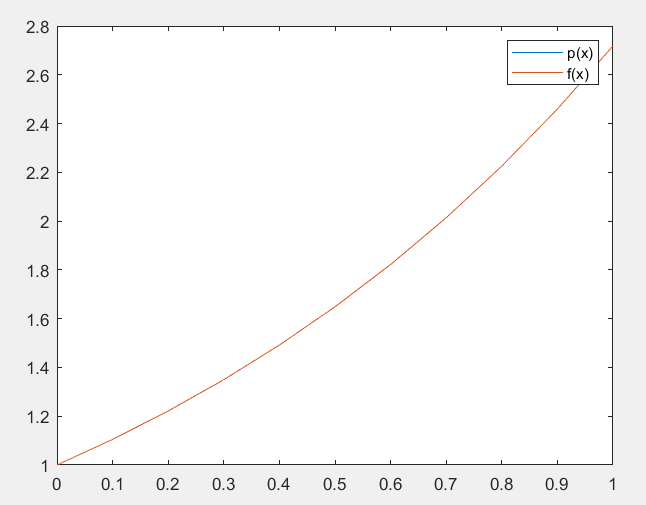
%Horner3.m  
function [v] = Horner3(A,X,C)  
%input - N: Degree of P(x)  
% - A: Coefficients of P(x)  
% - X: Independent variable  
%output - \int P(x)  
A=flip(A);  
N=length(A)-1;  
I=zeros(N+2,1);  
I(N+2)=A(N+1)/(N+1);  
for k=N+1:-1:2  
 I(k)=A(k-1)/(k-1)+I(k+1)\*X;  
end  
I(1)=C+I(2)\*X;  
v=I(1);  
end  
  
%test  
>> A=[3,2,1];  
>> X=1;  
>> C=1;  
>> [v] = Horner3(A,X,C)  
  
v =  
 4



%Horner1.m  
function [v] = Horner1(A,X)  
le=length(X);  
A=flip(A);  
N=length(A)-1;  
for i=1:le   
 B=zeros(N+1,1);  
 B(N+1)=A(N+1);  
 for k=N:-1:1  
 B(k)=A(k)+B(k+1)\*X(i);  
 end  
 v(i)=B(1);  
end

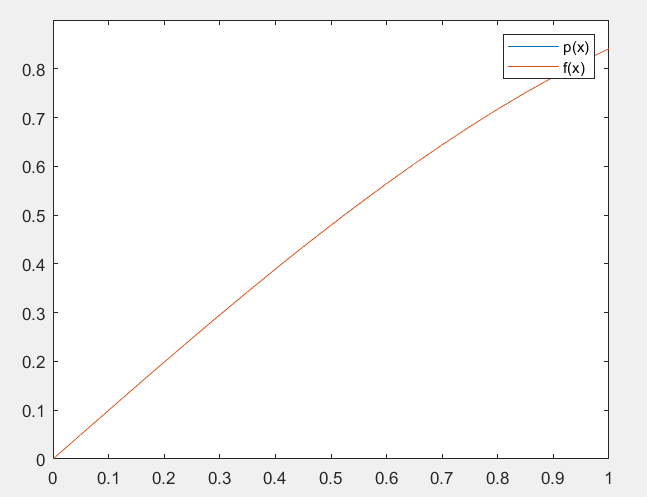
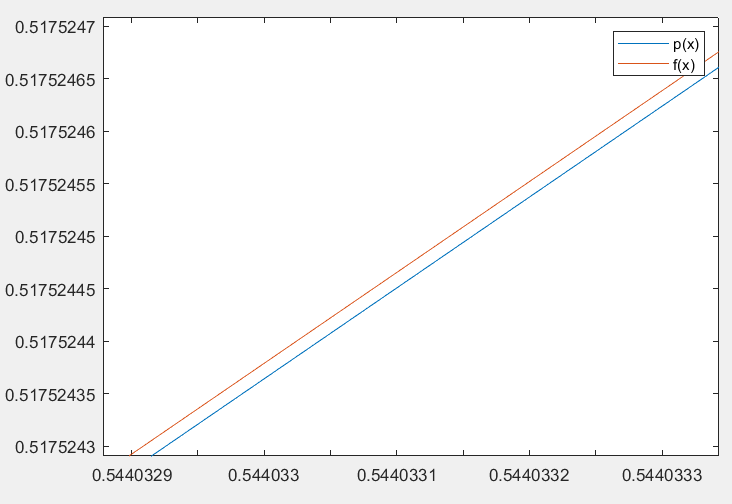
%The following commands are completed in the command line window  
%(i)  
X=[0,0.5,0.4,0.6,0.8,1]';  
A=[X.^0,X.^1,X.^2,X.^3,X.^4,X.^5];  
B=exp(X);  
X = uptrbk(A,B)  
X =  
 1.0000  
 1.0002  
 0.4982  
 0.1724  
 0.0329  
 0.0145  
  
%(ii)  
X=flip(X);  
format long  
[v] = Horner1(X,[0.3,0.4,0.5]) %The horner1 function used here has been improved, and the code is shown above  
v =  
 1.349860279454741 1.491824697641270 1.648721270700128  
exp([0.3,0.4,0.5])  
ans =  
 1.349858807576003 1.491824697641270 1.648721270700128  
% we can see that P(0.4)=f(0.4),P(0.5)=f(0.5)  
  
%(iii)  
[v] = Horner1(X,[-0.1,1.1])  
v =  
 0.904791419021796 3.004147840781791  
exp([-0.1,1.1])  
ans =  
 0.904837418035960 3.004166023946433  
  
%(iv)  
Horner3(X,1,1)-Horner3(X,0,1)  
ans =  
 1.718283695538349  
exp(1)-exp(0)  
ans =  
 1.718281828459046  
   
x=0:0.1:1;  
f=exp(x);  
plot(x,polyval(X,x),x,f)  
legend('p(x)','f(x)')  
  
%(v)  
point=0:0.1:1;  
[v] = Horner1(X,point);  
fp=exp(point);  
[v',fp',fp'-v']  
ans =  
 1.000000000000000 1.000000000000000 0  
 1.105179509812279 1.105170918075648 -0.000008591736631  
 1.221408067143980 1.221402758160170 -0.000005308983811  
 1.349860279454741 1.349858807576003 -0.000001471878738  
 1.491824697641270 1.491824697641270 0  
 1.648721270700128 1.648721270700128 0  
 1.822118800390509 1.822118800390509 0.000000000000000  
 2.013752395897021 2.013752707470477 0.000000311573456  
 2.225540928492468 2.225540928492468 0  
 2.459604486200630 2.459603111156950 -0.000001375043680  
 2.718281828459046 2.718281828459046 0

The graph of (a) (iv)

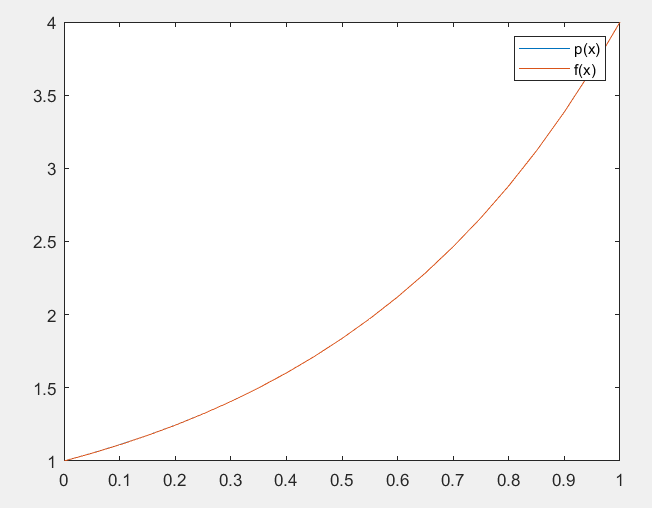
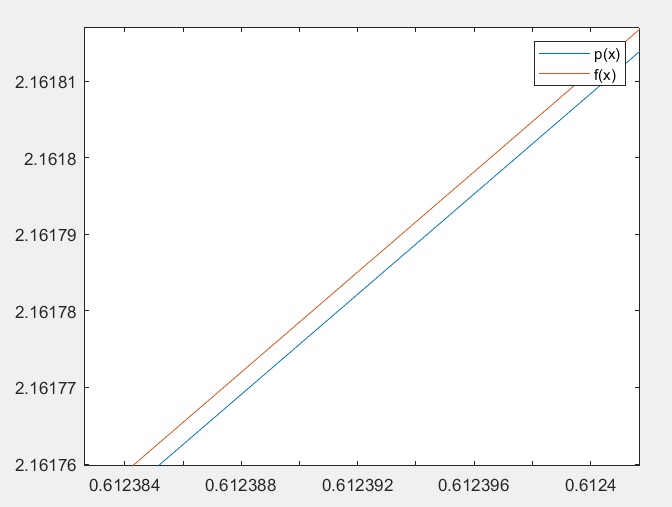
%The following commands are completed in the command line window  
%(i)  
>> X=[0,0.5,0.4,0.6,0.8,1]';  
>> A=[X.^0,X.^1,X.^2,X.^3,X.^4,X.^5];  
>> B=sin(X);  
>> X = uptrbk(A,B)  
X =  
 0  
 0.9999  
 0.0005  
 -0.1682  
 0.0022  
 0.0071  
  
%(ii)  
>> X=flip(X);  
>> format long  
>> [v] = Horner1(X,[0.3,0.4,0.5]) %The horner1 function used here has been improved, and the code is shown above  
v =  
 0.295519778718137 0.389418342308651 0.479425538604203  
>> sin([0.3,0.4,0.5])  
ans =  
 0.295520206661340 0.389418342308651 0.479425538604203  
% we can see that P(0.4)=f(0.4),P(0.5)=f(0.5)  
  
%(iii)  
>> [v] = Horner1(X,[-0.1,1.1])  
v =  
 -0.099820778985218 0.891212976980436  
>> sin([-0.1,1.1])  
ans =  
 -0.099833416646828 0.891207360061435  
  
%(iv)  
>> Horner3(X,1,1)-Horner3(X,0,1)  
ans =  
 0.459697158952903  
>> -cos(1)-(-cos(0))  
ans =  
 0.459697694131860  
   
>> x=0:0.05:1;  
f=sin(x);  
plot(x,polyval(X,x),x,f)  
legend('p(x)','f(x)')  
  
%(v)  
>> point=0:0.1:1;  
[v] = Horner1(X,point);  
fp=sin(point);  
[v',fp',fp'-v']  
ans =  
 0 0 0  
 0.099830982487393 0.099833416646828 0.000002434159435  
 0.198667806192501 0.198669330795061 0.000001524602560  
 0.295519778718137 0.295520206661340 0.000000427943203  
 0.389418342308651 0.389418342308651 0  
 0.479425538604203 0.479425538604203 0  
 0.564642473395035 0.564642473395035 0  
 0.644217781375738 0.644217687237691 -0.000000094138047  
 0.717356090899523 0.717356090899523 0  
 0.783326488732489 0.783326909627483 0.000000420894995  
 0.841470984807897 0.841470984807897 0

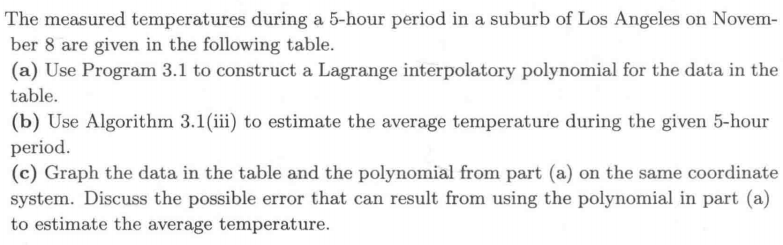
The graph of (b) (iv)

%The following commands are completed in the command line window  
%(i)  
>> X=[0,0.5,0.4,0.6,0.8,1]';  
>> A=[X.^0,X.^1,X.^2,X.^3,X.^4,X.^5];  
>> B=(X+1).^(X+1);  
>> X = uptrbk(A,B)  
X =  
 1.0000  
 1.0144  
 0.8830  
 0.8653  
 -0.2047  
 0.4420  
  
%(ii)  
>> X=flip(X);  
>> format long  
>> [v] = Horner1(X,[0.3,0.4,0.5]) %此处使用的Horner1函数是上面Horner1函数的改进，代码见上方  
v =  
 1.406559443634434 1.601692898202212 1.837117307087384  
>> ([0.3,0.4,0.5]+1).^([0.3,0.4,0.5]+1)  
ans =  
 1.406456673237886 1.601692898202212 1.837117307087384  
% we can see that P(0.4)=f(0.4),P(0.5)=f(0.5)  
  
%(iii)  
>> [v] = Horner1(X,[-0.1,1.1])  
v =  
 0.906503957590380 4.748141857254273  
>> ([-0.1,1.1]+1).^([-0.1,1.1]+1)  
ans =  
 0.909532576082962 4.749638091742242  
  
%(iv)  
>> Horner3(X,1,1)-Horner3(X,0,1)  
ans =  
 2.050575148634692  
>> syms x  
f=inline((x+1)^(x+1));  
g=quad(f,0,1)  
g =  
 2.050446241747296   
   
>> x=0:0.05:1;  
f=(x+1).^(x+1);  
plot(x,polyval(X,x),x,f)  
legend('p(x)','f(x)')  
  
%(v)  
>> point=0:0.1:1;  
[v] = Horner1(X,point);  
fp=(point+1).^(point+1);  
[v',fp',fp'-v']  
ans =  
 1.000000000000000 1.000000000000000 0  
 1.111115623542183 1.110534241054576 -0.000581382487607  
 1.244929494155687 1.244564747203978 -0.000364746951709  
 1.406559443634434 1.406456673237886 -0.000102770396548  
 1.601692898202212 1.601692898202212 0  
 1.837117307087384 1.837117307087384 0  
 2.121250571097592 2.121250571097592 -0.000000000000000  
 2.464671471194466 2.464694899484870 0.000023428290404  
 2.880650097068328 2.880650097068328 0  
 3.385678275712898 3.385570343918481 -0.000107931794417  
 4.000000000000000 4.000000000000000 0

The graph of (c) (iv)

P122 2  
2.  


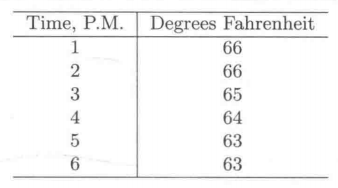
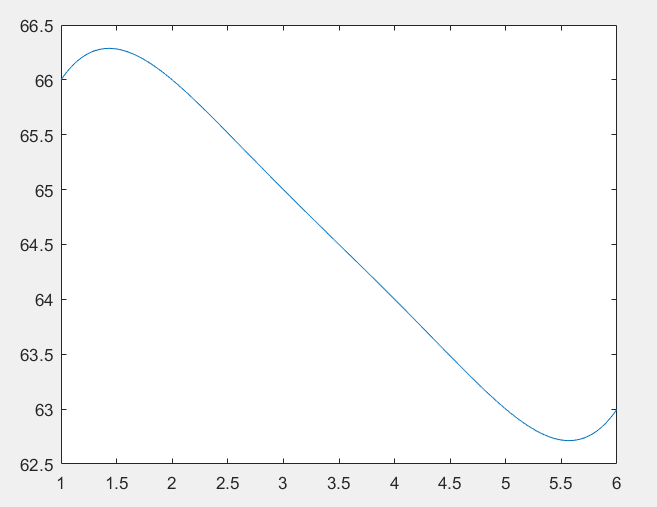


image-20210416200817394

%The following commands are completed in the command line window  
X=[1,2,3,4,5,6];  
Y=[66,66,65,64,63,63];  
format long  
[C] = lagran(X,Y);  
C'  
ans =  
 0.016666666666667  
 -0.291666666666657  
 2.000000000000000  
 -6.708333333333371  
 9.983333333333121  
 61.000000000000000  
x=1:0.01:6;  
plot(x,polyval(C,x))



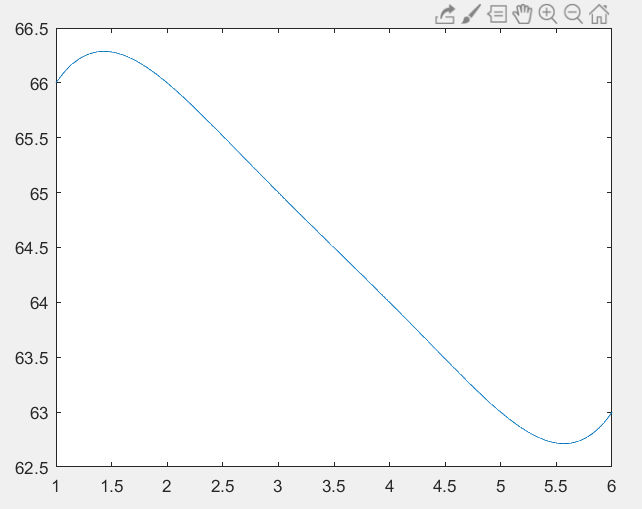
If is a continuous function on the closed, bounded interval,then there is at least one number in , for which

%The following commands are completed in the command line window  
(Horner3(C,6,1)-Horner3(C,1,1))/(6-1);  
ans =  
 64.500000000002260

According to Mean Value Theorem for Integrals，the average-temperature is about .

P132 1  
1.Use Program 3.2 and repeat Problem 2 in Algorithms and Programs from Section 3.3.

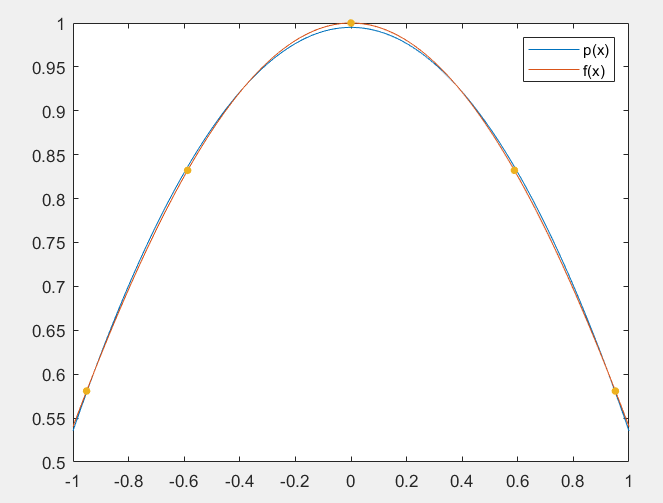
X=[1,2,3,4,5,6];  
Y=[66,66,65,64,63,63];  
format long  
[C] = newpoly(X,Y);  
C'  
ans =  
 0.016666666666667  
 -0.291666666666667  
 2.000000000000000  
 -6.708333333333334  
 9.983333333333334  
 61.000000000000000  
x=1:0.01:6;  
plot(x,polyval(C,x))  
(Horner3(C,6,1)-Horner3(C,1,1))/(6-1)  
ans =  
 64.500000000000028



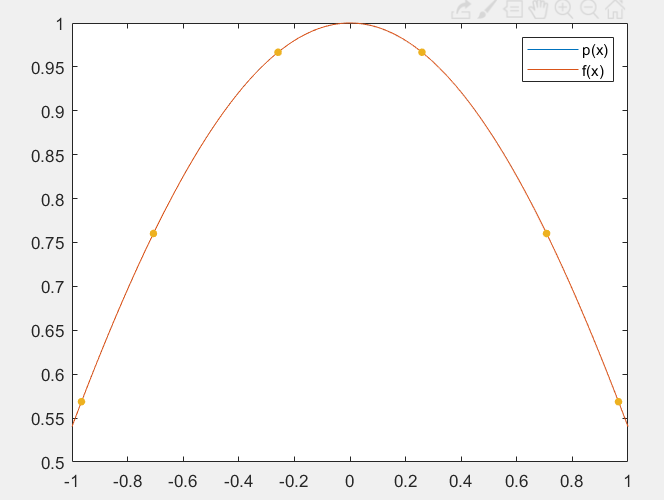
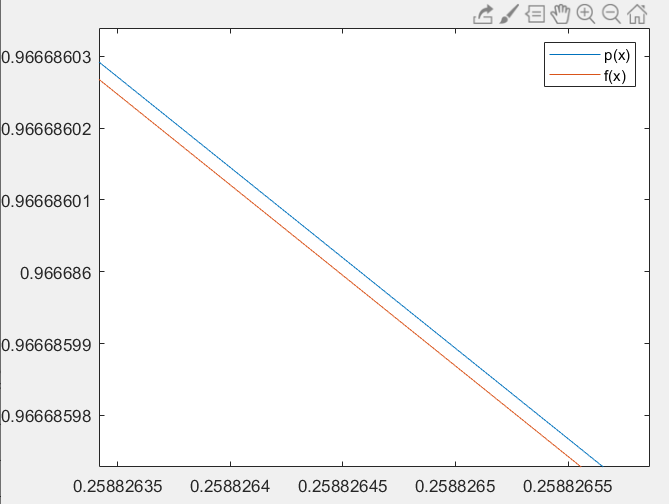
According to Mean Value Theorem for Integrals，the average-temperature is about .

P144 3, 7  
In Problems 3, use Program 3.3 to compute the coefficients for the Chebyshev polynomial approximation to over , when , , , and . In each case, plot and on the same coordinate system.  
Use Program 3.3 () to obtain an approximation for .  
Solution:  
3.

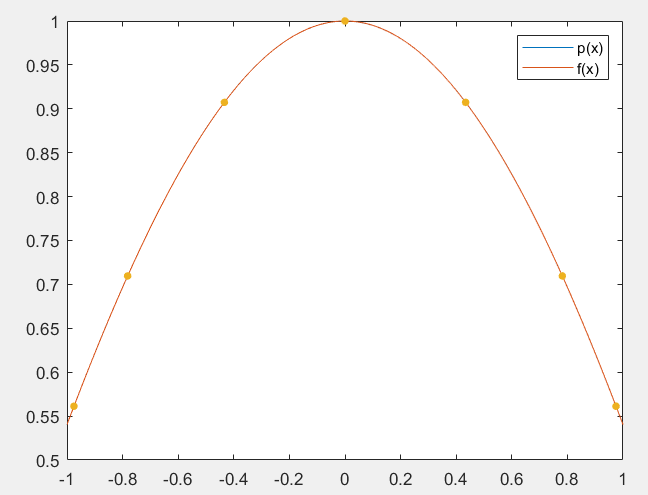
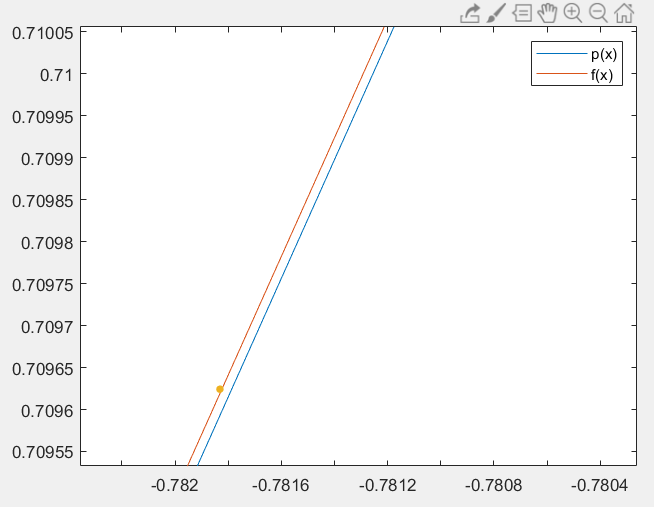
%N=4   
[C,X,Y] = cheby('cos(x)',4,-1,1);   
C'   
   
ans =   
 0.765197687084090   
 -0.000000000000000   
 -0.229807158311684   
 0.000000000000000   
 0.004995154604226   
   
P=C(1)\*poly2sym([1])+C(2)\*poly2sym([1,0])+C(3)\*poly2sym([2,0,-1])+C(4)\*poly2sym([4,0,-3,0]);   
   
x=-1:0.01:1;   
plot(x,polyval(sym2poly(P),x),x,cos(x));   
hold on   
scatter(X,Y,20,'filled');   
legend('p(x)','f(x)')



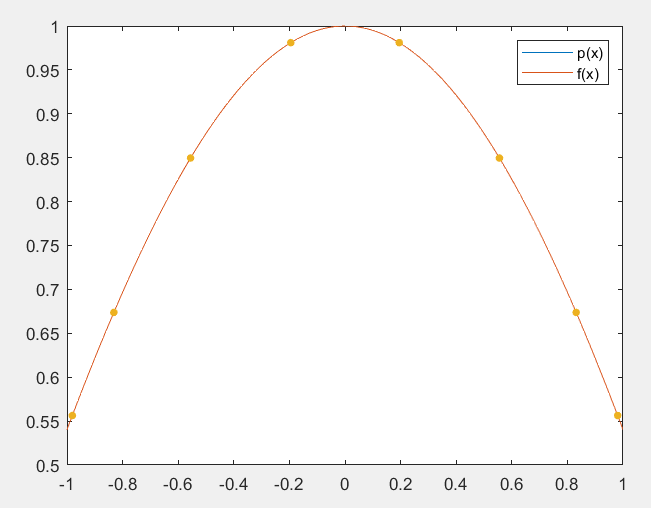
%N=5   
[C,X,Y] = cheby('cos(x)',5,-1,1);   
C'   
   
ans =   
 0.765197686556967   
 0.000000000000000   
 -0.229806969337677   
 0.000000000000000   
 0.004953089481336   
 -0.000000000000000   
   
P=C(1)\*poly2sym([1])+C(2)\*poly2sym([1,0])+C(3)\*poly2sym([2,0,-1])+C(4)\*poly2sym([4,0,-3,0])+C(5)\*poly2sym([8,0,-8,0,1]);   
   
x=-1:0.01:1;   
plot(x,polyval(sym2poly(P),x),x,cos(x));   
hold on   
scatter(X,Y,20,'filled');   
legend('p(x)','f(x)')

%N=6  
[C,X,Y] = cheby('cos(x)',6,-1,1);  
C'  
  
ans =  
 0.765197686557968  
 0.000000000000000  
 -0.229806969864801  
 0.000000000000000  
 0.004953278454343  
 0.000000000000000  
 -0.000042065122888  
  
P=C(1)\*poly2sym([1])+C(2)\*poly2sym([1,0])+C(3)\*poly2sym([2,0,-1])+C(4)\*poly2sym([4,0,-3,0])+C(5)\*poly2sym([8,0,-8,0,1])+C(6)\*poly2sym([16,0,-20,0,5,0]);  
  
x=-1:0.01:1;  
plot(x,polyval(sym2poly(P),x),x,cos(x));  
hold on  
scatter(X,Y,20,'filled');  
legend('p(x)','f(x)')

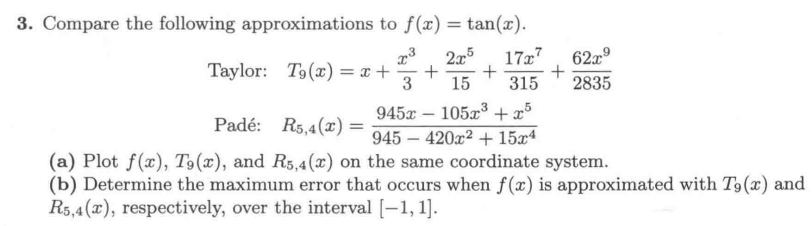
%N=7  
[C,X,Y] = cheby('cos(x)',7,-1,1);  
C'  
  
ans =  
 0.765197686557966  
 -0.000000000000000  
 -0.229806969863800  
 0.000000000000000  
 0.004953277927220  
 -0.000000000000000  
 -0.000041876149882  
 0.000000000000001  
  
P=C(1)\*poly2sym([1])+C(2)\*poly2sym([1,0])+C(3)\*poly2sym([2,0,-1])+C(4)\*poly2sym([4,0,-3,0])+C(5)\*poly2sym([8,0,-8,0,1])+C(6)\*poly2sym([16,0,-20,0,5,0])+C(7)\*poly2sym([32,0,-48,0,18,0,-1]);  
  
x=-1:0.01:1;  
plot(x,polyval(sym2poly(P),x),x,cos(x));  
hold on  
scatter(X,Y,20,'filled');  
legend('p(x)','f(x)')



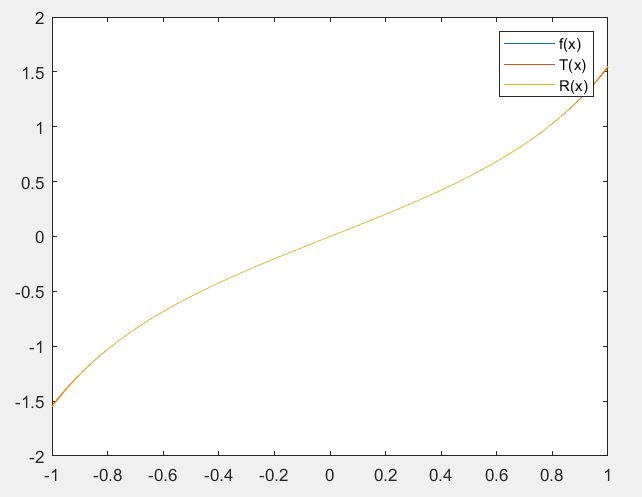
7.

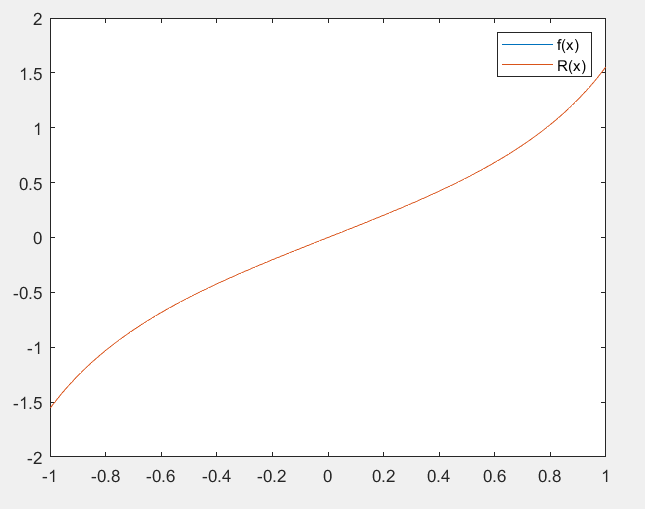
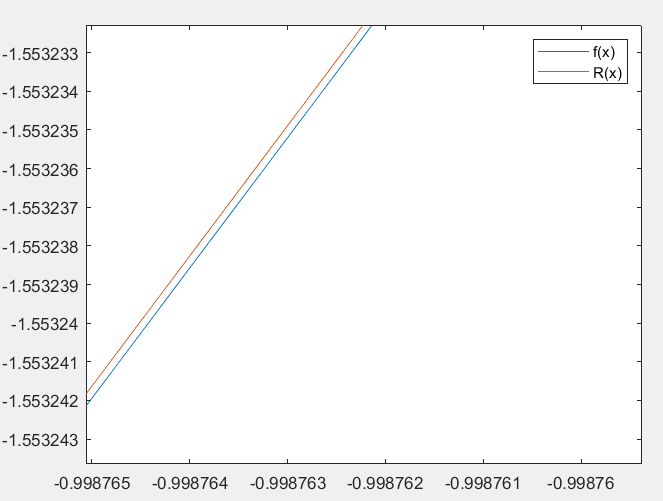
[C,X,Y] = cheby('cos(x.^2)',5,-1,1);   
C'   
   
ans =   
 0.823585327550802   
 0.000000000000000   
 -0.232291660907438   
 0.000000000000000   
 -0.053997234339571   
 -0.000000000000000   
   
P=C(1)\*poly2sym([1])+C(2)\*poly2sym([1,0])+C(3)\*poly2sym([2,0,-1])+C(4)\*poly2sym([4,0,-3,0])+C(5)\*poly2sym([8,0,-8,0,1]);   
   
Horner3(sym2poly(P),1,1)-Horner3(sym2poly(P),0,1)   
ans =   
 0.904615696809253   
   
syms x   
f=inline(cos(x^2));   
g=quad(f,0,1)   
g =   
 0.904524260466284

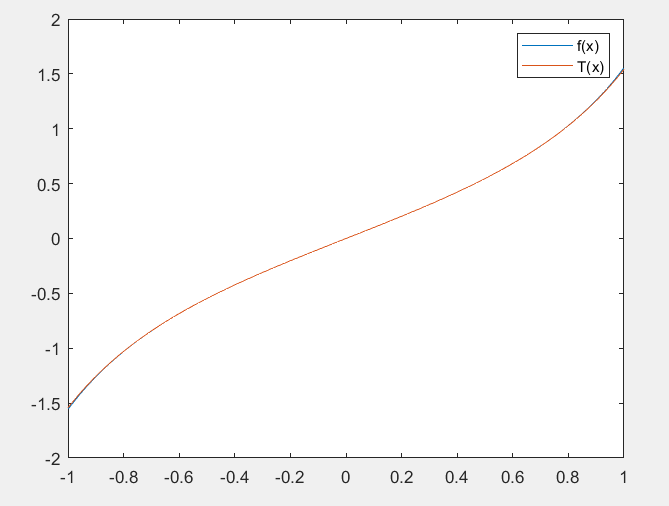
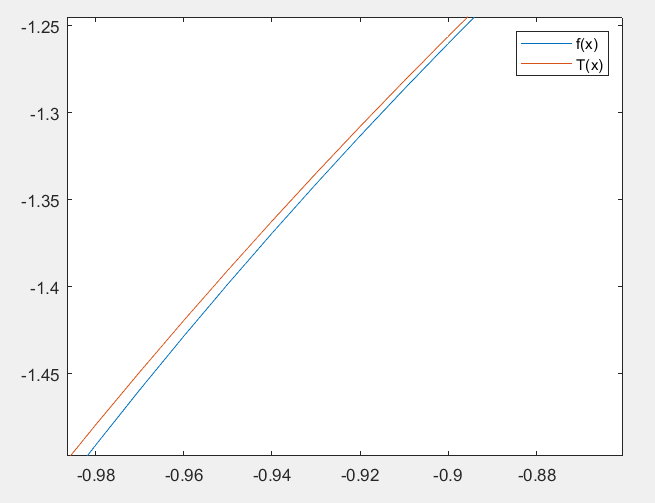
We obtain an approximation value 0.904524260466284 for .  
P149 3



x=-1:0.01:1;   
T=x+x.^3/3+x.^5\*(2/15)+x.^7\*(17/315)+x.^9\*(62/2835);  
R=(945\*x-105\*x.^3+x.^5)./(945-420\*x.^2+15\*x.^4);  
f=tan(x);  
plot(x,f,x,T,x,R)  
legend('f(x)','T(x)','R(x)')  
plot(x,f,x,R)  
legend('f(x)','R(x)')  
plot(x,f,x,T)  
legend('f(x)','T(x)')

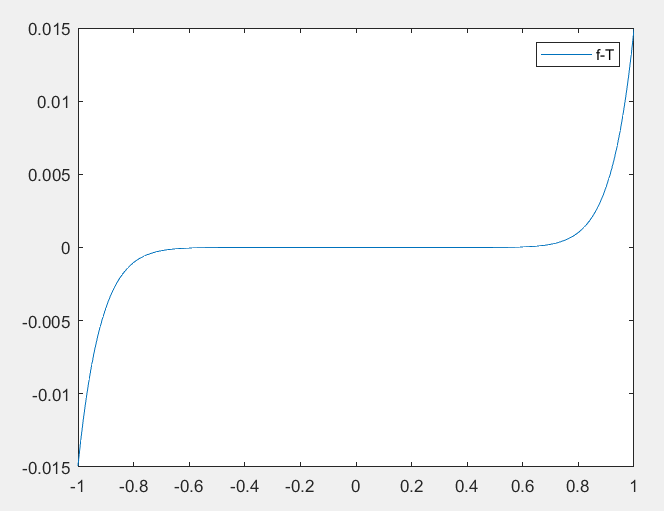
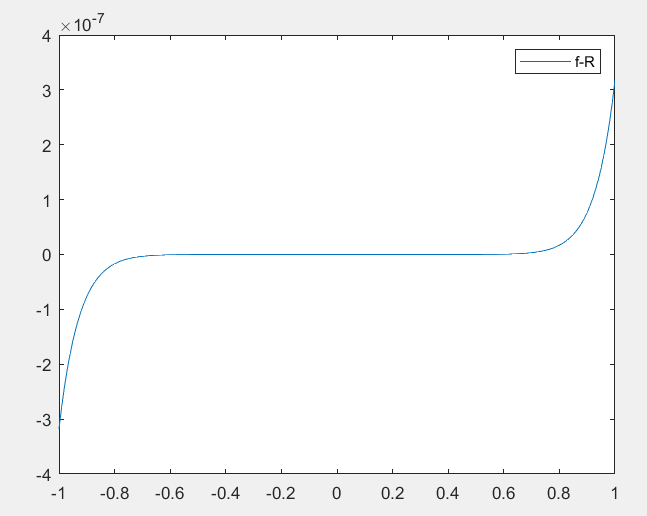


It can be seen that the fitting effect of Taylor approximation at the endpoint is not as good as that of Pade approximation

plot(x,f-T)   
legend('f-T')   
plot(x,f-R)   
legend('f-R')   
   
ER=(f-R)';   
ER(1)   
ans =   
 -3.172474949408866e-07   
ET=(f-T)';   
ET(1)   
ans =   
 -0.014903315483827

It can be seen from the image that the maximum error is obtained at the end point., .