Table of Contents

G12ISC 2018-2019 Coursework 3

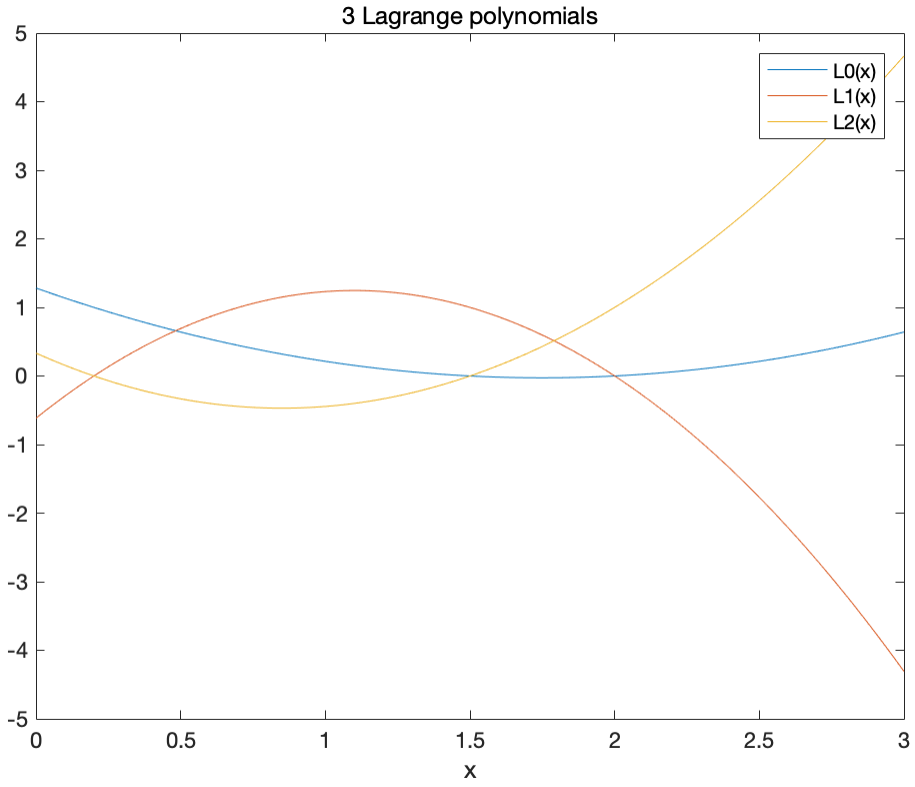
Subject: Polynomial interpolation Student Name: Lan SHUI Student ID: 4336432

```
clear all
close all
clc
```

Question 1

Following code produces one figure consisting of 3 Lagrange polynomials

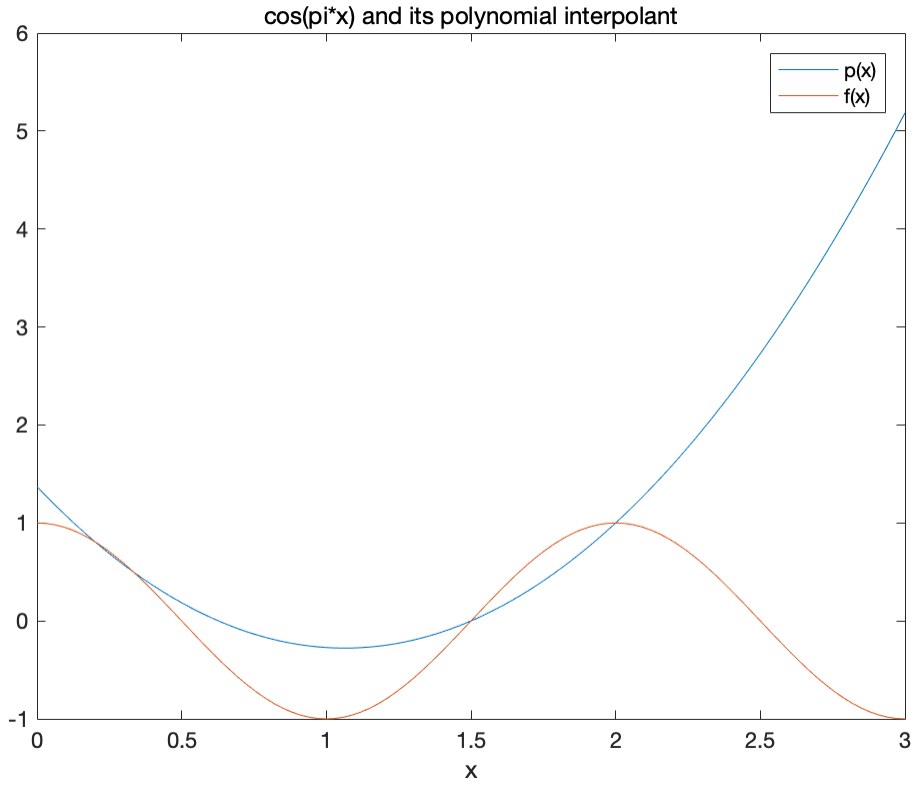
```
% 3 Lagrange polynomials
L0 = @(z) (z-3/2).*(z-2)/(117/50);
L1 = @(z) (z-1/5).*(z-2)/(-13/20);
L2 = @(z) (z-1/5).*(z-3/2)/(9/10);

% Plot the figure
x = linspace(0,3,100);
plot(x,L0(x),x,L1(x),x,L2(x));
% Format the figure
legend('L0(x)','L1(x)','L2(x)');
title('3 Lagrange polynomials');
xlabel('x');
```



Question 2

Following code produces one figure consisting of function f and its interpolant

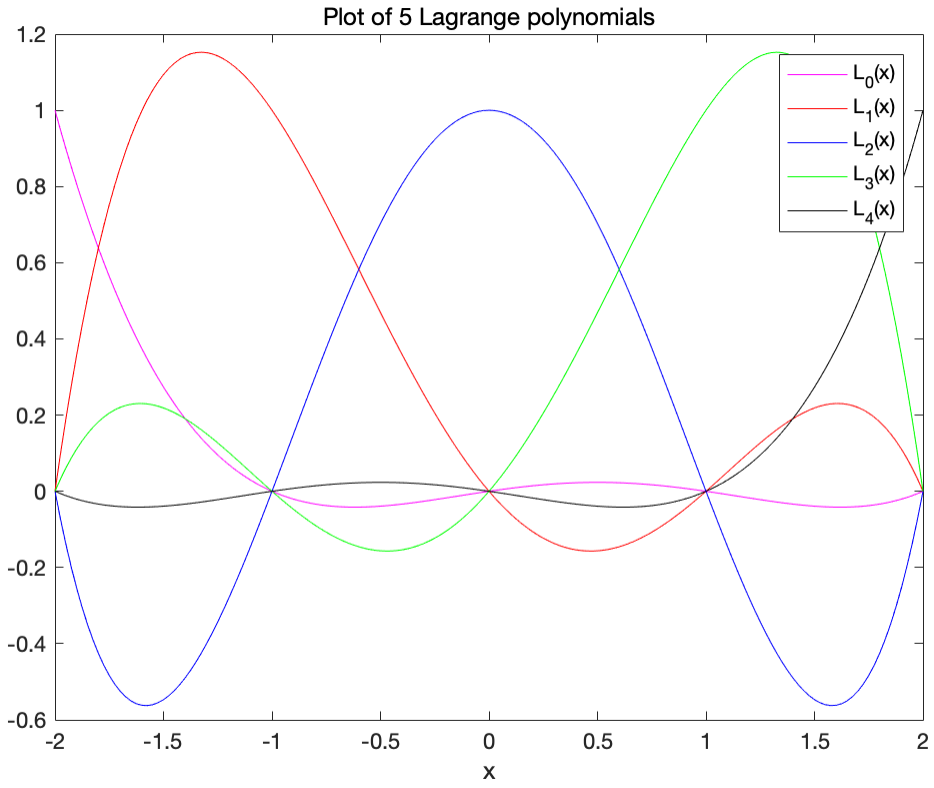
```
% Function f
f = @(z) cos(pi.*z);

% f's polynomial interpolant p in Lagrange's form
f0 = cos((1/5)*pi);
f1 = cos((3/2)*pi);
f2 = cos(2*pi);
p = @(z) f0*L0(z)+f1*L1(z)+f2*L2(z);

% Plot the figure
plot(x,p(x),x,f(x));
% Format the figure
title('cos(pi*x) and its polynomial interpolant')
legend('p(x)','f(x)');
xlabel('x');
```



Question 5

```
clear all
close all
clc

% Following code produces one figure consisting of 5 Lagrange
 polynomials
% L_0(x), L_1(x), L_2(x), L_3(x) and L_4(x).

% Construct evaluation points x
x = GridEq(160,-2,2);

% Construct vector z which are points used to construct L_k(x)
n = 4;
z = GridEq(n,-2,2);

% Plot the figure
y_0 = LagrPolyn(0,x,z);
y_1 = LagrPolyn(1,x,z);
y_2 = LagrPolyn(2,x,z);
y_3 = LagrPolyn(3,x,z);
y_4 = LagrPolyn(4,x,z);
plot(x,y_0,'m',x,y_1,'r',x,y_2,'b',x,y_3,'g',x,y_4,'k');
% Format the figure
```

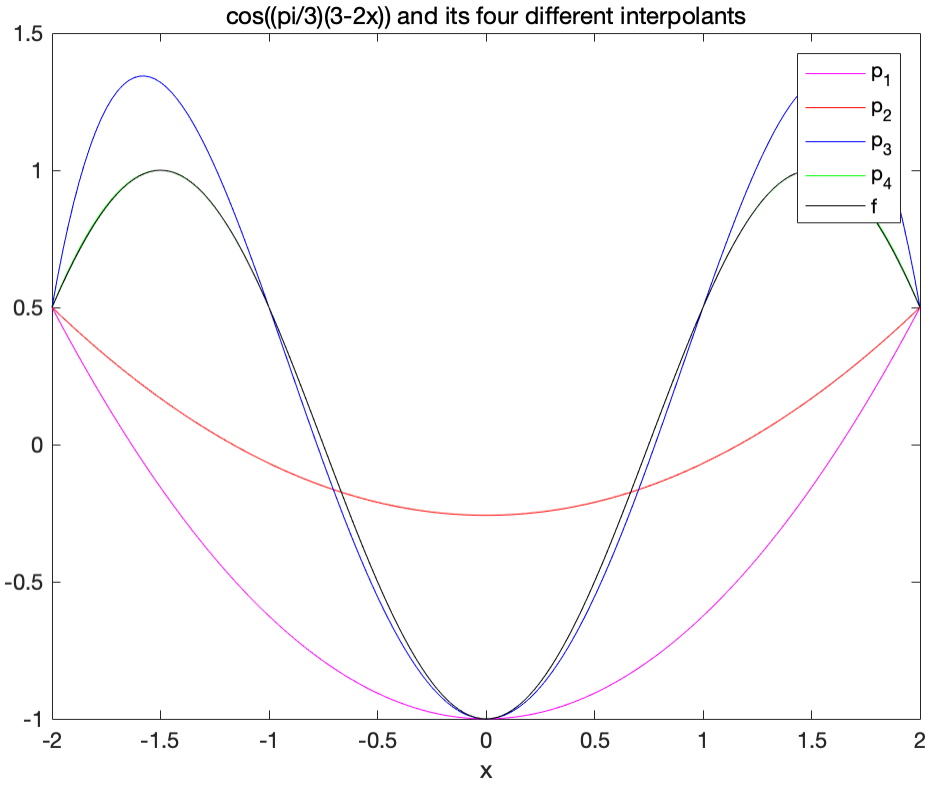
```
title('Plot of 5 Lagrange polynomials')
legend('L_0(x)','L_1(x)','L_2(x)','L_3(x)','L_4(x)');
xlabel('x');
```



Question 7

```
clear all
close all
clc

% Following code produces one figure consisting of function f and its
 4
% different interpolants using PolynInterp.m.

f = @(x) cos((pi/3)*(3-2.*x));

% Construct evaluation points x
x = GridEq(160,-2,2);

% Construct vectors z_1, z_2, z_3 and z_4 which are 4 sets of points
 used
% to construct Lagrange polynomials
z_1 = GridEq(2,-2,2);
z_2 = GridEq(3,-2,2);
z_3 = GridEq(4,-2,2);
z_4 = GridEq(8,-2,2);
```

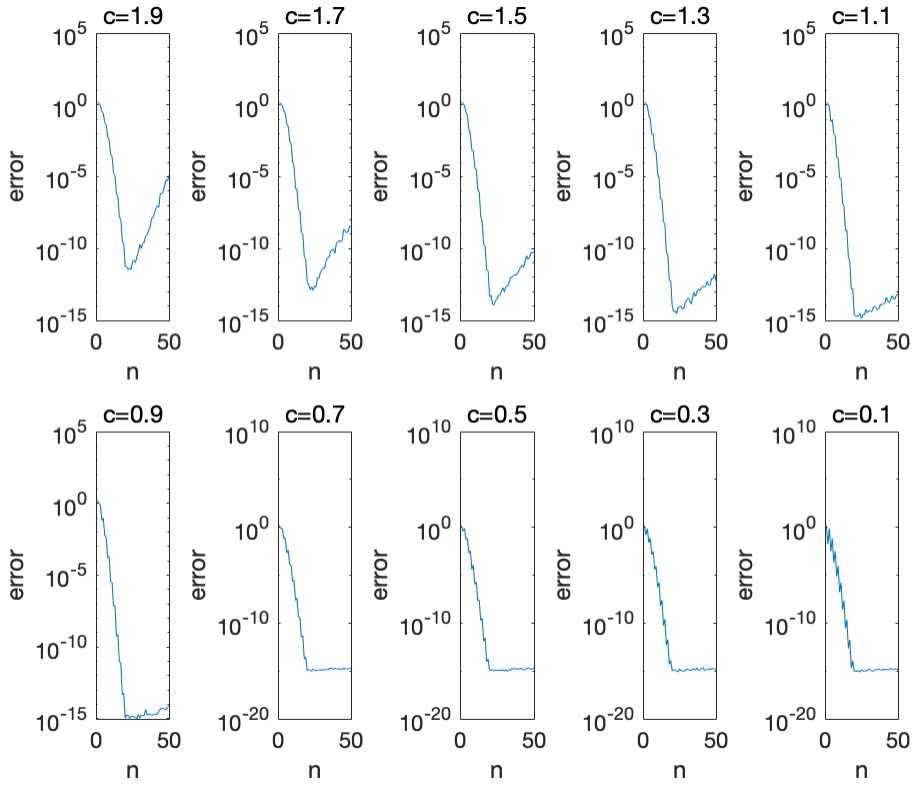
```
% Plot the figure
p_1 = PolynInterp(f,x,z_1);
p_2 = PolynInterp(f,x,z_2);
p_3 = PolynInterp(f,x,z_3);
p_4 = PolynInterp(f,x,z_4);
plot(x,p_1,'m',x,p_2,'r',x,p_3,'b',x,p_4,'g',x,f(x),'k');
% Format the figure
title('cos((pi/3)(3-2x)) and its four different interpolants')
legend('p_1','p_2','p_3','p_4','f');
xlabel('x');
```



Question 9

```
% Following code produces one figure of error for polynomial
 interpolation
% versus the number of points used to construct it.

f = @(x) cos((pi/3)*(3-2.*x));

% Construct a large range of n
n = (1:50);

% Initialize the error
e = zeros(1,50);
```

```
% Compute the error under different n
for i = 1:50
    z = GridEq(i,-2,2);
    % vector containing points used to construct the interpolation
    e(i) = PolyInterpolError(-2,2,f,z);
end

% Plot the figure
semilogy(n,e);
% Format the figure
title('error of interpolation versus the number of points')
xlabel('n');
ylabel('e');

% From the plot, we can see that P_n dosen't converge to f. And the
 error
% tends to increase after n is larger than approximately 23.

% Such huge error is called Runge phenomenon and the error appears at
 the
% edge of the interval. So we can figure out a subinterval [-c,c]
 where the
% P_n dose converge to f.


% Construct a range of c in [0,2]
c = linspace(1.9,0.1,10);

% Plot error in different intervals
f = @(x) cos((pi/3)*(3-2.*x));
n = (1:50);
e = zeros(1,50);
for j = 1:10
    for i = 1:50
    z = GridEq(i,-2,2);
    e(i) = PolyInterpolError(-c(j),c(j),f,z);
    end
    subplot(2,5,j);
    semilogy(n,e);
    xlabel('n');
    ylabel('error');
    title(['c=',num2str(c(j))]);
end

% From the diagram, the largest value of c is approximately 0.7.
```

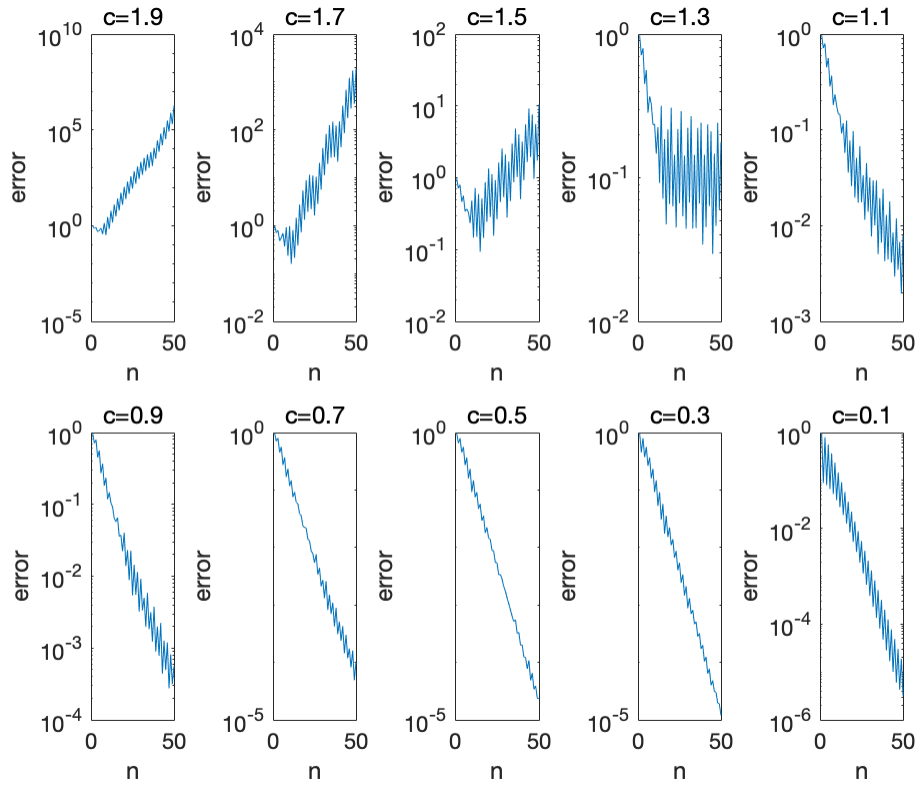


Question 10

```
clear all
close all
clc

% Following code investigates the same issue as Q7-9 but with a
 different
% function.

% Q7'
f = @(x) 1./(1+10*(x.^2));

% Construct evaluation points x
x = GridEq(160,-2,2);

% Construct vectors z_1, z_2, z_3 and z_4 which are 4 sets of points
 used
% to construct Lagrange polynomials
z_1 = GridEq(2,-2,2);
z_2 = GridEq(3,-2,2);
z_3 = GridEq(4,-2,2);
z_4 = GridEq(8,-2,2);

% Plot the figure
```

```
p_1 = PolynInterp(f,x,z_1);
p_2 = PolynInterp(f,x,z_2);
p_3 = PolynInterp(f,x,z_3);
p_4 = PolynInterp(f,x,z_4);
plot(x,p_1,'m',x,p_2,'r',x,p_3,'b',x,p_4,'g',x,f(x),'k');
title('1/(1+10*(x^2)) and its four different interpolants')
legend('p_1','p_2','p_3','p_4','f');
xlabel('x');

% Q9'
f = @(x) 1./(1+10.*(x.^2));

% Construct a large range of n
n = (1:50);

% Initialize the error
e = zeros(1,50);

% Compute the error under different n
for i = 1:50
    z = GridEq(i,-2,2);
    % vector containing points used to construct the interpolation
    e(i) = PolyInterpolError(-2,2,f,z);
end

% Plot the figure
semilogy(n,e);
title('error of interpolation versus the number of points')
xlabel('n');
ylabel('e');

% From the plot, we can see that P_n dosen't converge to f. Again we
 can
% figure out a subinterval [-c,c] where the P_n dose converge to f.


% Construct a range of c in [0,2]
c = linspace(1.9,0.1,10);

% Plot error in different intervals
f = @(x) 1./(1+10.*(x.^2));
n = (1:50);
e = zeros(1,50);
for j = 1:10
    for i = 1:50
    z = GridEq(i,-2,2);
    e(i) = PolyInterpolError(-c(j),c(j),f,z);
    end
    subplot(2,5,j);
    semilogy(n,e);
    xlabel('n');
    ylabel('error');
    title(['c=',num2str(c(j))]);
end
```

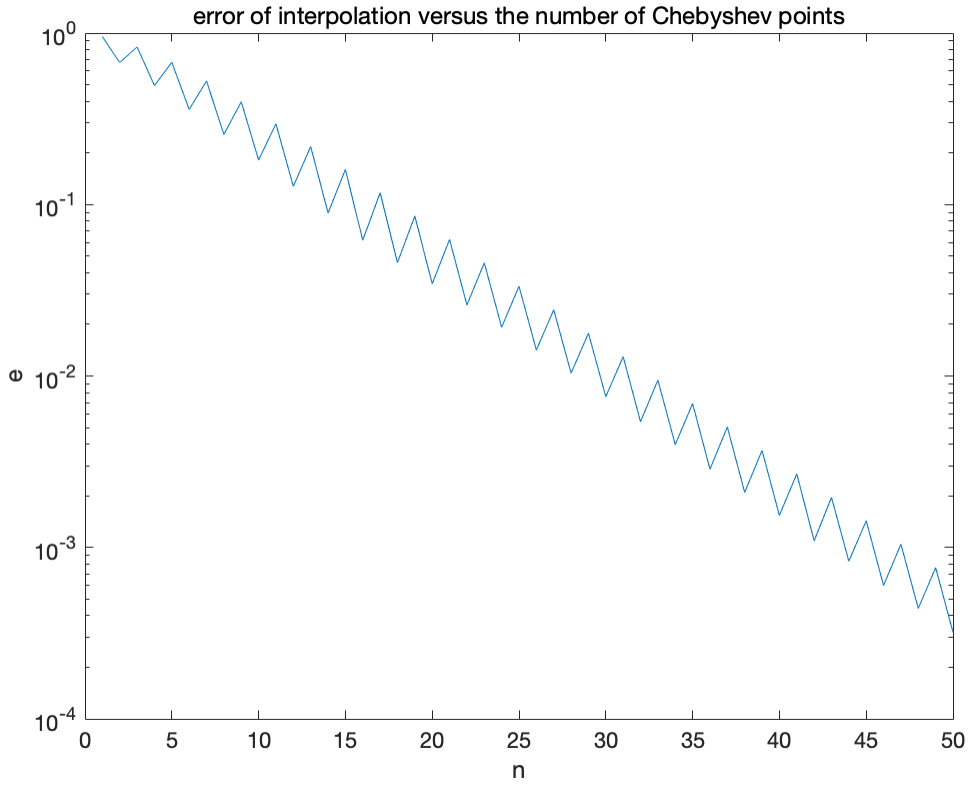
```
% From the diagram, the largest value of c is approximately 1.1.
```



Question 12

```
clear all
close all
clc

% Following code investigates the same issue as Q10 but using
 Chebyshev
% points to contruct the interpolation

% Q7''
f = @(x) 1./(1+10*(x.^2));

% Construct evaluation points x
x = GridEq(160,-2,2);

% Construct vectors z_1, z_2, z_3 and z_4 which are 4 sets of points
 used
% to construct Lagrange polynomials
z_1 = GridCheb(2,-2,2);
z_2 = GridCheb(3,-2,2);
z_3 = GridCheb(4,-2,2);
z_4 = GridCheb(8,-2,2);
```

```
% Plot the figure
p_1 = PolynInterp(f,x,z_1);
p_2 = PolynInterp(f,x,z_2);
p_3 = PolynInterp(f,x,z_3);
p_4 = PolynInterp(f,x,z_4);
plot(x,p_1,'m',x,p_2,'r',x,p_3,'b',x,p_4,'g',x,f(x),'k');
title('1/(1+10*(x^2)) and its four interpolants using Chebyshev
 points')
legend('p_1','p_2','p_3','p_4','f');
xlabel('x');

% Q9''
f = @(x) 1./(1+10*(x.^2));

% Construct a large range of n
n = (1:50);

% Initialize the error
e = zeros(1,50);

% Compute the error under different n
for i = 1:50
    z = GridCheb(i,-2,2);
    % vector containing points used to construct the interpolation
    e(i) = PolyInterpolError(-2,2,f,z);
end

% Plot the figure
semilogy(n,e);
title('error of interpolation versus the number of Chebyshev points')
xlabel('n');
ylabel('e');

% From the plot, we can see that P_n converges to f because the error
 keeps
% decreasing as the number of Chebyshev points increases.
```



Published with MATLAB® R2017a