Computer Problem #2: Interpolation Package

By: Adam Headley 3 November 2014

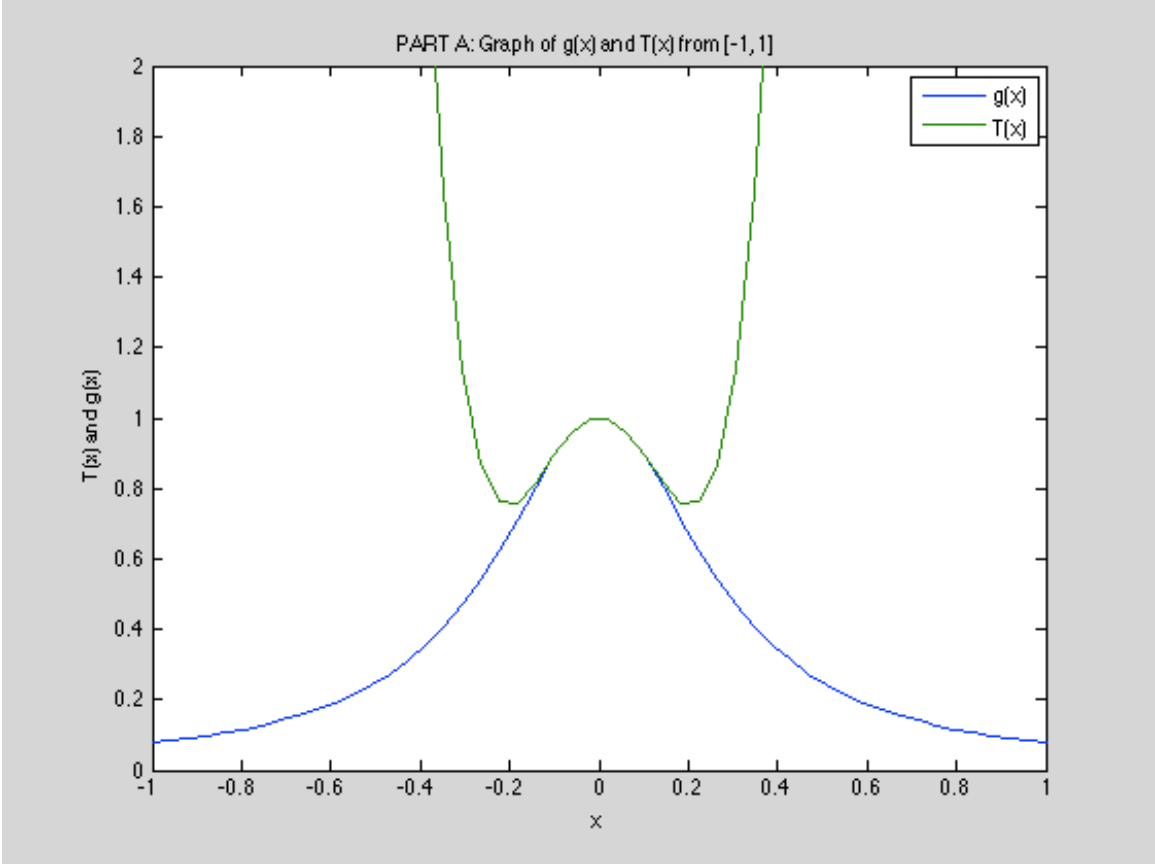
A

Taylor series expansion with remainder term about $x = 0$ of the function $g(x) = \frac{1}{1+12x^2}$.

5th order Taylor Polynomial: $T(x) = 144x^4 - 12x^2 + 1$.

Remainder Term: $R(x) = \mathrm{x}^6 \left(\frac{497664\mathrm{z}^2}{(12\xi(x)^2+1)^5} - \frac{19906560\mathrm{z}^4}{(12\xi(\mathrm{x})^2+1)^6} + \frac{191102976\mathrm{z}^6}{(12\xi(x)^2+1)^7} - \frac{1728}{(12\xi(x)^2+1)^4} \right)$

$\Rightarrow T(x) + R(x) = \left(\frac{497664\mathrm{z}^2}{(12\xi(x)^2+1)^5} - \frac{19906560\mathrm{z}^4}{(12\xi(x)^2+1)^6} + \frac{191102976\mathrm{z}^6}{(12\xi(x)^2+1)^7} - \frac{1728}{(12\xi(x)^2+1)^4} \right) \mathrm{x}^6 + 144\mathrm{x}^4 - 12\mathrm{x}^2 + 1.$



B

See the Appendix for the function scripts of **COEFF.m** and **EVAL.m**

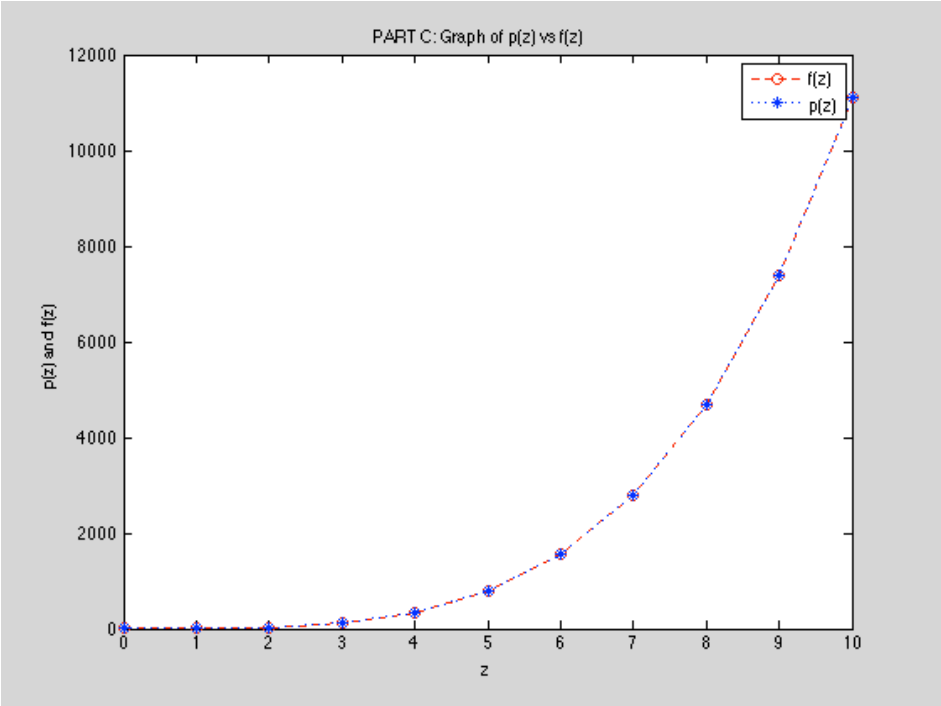
C

By applying these functions to the following case: $n = 4$; $x_i = 1, 2, 4, 8, 16$; $y_i = f(x_i)$, where $f(x) = 1 + x^2 + x^3 + x^4$ the result is a table of coefficients and a graph of f and p. Note that in this case $p(x)$ should be identical to $f(x)$, but written in a different form. Given a set of $n + 1$ data points (x_i, y_i) where no two x_i are the same, one is looking for a polynomial p of degree at most n with the property $p(x_i) = y_i$, $i = 0, \ldots, n$ such that a polynomial p exists and is unique. If one were to simplify

$$p(z) = 26z + 43(z-1)(z-2) + 16(z-1)(z-2)(z-4) + (z-1)(z-2)(z-4)(z-8) - 21$$

they would get $f(z)$, where z is evenly spaced points, in this case 11 points between 0 and 10.

i	c_i
0	5
1	26
2	43
3	16
4	1



D

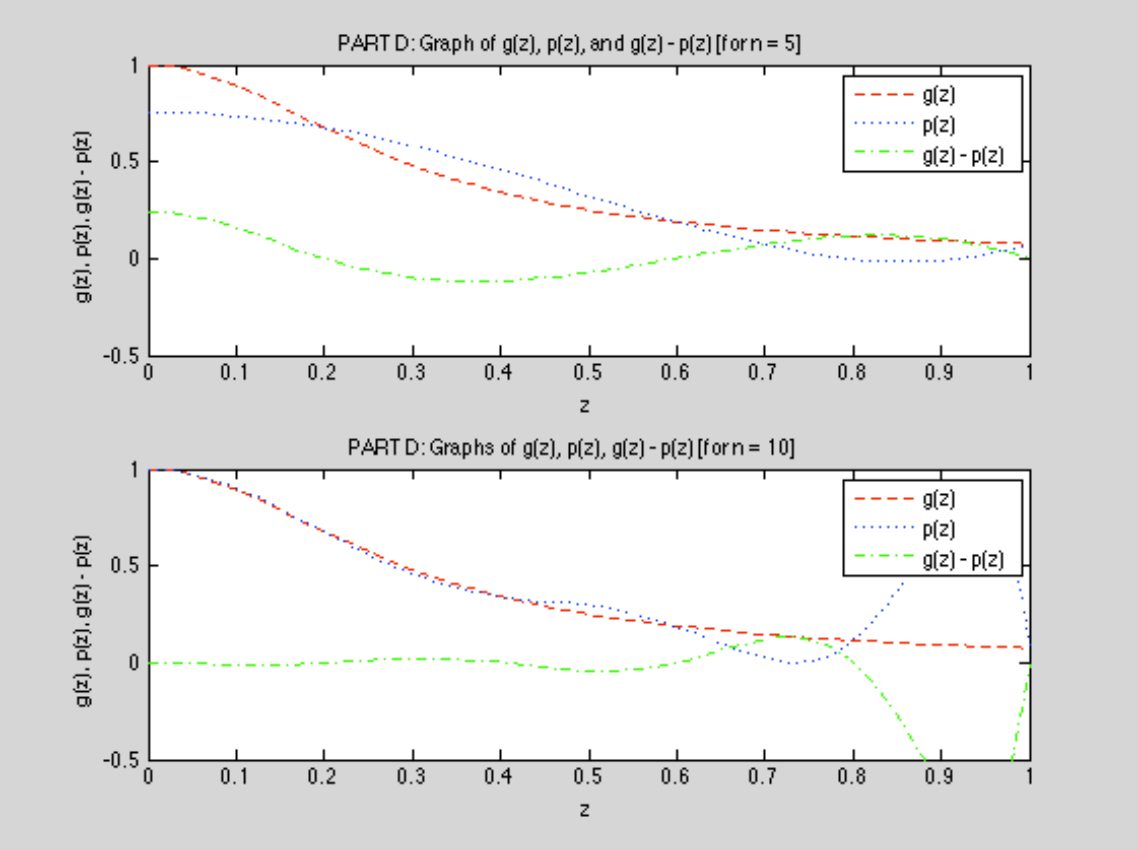
For each of $n = 5, 10$ $x_i = 1 + \frac{2i}{n}$, $i = 0,1, \dots, n$ so that $x_i \in [-1,1]$ are equally spaced. $y_i = g(x_i)$, where $g(x) = \frac{1}{(1+12x^2)}$. The coefficients c_i are in the following table for when $i = 0,1, \dots, n$.

i	c_i (n = 5)
0	0.07692307692307693
1	0.27761711972238284
2	1.1770590717959142
3	-2.2509496193706724
4	1.4068435121066705
5	-4.440892098500626e-16

Max absolute error: 0.243102558892

i	c_i (n = 10)
0	0.07692307692307693
1	0.19142148174406234
2	0.43097818989160225
3	0.9844796225241386
4	1.294942569915238
5	-8.705893302493056
6	3.6181304641894783
7	26.85622612594264
8	-68.73915020330553
9	95.91509330693793
10	-95.91509330693793

Max absolute error: 0.794348647072



E

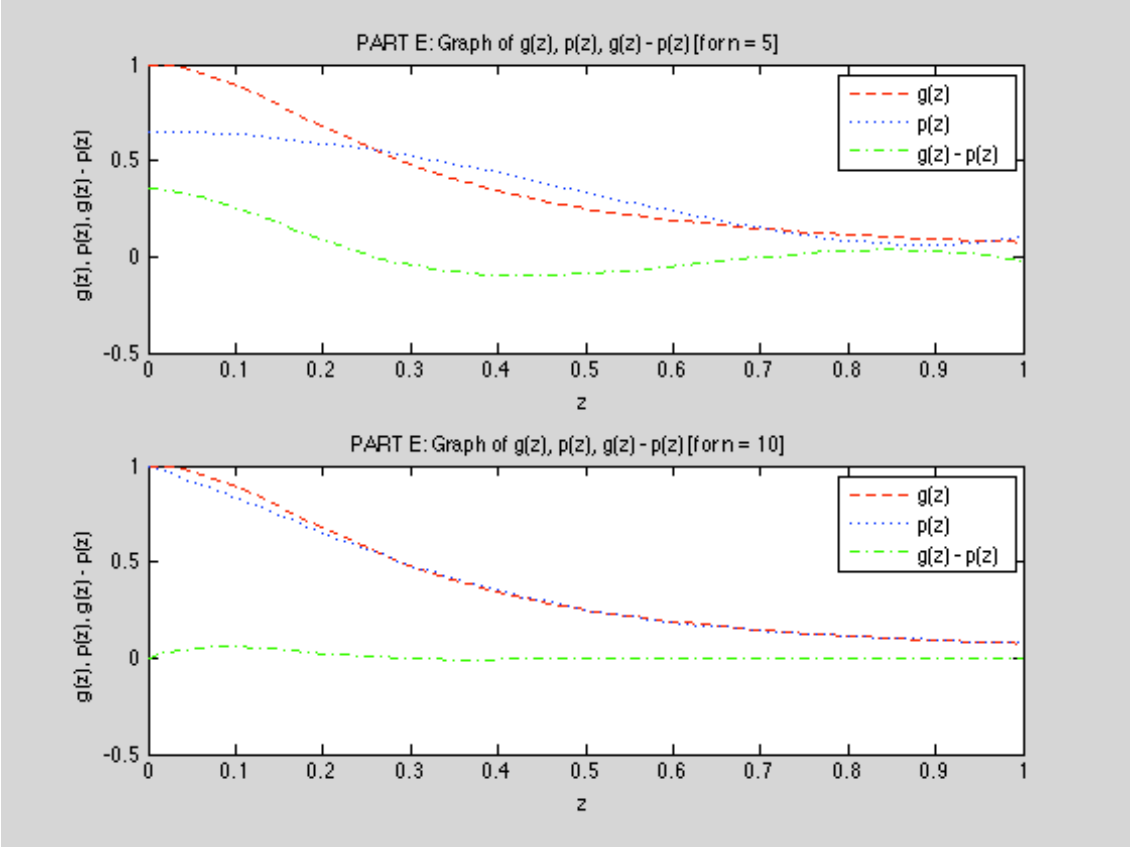
For each of $n = 5, 10$ put $x_i = \cos\left(\frac{(2i+1)}{(n+1)}\frac{\pi}{2}\right)$, $i = 0,1,\ldots,n$; These x_i's are called the Chebyshev abscissae. $y_i = g(x_i)$, where $g(x) = \frac{1}{(1+12x^2)}$. The coefficients c_i are in the following table for when $i = 0,1,\ldots,n$.

i	c_i (n = 5)
0	0.08199307169515309
1	-0.23516071291384635
2	0.9656341437984364
3	1.5643941264707057
4	0.9350649350649346
5	-9.195099618698094e-16

Max absolute error: 0.350649350649

i	c_i (n = 10)
0	0.07838859486798773
1	-0.16348421210906214
2	0.29622267111512607
3	-0.5871162119511127
4	1.1468871401234821
5	3.1680091866668265
6	-1.5632793235400972
7	-11.85650229775745
8	-21.096365813629333
9	-25.557668260466595
10	-25.82048355297294

Max absolute error: 0.0407371696001



F

The Taylor polynomial about 0 approximates $g(z)$ extremely well from -0.1 to 0.1. By observing Part A's graph it is clearly obvious that from $(-\infty, -0.1]$ and $[0.1,\infty)$ the error is massive. In Part D, when n = 5, it is clear that $p(z)$ does a poor job of approximating $g(z)$. This is easily seen by looking at the absolute error $g(z) - p(z)$. Notice it fluctuates above and below the line y = 0 intersecting it three times implying the polynomial simply intersects the function $g(z)$ at three points. However, when n = 10, from z = 0 to z = 0.4, it is clear $p(z)$ does a good job of approximating $g(z)$. Notice the error $g(z) - p(z)$ staying at an almost constant 0, then it starts increasing fluctuation right around z = 0.4. After z = 0.4, the error spikes because of the increase in fluctuation. In Part E, when n = 5, the approximating polynomial does not do a good job until around z = 0.7. The error here decreasingly fluctuates about the line y = 0 as z increases. When n = 10, the interpolating polynomial gives a much better approximation, $g(z)$ and $p(z)$ are almost identical, and the error, $g(z) - p(z)$, hovers at a constant 0. The polynomials in Part E give the best approximation for $g(z)$. My conclusion for part E, is while z is in the interval (-0.289,0.289) the degree of the Taylor approximation required increases as x moves away from 0. A Taylor polynomial whose error is less than 0.04 is not achievable when z is outside of this interval.

Appendix

```
%% A
clear all;
clc;                        % clears workspace and command window

syms x z;                   % let z = ξ(x)

x0 = 0;                     % taylor polynomial about x0 = 0

n = 5;                      % degree of taylor polynomial

g(x) = 1 / (1 + 12*x^2);  % function we're working with

T(x) = taylor(g(x));        % 5th order Taylor expansion at 0

R(x) = (diff(g(z), n+1)/factorial(n+1)) * (x - x0)^(n+1);  % g(z) = g(ξ(x))

fprintf('The 5th order Taylor Polynomial is T(x): ')
disp(T(x))
fprintf('The 5th order remainder term R(x) is: ')
disp(R(x))
fprintf('g(x), T(x) + R(x), of degree 5 is: ')
disp(T(x)+R(x))

% graph of T(x) and g(x)
i = linspace(-1,1,50);       % 50 evenly spaced points in interval [-1,1]
figure
plot(i,g(i),i,T(i))
title('PART A: Graph of g(x) and T(x) from [-1,1]');    % title of graph
xlabel('x');                 % x axis label
ylabel('T(x) and g(x)');    % y axis label
legend('g(x)','T(x)');       % legend of graph
axis([-1,1,0,2])
```

```
%% B
% function scripts for COEFF and EVAL
```

```
function c = COEFF(x,y,n)
% Computes the coefficients, c_i, usig divided differences.  The values n,
% x_i, and y_i are given.  COEFF should not modify any of its arguments
% except c.

F = zeros(n,n); % intializaing F

for i = 0:n
      F(i+1,1) = y(i+1);
end
% this will create a lower triangular matrix
for i = 1:n          % we don't need the first column (the f(x_n) values) of the F matrix; the
loop above did that already
   for j = 1:i
      F(i+1,j+1) = (F(i+1,j) - F(i,j)) / (x(i+1) - x((i-j)+1)); % F(i+1,(j+1)-1) -> F(i+1,j) &
F((i+1)-1,(j+1)-1) -> F(i,j)
   end
end
```

%----------%

```
function P = EVAL(x,c,n,z)
% Returns the values of p(z) where p(x) is the polynomial given.  The values
% of x_i, c_i, and n are given, and should not be modified.

% this below will create a polynomial p(z)
P1 = 0;     % initializing p

for i = 1:n
   P1 = (c(i+1) * prod(z(1:i) - x(1:i))) + P1;
end
P(z) = c(1) + P1     % little p returns the value, big P returns polynomial
```

%----------%

```
%% C
% part a
```

```
% given info
syms x z;
f(x) = 1 + x + x^2 + x^3 + x^4;
n = 4;
x = [1 2 4 8 16];
y = [f(x(1)) f(x(2)) f(x(3)) f(x(4)) f(x(5))];

c = COEFF(x,y,n);       % calling the COEFF function that i wrote

fprintf('Part C (a): \n\n')
fprintf('  i  |  c_i  \n')
fprintf('-----|-------\n')
for i = 0:n
   fprintf('  %d  |  %d  \n',i,c(i+1))
end
fprintf('\n')


%-------------------------------------------------------------------------%


% part b

P(z) = EVAL(x,c,n,z) % returns the value of P(z)

% code below gives graph
z = linspace(0,10,11);      % 11 even points from 0 to 10
figure
plot(z,f(z),'--ro', z, P(z),':b*')
title('PART C: Graph of P(z) vs f(z)');
xlabel('z');
ylabel('P(z) and f(z)');
legend('f(z)','p(z)')



%-------------------------------------------------------------------------%




%% D
% for n = 5
```

```
syms x z;
g(x) = 1/(1 + 12*x^2);
n1 = 5;

% loop below creates an array x1 , for n = 5,
% where each element is x1_i, i = 0,1,...,n
for i = 0:n1
   x1(i+1) = -1 + ((2*i)/n1);
end

% creates array y1, for n = 5, where each element is g(x1_i)
for i = 0:n1
   y1(i+1) = g(x1(i+1));
end

% coefficients for x1 y1
c1 = COEFF(x1,y1,n1);
fprintf('Part D (n = 5): \n\n')
fprintf('  i  |  c1_i  \n')
fprintf('-----|-------\n')

for i = 0:n
   fprintf('  %d  |  %d  \n',i,c1(i+1))
end
fprintf('\n')

% this loop will give us the polynomial p(z)
P0 = 0;     % initializing p
for i = 1:n1
   P0 = (c1(i+1) * prod(z - x1(1:i))) + P0;
end
P1(z) = c1(1) + P0;     % this is p(z) for n = 5

z1 = linspace(0,1,50);
figure(2)
subplot(2,1,1);
plot(z1,g(z1),'--r',z1,P1(z1),':b',z1,g(z1) - P1(z1),'-.g')
title('PART D: Graph of g(z), p(z), and g(z) - p(z) [for n = 5]');
xlabel('z');
ylabel('g(z), p(z), g(z) - p(z)');
legend('g(z)','p(z)','g(z) - p(z)');
axis([0,1,-0.5,1])

%-------------------------------------------------------------------------%
% for n = 10
n2 = 10;
```

```
% creates array x2, for n = 10, where each element is x2_i, i = 0,1,...,n
for i = 0:n2
   x2(i+1) = -1 + ((2*i)/n2);
end

% creates array y2, for n = 10, where each element is g(x2_i)
for i = 0:n2
   y2(i+1) = g(x2(i+1));
end

% coefficients for x2 y2
c2 = COEFF(x2,y2,n2);
fprintf('Part D (n = 10): \n\n')
fprintf('  i  |  c2_i  \n')
fprintf('-----|-------\n')
for i = 0:n
   fprintf('  %d  |  %d  \n',i,c2(i+1))
end
fprintf('\n')

% this loop will give us the polynomial p(z)
P0 = 0;         % initializing p

for i = 1:n2
   P0 = (c2(i+1) * prod(z - x2(1:i))) + P0;
end
P2(z) = c2(1) + P0; % p(z) for n = 10

% graph

z2 = linspace(0,1,50);

subplot(2,1,2);
plot(z2,g(z2),'--r',z2,P2(z2),':b',z2,g(z2) - P2(z2),'-.g')
title('PART D: Graphs of g(z), p(z), g(z) - p(z) [for n = 10]');
xlabel('z');
ylabel('g(z), p(z), g(z) - p(z)');
legend('g(z)','p(z)','g(z) - p(z)');
axis([0,1,-0.5,1])

%-------------------------------------------------------------------------%


%% E
% n = 5
```

```
syms x z;
g(x) = 1 / (1 + 12*x^2);
n1 = 5;

% loop to create x1 array (different than PART D)
for i = 0:n1
   x1(i+1) = cos(( ((2*i)+1)/(n1+1)) * pi/2);
end

% y1 array (different than PART D)
for i = 0:n1
   y1(i+1) = g(x1(i+1));
end

% coeff x1 y1
 c1 = COEFF(x1,y1,n1);
% prints out table
fprintf('Part E (n = 5): \n\n')
fprintf('  i  |  c1_i  \n')
fprintf('-----|-------\n')
for i = 0:n
   fprintf('  %d  |  %d  \n',i,c1(i+1))
end
fprintf('\n')

% this loop will give us the polynomial p(z)
P0 = 0;         % initializing p

for i = 1:n1
   P0 = (c1(i+1) * prod(z - x1(1:i))) + P0;
end
P1(z) = c1(1) + P0;     % p(z) for n = 5

% graph for n = 5
z1 = linspace(0,1,50);
figure
subplot(2,1,1);
plot(z1,g(z1),'--r',z1,P1(z1),':b',z1,g(z1) - P1(z1),'-.g')
title('PART E: Graph of g(z), p(z), g(z) - p(z) [for n = 5]');
xlabel('z');
ylabel('g(z), p(z), g(z) - p(z)');
legend('g(z)','p(z)','g(z) - p(z)');

%-------------------------------------------------------------------------%
`

n2 = 10
```

```
% loop to create x2 array (different than PART D)
for i = 0:n2
   x2(i+1) = cos(( ((2*i)+1)/(n2+1)) * pi/2);
end

% y2 array (different than PART D)
for i = 0:n2
   y2(i+1) = g(x2(i+1));
end

% coeff x2 y2
 c2 = COEFF(x2,y2,n2);

% prints out table
fprintf('Part E (n = 10): \n\n')
fprintf('  i  |  c2_i  \n')
fprintf('-----|-------\n')
for i = 0:n
   fprintf('  %d  |  %d  \n',i,c2(i+1))
end
fprintf('\n')

% this loop will give us the polynomial p(z)
P0 = 0;      % initializing p

for i = 1:n1
   P0 = (c2(i+1) * prod(z - x2(1:i))) + P0;
end
P2(z) = c2(1) + P0;         % p(z) for n = 10

% graph for n = 10
z1 = linspace(0,1,50);
subplot(2,1,2);
plot(z2,g(z2),'--',z2,P2(z2),':b',z2,g(z2) - P2(z2),'-.g')
title('PART E: Graph of g(z), p(z), g(z) - p(z) [for n = 10]');
xlabel('z');
ylabel('g(z), p(z), g(z) - p(z)');
legend('g(z)','p(z)','g(z) - p(z)');

%------------------------------------------------------------------------%
```