Homework 2

1a) Degree 1:

$P_1(x) = f(x_1) \cdot \frac{(x-x_2)}{x_1-x_2} + f(x_2) \cdot \frac{(x-x_1)}{x_2-x_1}$

$= 1.64872 \cdot \frac{(x-0.5)}{0.25-0.5} + 2.71828 \cdot \frac{(x-0.25)}{0.5-0.25}$

$P_1(0.43) = 1.64872 \cdot \frac{(0.43-0.5)}{0.25-0.5} + 2.71828 \cdot \frac{(0.43-0.25)}{0.5-0.25}$

$= 2.4188032$

Degree 2:

$P_2(x) = f(x_1) \cdot \frac{(x-x_2)(x-x_3)}{(x_1-x_2)(x_1-x_3)} + f(x_2) \cdot \frac{(x-x_1)(x-x_3)}{(x_2-x_1)(x_2-x_3)} + f(x_3) \cdot \frac{(x-x_1)(x-x_2)}{(x_3-x_1)(x_3-x_2)}$

$P_2(0.43) = 1.64872 \cdot \frac{(0.43-0.5)(0.43-0.75)}{(0.25-0.5)(0.25-0.75)} + 2.71828 \cdot \frac{(0.43-0.25)(0.43-0.75)}{(0.5-0.25)(0.5-0.75)}$

$+ 4.48169 \cdot \frac{(0.43-0.25)(0.43-0.5)}{(0.75-0.25)(0.75-0.5)}$

$= 2.34886312$

Degree 3:

$P_3(x) = f(x_0) \cdot \frac{(x-x_1)(x-x_2)(x-x_3)}{(x_0-x_1)(x_0-x_2)(x_0-x_3)} + f(x_1) \cdot \frac{(x-x_0)(x-x_2)(x-x_3)}{(x_1-x_0)(x_1-x_2)(x_1-x_3)}$

$+ f(x_2) \cdot \frac{(x-x_0)(x-x_1)(x-x_3)}{(x_2-x_0)(x_2-x_1)(x_2-x_3)} + f(x_3) \cdot \frac{(x-x_0)(x-x_1)(x-x_2)}{(x_3-x_0)(x_3-x_1)(x_3-x_2)}$

$P_3(0.43) = 2.3606647408$

b) Absolute error:

$|f(0.43) - P_1(0.43)| = 0.05564$

$|f(0.43) - P_2(0.43)| = 0.014298$

Error bound:

$R_1(x) = \frac{f^{(2)}(\xi(x))}{2!}(x-x_0)(x-x_1) = \frac{4e^{2(\xi(x))}}{2} \cdot (0.43-0)(0.43-0.5)$

for $\xi(x) \in (0, 0.43)$

$= \left| \frac{4e^{2(0.43)}}{2} \cdot 0.0774 \right| = 0.365817$

$R_2(x) = \frac{f^{(3)}(\xi(x))}{3!}(x-x_0)(x-x_1)(x-x_2)$

$= \frac{8e^{2(\xi(x))}}{6} \cdot (0.43-0)(0.43-0.25)(0.43-0.5)$

for $\xi(x) \in (0, 0.43)$

$= \left| \frac{8e^{2(0.43)}}{6} \cdot -0.005418 \right|$

$= 0.017071$

Both error bounds for $n=1$ and $n=2$ are higher than the absolute error

$$\frac{f[x_2]-f[x_1]}{x_2-x_1} = \frac{2.71828-1.64872}{0.5-0.25} = 4.27824$$

$$\frac{f[x_2,x_3]-f[x_1,x_2]}{x_3-x_1} = \frac{7.05364-4.27824}{0.75-0.25} = 5.5508$$

1c.

$$\begin{array}{llllll}
0 = x_0 & f[x_0]=f(x_0)=1 & & & & \\
 & & f[x_0,x_1] \; 2.59488 & & & \\
0.25 = x_1 & f[x_1]=f(x_1)=1.64872 & & f[x_0,x_1,x_2] \; 3.36672 & & \\
 & & f[x_1,x_2] \; 4.27824 & & f[x_0,x_1,x_2,x_3] & 2.3606047 \\
0.5 = x_2 & f[x_2]=f(x_2)=2.71828 & & f[x_1,x_2,x_3] \; 5.5508 & & \\
 & & f[x_2,x_3] \; 7.05364 & & & \\
0.75 = x_3 & f[x_3]=f(x_3)=4.48169 & & & &
\end{array}$$

$$\frac{f[x_1]-f[x_0]}{x_1-x_0} = \frac{1.64872-1}{0.25-0} = 2.59488$$

$$\frac{f[x_3]-f[x_2]}{x_3-x_2} = \frac{4.48169-2.71828}{0.75-0.5} = 7.05364$$

$$\frac{f[x_1,x_2]-f[x_0,x_1]}{x_2-x_0} = \frac{4.27824-2.59488}{0.5-0} = 3.36672$$

The values do match!

$$f[x_0,x_1,x_2,x_3] = \frac{f[x_1,x_2,x_3]-f[x_0,x_1,x_2]}{x_3-x_0} = 2.91210667$$

$$P_3(0.43) = 1 + 2.59488\cdot(0.43-0) + 3.36672(0.43-0)(0.43-0.25) + 2.91210667\cdot(0.43-0)(0.43-0.25)\cdot(0.43-0.5)$$
$$= 2.3606$$

2.

$$\begin{array}{llllll}
x_0 = 0 & P_0 & & & & \\
 & & P_{0,1}(1.5)=2.5 & & & \\
x_1 = 1 & P_1 & & P_{0,1,2}(1.5)=? \leftarrow 3.25 & & \\
 & & P_{1,2}(1.5)=3.5 & & P_{0,1,2,3}(1.5) & \text{Find} \\
x_2 = 2 & P_2 & & P_{1,2,3}(1.5)=4 & & \\
 & & P_{2,3}(1.5)=? & & & \\
x_3 = 3 & P_3 & & & &
\end{array}$$

$$P_{0,1,2}(1.5) = \frac{(x-x_0)P_{1,2}-(x-x_2)\cdot P_{0,1}}{x_2-x_0} = \frac{(1.5-0)\cdot 3.5-(1.5-2)\cdot 2.5}{2-0}$$
$$= 3.25$$

$$P_{0,1,2,3}(1.5) = \frac{(x-x_0)\cdot P_{1,2,3}-(x-x_3)\cdot P_{0,1,2}}{x_3-x_0}$$
$$= \frac{(1.5-0)\cdot 4-(1.5-3)\cdot 3.25}{3-0}$$
$$= 3.625$$

Continued...

3a)

```
N = length(xpts); %length of vector containing x points
Q = zeros(N,N); %create square matrix (NxN) of all zeroes
Q(:,1) = ypts;  %look through all the rows and first column in the matrix Q
and set this equal to ypts

% Recursively generate Lagrange interpolating polynomials
for i = 1:N-1 %set i to index row 1 and end at row N-1
    for j = 1:i % set j to index first column and end at column (i) which
will form a table with numbers in the lower triangle and zeroes in the upper
triangle of the table
        Q(i+1,j+1) = ((xeval-xpts(i-j+1))*Q(i+1,j)-(xeval-
xpts(i+1))*Q(i,j))...
            /(xpts(i+1)-xpts(i-j+1)); %this is the formula to evaluate using
Neville's method

     end %end the for loop for j
end %end the for loop for i
```

The indexing used in computing Q in the two for loops in the nevilles.m file on MATLAB is equivalent to that of the algorithm on page 120 of the book even though it looks like a different method. When calculating the variable (i), the book iterates from 1 to N (where N is the number of x points) while the algorithm in MATLAB calculates (i) from 1 to N-1. Since the algorithm in MATLAB calculates (i) from 1 to N-1, it needs to start calculating at (i+1) in the Neville's formula so that there is no out of bounds error. This balances out with the algorithm in the book which just calculates (i) from 1 to N so therefore nothing needs to be added to (i) when using the formula. Variable (j) remains the same in both cases where it is being calculated from 1 to (i) so it will not change in the formula.

3b)

```
>> xpts = [1;1.3;1.6;1.9;2.2]

xpts =

   1.0000
   1.3000
   1.6000
   1.9000
   2.2000

>> ypts =
[0.765197686557967;0.620085989561509;0.455402167639381;0.281818559374385;0.110362
266922174]

ypts =
```

```
   0.7652
   0.6201
   0.4554
   0.2818
   0.1104

>> nevilles_mod(xpts,ypts,1.5)

ans =

   0.7652        0         0         0         0
   0.6201   0.5233         0         0         0
   0.4554   0.5103    0.5125         0         0
   0.2818   0.5133    0.5113    0.5118         0
   0.1104   0.5104         0         0         0

>> nevilles(xpts,ypts,1.5)

ans =

   0.7652        0         0         0         0
   0.6201   0.5233         0         0         0
   0.4554   0.5103    0.5125         0         0
   0.2818   0.5133    0.5113    0.5118         0
   0.1104   0.5104    0.5137    0.5118    0.5118
```

The full Neville's table generates results in (5(row), 3(column)), (5, 4), and (5,5) but the modified Neville's table does not. This is because an if condition is set to stop running when successive polynomials Q(i,i) and Q(i-1,i-1) differ in absolute value by less than 10^-3. What this means is that, for example, when finding the absolute value of 0.5133 (located in (4,2) and 0.5137 (which is located in (5,3) in the full Neville's table), we can see that the value is less than 10^-3. So, 0.5137 is not included in Neville's table. This is similar for results that are located in (5,4) and (5,5) in the full Neville's table.

4. After condensing $P(X)$ and $Q(x)$, we can see they are identical equations: x^3-3x+1

$Q(-2) = P(-2) = (-2)^3 - 3(-2) + 1 = -1$

$Q(-1) = P(-1) = (-1)^3 - 3(-1) + 1 = 3$

$Q(0) = P(0) = (0)^3 - 3(0) + 1 = 1$

$Q(1) = P(1) = (1)^3 - 3(1) + 1 = -1$

$Q(2) = P(2) = (2)^3 - 3(2) + 1 = 3$

Since $P(X)$ and $Q(x)$ are equal to the original $f(x)$ results, these two equations interpolate the data. The uniqueness property of interpolating polynomials is not violated because both equations are identical.