Math 104A

Alejandro Stawsky

HW5

1. (a) $p_n(x) = \sum_{k=0}^{n} l_k(x) f_k(x)$ → Lagrange form

(1) $p_n(x) = f[x_0, \dots, x_n](x-x_0)\cdots(x-x_n) + \dots + f[x_0]$ → Newton form

$$p_n(x) = \sum_{k=0}^{n} l_k(x) f_k(x) = \sum_{k=0}^{n} \left(\prod_{\substack{j=0 \\ j \neq k}}^{n} \frac{(x-x_k)}{(x_j - x_k)} \right) f_k(x)$$

$$\Rightarrow \sum_{k=0}^{n} \frac{x^n f_n(x)}{\prod_{\substack{j=0 \\ j \neq k}}^{n} (x_j - x_k)} \text{ is the highest order term}$$

(1) $\Rightarrow f[x_0, \dots, x_n](x-x_0)\cdots(x-x_n)$ is the highest order term

$$\Rightarrow f[x_0, \dots, x_n] = \sum_{k=0}^{n} \frac{f_n(x)}{\prod_{\substack{j=0 \\ j \neq k}}^{n} (x_k - x_j)} \quad \blacksquare$$

(b) As seen in (a) $f[x_0, \dots, x_n]$ is the coefficient of the highest order term. By the uniqueness of $p_n(x)$ which interpolates $f(x)$ at $x_0, \dots, x_n$, we get $f[x_0, \dots, x_n] = \sum_{k=0}^{n} \frac{f_n(x)}{\prod_{\substack{j=0 \\ j \neq k}}^{n} (x_k - x_j)}$, which is invariant to the order the nodes are introduced.

$\therefore$ divided differences are symmetric functions of their arguments. $\blacksquare$

```
% Math 104A, HW 5: Interpolation #2
% Alejandro Stawsky

% 2.a

% used example from the class notes

Nodes=[];
for j=0:3
    Nodes=[Nodes,j];
end

Nodes
```

```
Nodes =
     0     1     2     3
```

```
for i=1:length(Nodes) % all the zero divided differences
    g(Nodes(i))
end
```

```
ans = 1
ans = 2
ans = 5
ans = 10
```

```
[Coefficients,Approx]=NewtonDivDifCoeff(Nodes,@g,11)
```

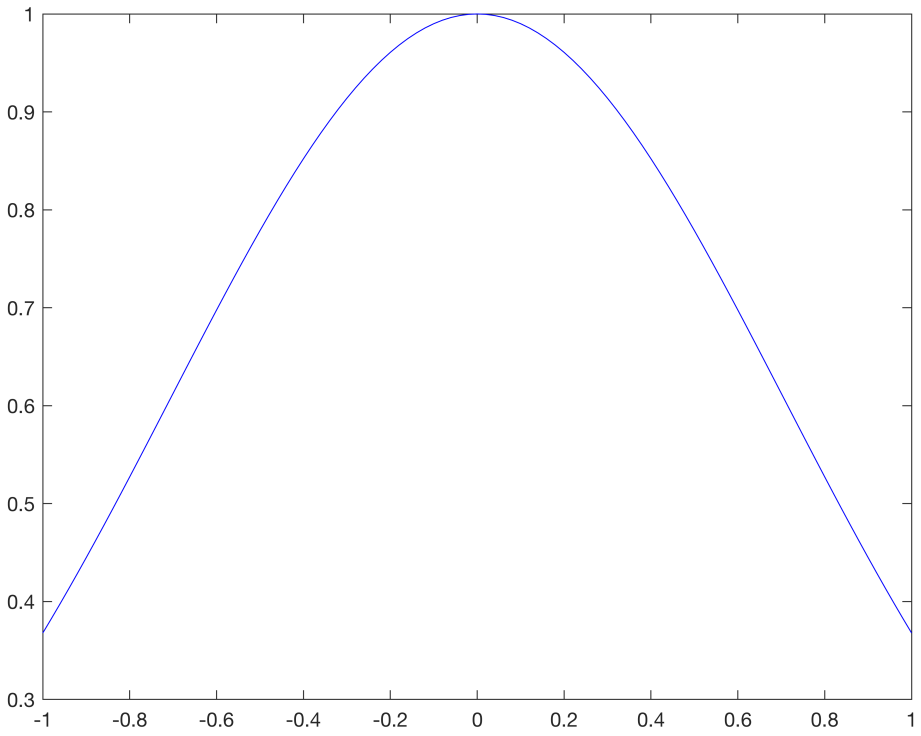
```
Coefficients =
     1     1     1     0
Approx = 122
```

```
% 2.b

Nodes=[]; % Defining the nodes
for j=0:10
    Nodes=[Nodes,-1+j*(2/10)];
end

x=[];
P=[];
[C,~]=NewtonDivDifCoeff(Nodes,@f,0); % Defining the coefficients
C
```

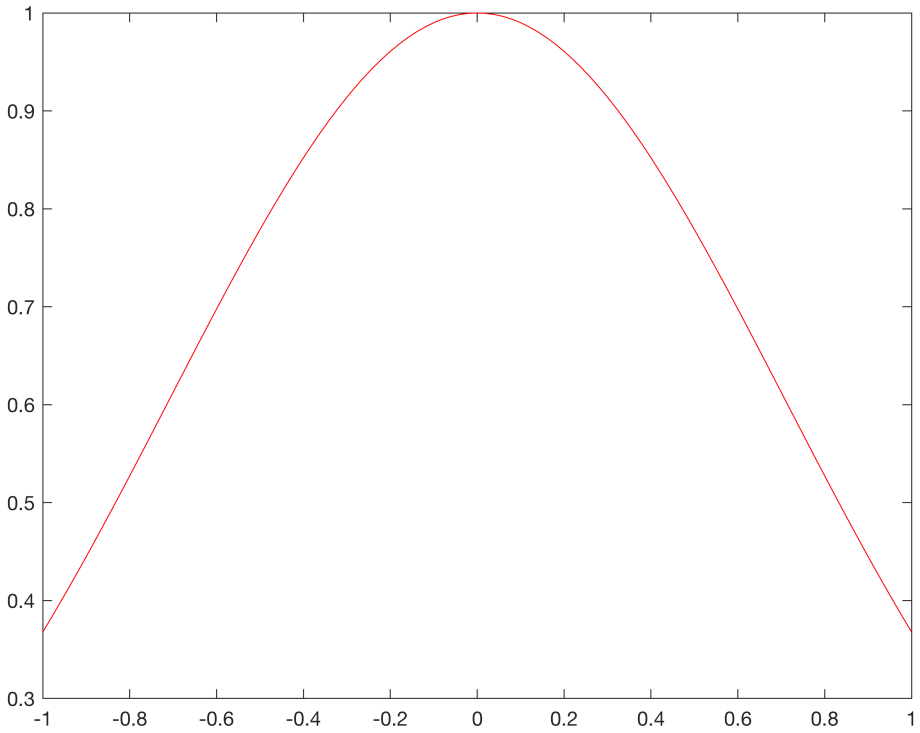
```
C =
    0.3679    0.7971    0.1371   -0.5602   -0.0786    0.2425   -0.0212   -0.0622 ···
```

```
for j=0:100 % Defining the polynomial approximations
    [~,p] = NewtonDivDifCoeff(Nodes,@f,-1+j*(1/50));
    P=[P,p];
```

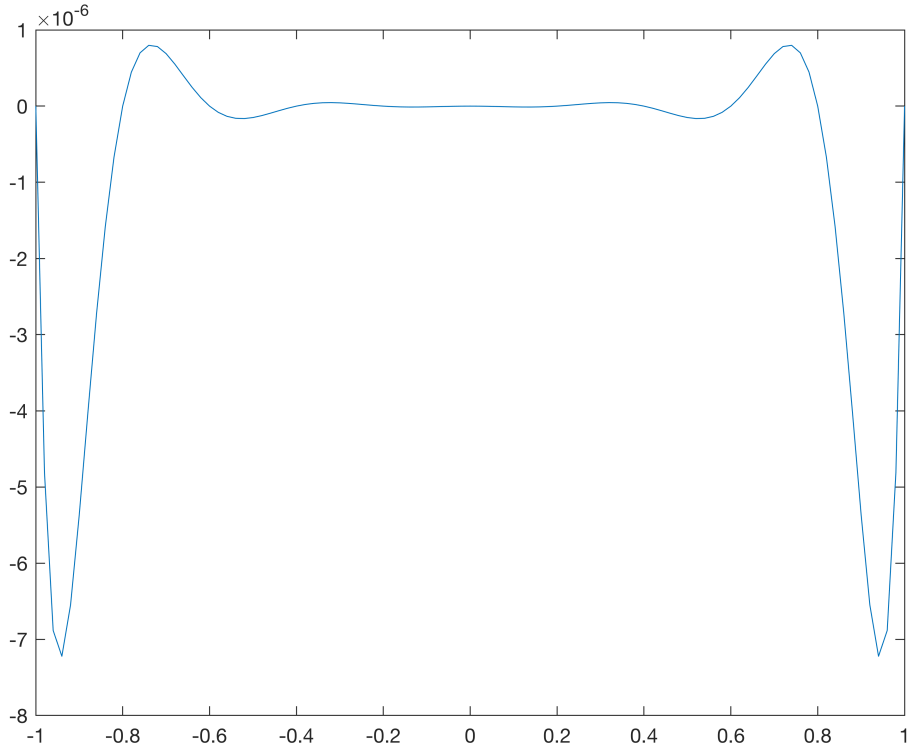
```
    x = [x,-1+j*(1/50)]; % Defining the evaluation points
end

plot(x,P,'b')% plotting the polynomial approximation
```




```
plot(x,f(x),'r') % plotting the actual function
```




```
plot(x,f(x)-P) % plotting the error
```




```
% 3

Nodes = [-0.106530659712633, 0.051188363905973]; % using f inverse so x's become y's
Values = [.5, .6];
p1 = Values(1)+Nodes(1).*((Values(1)-Values(2))./(Nodes(2)-Nodes(1))) % the value of p1(0)
```

```
p1 = 0.5675
```

```
function y = g(x)
    y = 1+x^2;
end

function y = f(x)
    y = exp(-(x).^2);
end

function [C,p] = NewtonDivDifCoeff(Nodes,f,x)
    % Nodes is the set of nodes, f is the function we are
      % approximating and x is the value we are approximating at
        % C is the array of coefficients and p is the polynomial

    C=[];
```

```
    for i=1:length(Nodes) % all the zero divided differences
        C=[C, f(Nodes(i))];
    end
    for k=2:length(Nodes) % Computing the set of coefficients of the int. poly.
        for j=length(Nodes):-1:k
            C(j)=(C(j)-C(j-1))/(Nodes(j)-Nodes(j-k+1));
        end
    end
    p = C(length(Nodes)); % computing the integration polynomial p
    for j=(length(Nodes)-1):-1:1
        p = C(j) + (x - Nodes(j))*p;
    end

end
```

4. Obtain the Hermite Polynomial corresponding to the data

$f(0)=0,\ f'(0)=0,\ f(1)=2,\ f'(1)=3.$

	1st D.D	2nd D.D.	3rd D.D	4th D.D.
0	$f(0)=0$			
0	$f(0)=0$	$f'(0)=0$		
1	$f(1)=2$	$f[0,1]=2$	$f[0,0,1]=2$	
1	$f(1)=2$	$f'(1)=3$	$f[0,1,1]=1$	$f[0,0,1,1]=-1$

$$p(x) = 0+0+2(x-0)^2+(-1)(x-0)^2(x-1)$$

$$= 2x^2-x^3+x^2 = \boxed{-x^3+3x^2 = p(x)}.$$