# Chapter 3. Interpolation and Polynomial Approximation

## 3.1 Interpolation and the Lagrange Polynomial

### Theorem 3.2

If $x_0, x_1, \cdots, x_n$ are $n+1$ distinct numbers and $f$ is a function whose values are given at these numbers, then a unique polynomial $P(x)$ of degree at most $n$ exists with

$$f(x_k) = P(x_k), \qquad \text{for each } k = 0, 1, \cdots, n.$$

This polynomial is given by

$$P(x) = f(x_0)L_{n,0}(x) + \cdots + f(x_n)L_{n,n}(x) = \sum_{k=0}^{n} f(x_k)L_{n,k}(x),$$

where, for each $k = 0, 1, \cdots, n$,

$$L_{n,k}(x) = \frac{(x-x_0)(x-x_1)\cdots(x-x_{k-1})(x-x_{k+1})\cdots(x-x_n)}{(x_k-x_0)(x_k-x_1)\cdots(x_k-x_{k-1})(x_k-x_{k+1})\cdots(x_k-x_n)} = \prod_{i=0,\ i\neq k}^{n} \frac{(x-x_i)}{(x_k-x_i)}.$$

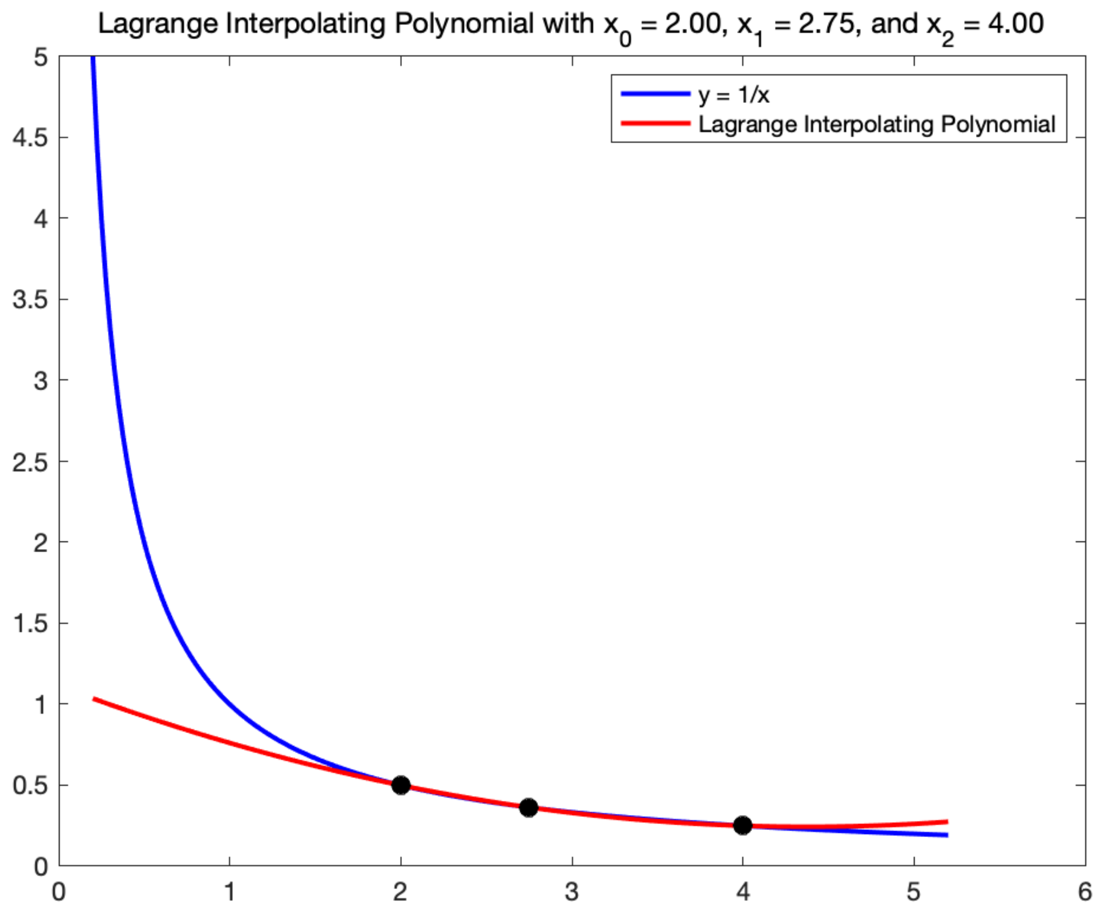
### Example 2

Use the numbers (called nodes) $x_0 = 2$, $x_1 = 2.75$, and $x_2 = 4$ to find the second Lagrange interpolating polynomial for $f(x) = 1/x$ and approximate $f(3) = 1/3$.

```
x = [2, 2.75, 4];
fval = 1./x;
t = 3;
val = Lagrange_polynomial(t,x,fval);
sprintf('%.5f',val)
```

```
ans = '0.32955'
```

```
t = linspace(0.2,5.2,200);
val = zeros(size(t));
for j = 1:length(val)
    val(j) = Lagrange_polynomial(t(j),x,fval);
end
figure()
plot(t,1./t,'b-',linewidth=2)
hold on
plot(t,val,'r-',linewidth=2)
plot(x,1./x,'ko',markersize=8,markerfacecolor='k')
legend('y = 1/x', 'Lagrange Interpolating Polynomial')
title(sprintf('Lagrange Interpolating Polynomial with x_0 = %.2f, x_1 = %.2f, and x_2 = %.2f',x(1),x(2),x(3)))
```




## 3.5 Cubic Spline Interpolation

### Natural Cubic Spline

**INPUT** $n$; $x_0, x_1, \cdots, x_n$; $a_0 = f(x_0), a_1 = f(x_1), \cdots, a_n = f(x_n)$.

**OUTPUT** $a_j, b_j, c_j, d_j$ for $j = 0, 1, \cdots, n-1$

(Note: $S(x) = S_j(x) = a_j + b_j(x - x_j) + c_j(x - x_j)^2 + d_j(x - x_j)^3$ for $x_j \le x \le x_{j+1}$)

**Step 1** For $i = 0, 1, \cdots, n-1$ set $h_i = x_{i+1} - x_i$.

**Step 2** For $i = 1, 2, \cdots, n-1$ set $\alpha_i = \dfrac{3}{h_i}(a_{i+1} - a_i) - \dfrac{3}{h_{i-1}}(a_i - a_{i-1})$.

**Step 3** Set $\ell_0 = 1$; $\mu_0 = 0$; $z_0 = 0$.

**Step 4** For $i = 1, 2, \cdots, n-1$

set $\ell_i = 2(x_{i+1} - x_{i-1}) - h_{i-1}\mu_{i-1}$; $\mu_i = h_i/\ell_i$; $z_i = (\alpha_i - h_{i-1}z_{i-1})/\ell_i$.

**Step 5** Set $\ell_n = 1$; $z_n = 0$; $c_n = 0$.

**Step 6** For $j = n-1, n-2, \cdots, 0$

set $c_j = z_j - \mu_j c_{j+1}$; $b_j = (a_{j+1} - a_j)/h_j - h_j(c_{j+1} + 2c_j)/3$; $d_j = (c_{j+1} - c_j)/(3h_j)$.

**Step 7** OUTPUT $(a_j, b_j, c_j, d_j$ for $j = 0, 1, \cdots, n-1)$;

STOP.

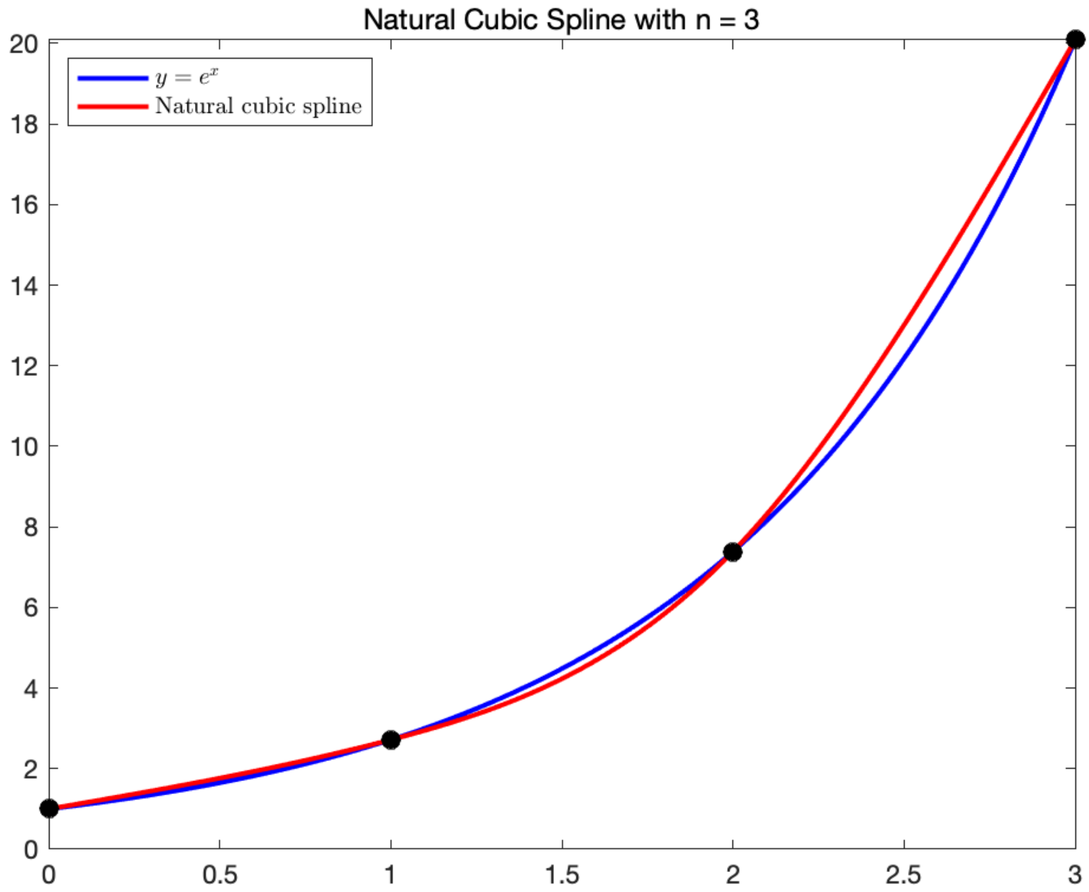
**Example 2**

Use the data points $(0, 1)$, $(1, e)$, $(2, e^2)$, and $(3, e^3)$ to form a natural spline $S(x)$ that approximates $f(x) = e^x$.

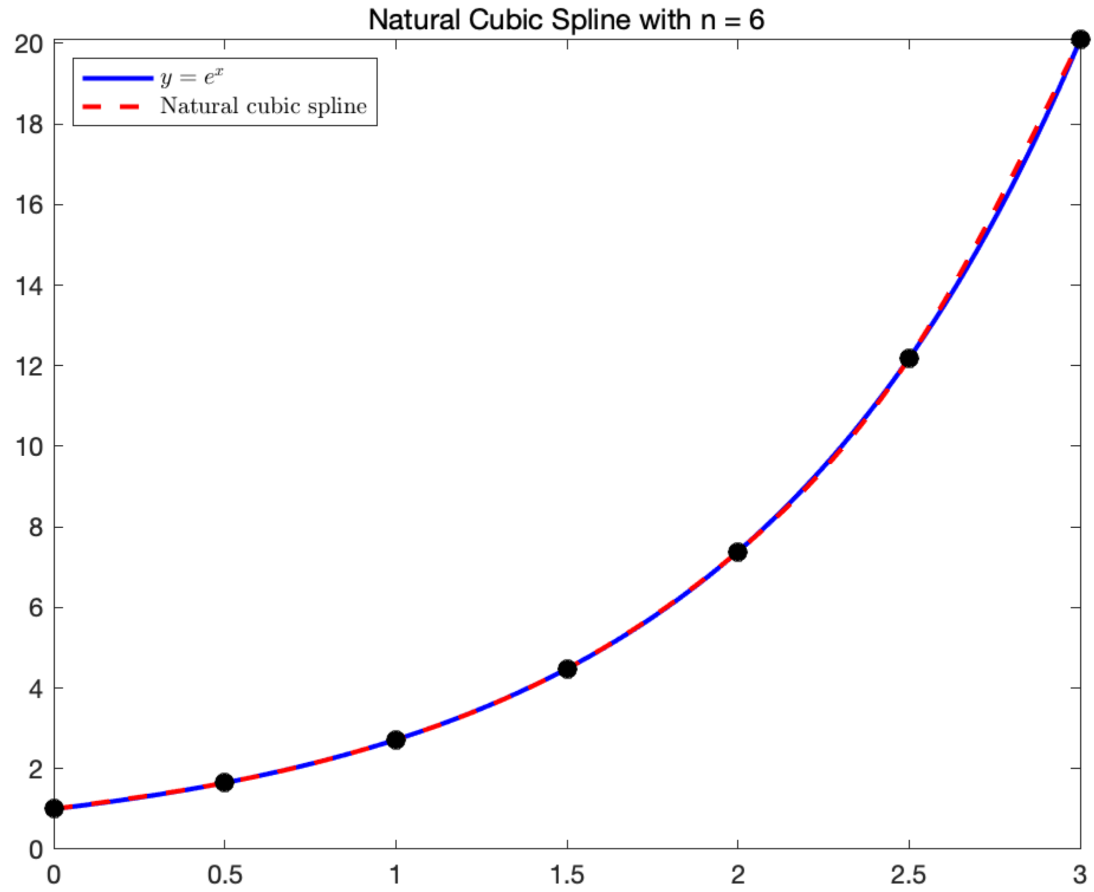
```
n = 3;
x = [0,1,2,3];
fval = exp(x);
[a,b,c,d] = natural_cubic_spline(n,x,fval);
fprintf(' a: [%s]\n b: [%s]\n c: [%s]\n d: [%s]', join(string(a), ','), join(string(b), ','), join(string(c), ','), join(string(d), ','));
```

```
a: [1,2.71828,7.38906,20.0855]
b: [1.466,2.2229,8.8098]
c: [0,0.75685,5.8301,0]
d: [0.25228,1.6911,-1.9434]
```

```
nodes = linspace(0,3);
fval = evaluate_cubic_spline(nodes,x,a,b,c,d);
figure()
plot(nodes,exp(nodes),'b-',linewidth=2);
hold on
plot(nodes,fval,'r-',linewidth=2)
plot(x,exp(x),'ko',markersize=8,markerfacecolor='k')
legend('$y = e^x$', 'Natural cubic spline','Interpreter','latex','location','northwest')
title(sprintf('Natural Cubic Spline with n = %d',n))
```




```
x = [0,0.5,1,1.5,2,2.5,3];
n = length(x)-1;
fval = exp(x);
[a,b,c,d] = natural_cubic_spline(n,x,fval);
nodes = linspace(0,3);
fval = evaluate_cubic_spline(nodes,x,a,b,c,d);
figure()
plot(nodes,exp(nodes),'b-',linewidth=2);
hold on
plot(nodes,fval,'r--',linewidth=2)
plot(x,exp(x),'ko',markersize=8,markerfacecolor='k')
legend('$y = e^x$', 'Natural cubic spline','Interpreter','latex','location','northwest')
title(sprintf('Natural Cubic Spline with n = %d',n))
```




**Clamped Cubic Spline**

**INPUT** $n$; $x_0,\ x_1,\ \cdots,\ x_n$; $a_0 = f(x_0),\ a_1 = f(x_1),\ \cdots,\ a_n = f(x_n)$; $FPO = f'(x_0)$; $FPN = f'(x_n)$.

**OUTPUT** $a_j,\ b_j,\ c_j,\ d_j$ for $j = 0,\ 1,\ \cdots,\ n-1$

(Note: $S(x) = S_j(x) = a_j + b_j(x - x_j) + c_j(x - x_j)^2 + d_j(x - x_j)^3$ for $x_j \le x \le x_{j+1}$)

**Step 1** For $i = 0,\ 1,\ \cdots,\ n-1$ set $h_i = x_{i+1} - x_i$.

**Step 2** Set $\alpha_0 = 3(a_1 - a_0)/h_0 - 3FPO$; $\quad \alpha_n = 3FPN - 3(a_n - a_{n-1})/h_{n-1}$

**Step 3** For $i = 1,\ 2,\ \cdots,\ n-1$ set $\alpha_i = \dfrac{3}{h_i}(a_{i+1} - a_i) - \dfrac{3}{h_{i-1}}(a_i - a_{i-1})$.

**Step 4** Set $\ell_0 = 2h_0$; $\quad \mu_0 = 0.5$; $\quad z_0 = \alpha_0/\ell_0$.

**Step 5** For $i = 1,\ 2,\ \cdots,\ n-1$

set $\ell_i = 2(x_{i+1} - x_{i-1}) - h_{i-1}\mu_{i-1}$; $\quad \mu_i = h_i/\ell_i$; $\quad z_i = (\alpha_i - h_{i-1}z_{i-1})/\ell_i$.

**Step 6** Set $\ell_n = h_{n-1}(2 - \mu_{n-1}$; $\quad z_n = (\alpha_n - h_{n-1}z_{n-1})/\ell_n$; $\quad c_n = z_n$.

**Step 7** For $j = n-1,\ n-2,\ \cdots,\ 0$

set $c_j = z_j - \mu_j c_{j+1}$; $\quad b_j = (a_{j+1} - a_j)/h_j - h_j(c_{j+1} + 2c_j)/3$; $\quad d_j = (c_{j+1} - c_j)/(3h_j)$.

**Step 8** OUTPUT $(a_j,\ b_j,\ c_j,\ d_j$ for $j = 0,\ 1,\ \cdots,\ n-1)$;

STOP.

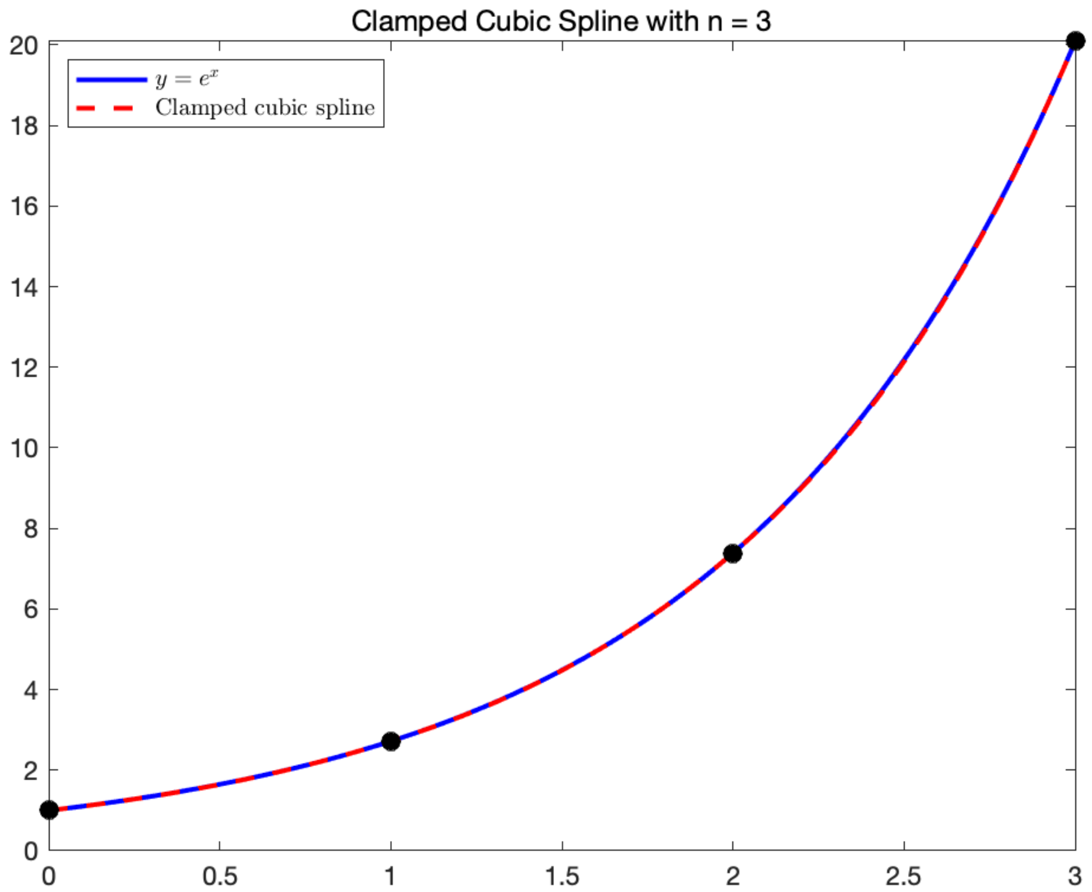
**Example 4**

Example 2 used a natural spline and the data points $(0,1)$, $(1,e)$, $(2,e^2)$, and $(3,e^3)$ to form a new approximating function $S(x)$. Determine the clamped spline $s(x)$ that uses this data and the additional information that, since $f'(x) = e^x$, so $f'(x) = 1$ and $f'(e) = e^3$.

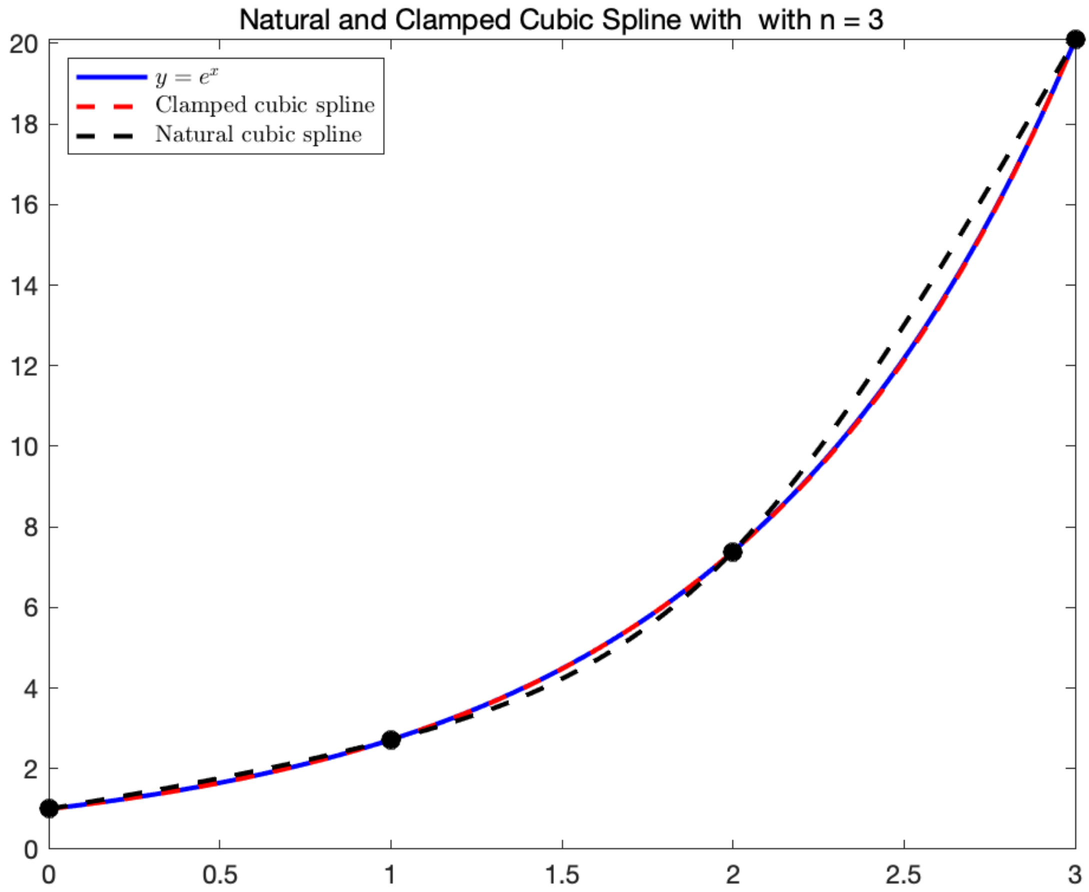
```
n = 3;
x = [0,1,2,3];
fval = exp(x);
FPO = 1;
FPN = exp(3);
[a,b,c,d] = clamped_cubic_spline(n,x,fval,FPO,FPN);
fprintf(' a: [%s]\n b: [%s]\n c: [%s]\n d: [%s]', join(string(a), ','), join(string(b), ','), join(string(c), ','), join(string(d), ','));
```

```
a: [1,2.71828,7.38906,20.0855]
b: [1,2.7102,7.3265]
c: [0.44468,1.2655,3.3509,9.4081]
d: [0.2736,0.69513,2.0191]
```

```
nodes = linspace(0,3);
fvalc = evaluate_cubic_spline(nodes,x,a,b,c,d);
figure()
plot(nodes,exp(nodes),'b-',linewidth=2);
hold on
plot(nodes,fvalc,'r--',linewidth=2)
plot(x,exp(x),'ko',markersize=8,markerfacecolor='k')
legend('$y = e^x$', 'Clamped cubic spline','Interpreter','latex','location','northwest')
title(sprintf('Clamped Cubic Spline with n = %d',n))
```




```
[an,bn,cn,dn] = natural_cubic_spline(n,x,fval);
fvaln = evaluate_cubic_spline(nodes,x,an,bn,cn,dn);
figure()
plot(nodes,exp(nodes),'b-',linewidth=2);
hold on
plot(nodes,fvalc,'r--',linewidth=2)
plot(nodes,fvaln,'k--',linewidth=2)
plot(x,exp(x),'ko',markersize=8,markerfacecolor='k')
legend('$y = e^x$', 'Clamped cubic spline', 'Natural cubic spline', 'Interpreter','latex','location','northwest')
title(sprintf('Natural and Clamped Cubic Spline with  with n = %d',n))
```



Natural and Clamped Cubic Spline with  with n = 3
$y = e^x$
Clamped cubic spline
Natural cubic spline
0
2
4
6
8
10
12
14
16
18
20
0
0.5
1
1.5
2
2.5
3