3.2 - Interpolation and Lagrange Polynomials

1. Polynomial Interpolation:

Problem: Given \( n + 1 \) pairs of data points \((x_i, y_i)\), where \( y_i = f(x_i), i = 0, 1, \ldots, n \) for some function \( f(x) \), we want to find a polynomial \( P_k(x) \) of lowest possible degree for which \( P_k(x_i) = y_i, \ i = 0, 1, \ldots, n \) (\( P_k(x) \) agrees with \( f(x) \) at \( x_0, \ldots, x_n \)).

The polynomial \( P_k(x) \) is said to **interpolate** the data \((x_i, y_i), i = 0, 1, \ldots, n \) and is called an **interpolating polynomial**. Graphically, \( P_k(x) \) is an approximation to \( f(x) \) and satisfies the conditions:

\[
P_k(x_i) = f(x_i), \ i = 0, 1, \ldots, n.
\]

For example,

\[
\begin{array}{c|ccccc}
\hline
x & -3.0 & -2.5 & -2.0 & -1.5 & -1.0 & -0.5 & 0.5 & 1.0 & 1.5 & 2.0 & 2.5 & 3.0 \\
\hline
y & -10 & -5 & 5 & 10 & & & & & & & & \\
\hline
\end{array}
\]

\[
y = f(x), \quad -\ -\ -\ y = P_k(x)
\]

Obviously, for this example \( P_k(x) \) is not a good approximation to \( f(x) \) though \( P_k(x) \) satisfies the conditions:

\[
P_k(x_i) = y_i \quad \text{for} \ i = 0, 1, 2.
\]

How to form \( P_k(x) \)?

There are several ways to form \( P_k(x) \). Let us start with a way we know already. Let

\[
P_k(x) = a_0 + a_1 x + a_2 x^2 + \ldots + a_k x^k.
\]

We want to find coefficients \( a_0, a_1, \ldots, a_k \) such that the following \( n + 1 \) equations hold:

\[
P_k(x_i) = a_0 + a_1 x + a_2 x^2 + \ldots + a_k x^k = y_i \quad \text{for} \ i = 0, 1, \ldots, n.
\]

Rewrite these \( n + 1 \) equations in a matrix-vector form:

\[
\begin{pmatrix}
1 & x_0 & x_0^2 & \cdots & x_0^k \\
1 & x_1 & x_1^2 & \cdots & x_1^k \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
1 & x_n & x_n^2 & \cdots & x_n^k
\end{pmatrix}
\begin{pmatrix}
a_0 \\
a_1 \\
a_2 \\
\vdots \\
a_k
\end{pmatrix}
=
\begin{pmatrix}
y_0 \\
y_1 \\
\vdots \\
y_n
\end{pmatrix}.
\]

The vector

\[
\begin{pmatrix}
a_0 \\
a_1 \\
a_2 \\
\vdots \\
a_k
\end{pmatrix}
\]

is a solution of the system (*) of \( n + 1 \) linear equations in \( k + 1 \) unknowns.
Consider the case where $k = n$. Recall in Linear Algebra, we learned that this system has a unique solution if the coefficient matrix

$$
\begin{bmatrix}
1 & x_0 & x_0^2 & \cdots & x_0^k \\
1 & x_1 & x_1^2 & \cdots & x_1^k \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
1 & x_n & x_n^2 & \cdots & x_n^k \\
\end{bmatrix}
$$

is nonsingular and the solution vector is of the form:

$$
\begin{bmatrix}
a_0 \\
a_1 \\
a_2 \\
a_k \\
\end{bmatrix} = 
\begin{bmatrix}
1 & x_0 & x_0^2 & \cdots & x_0^k \\
1 & x_1 & x_1^2 & \cdots & x_1^k \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
1 & x_n & x_n^2 & \cdots & x_n^k \\
\end{bmatrix}^{-1} \begin{bmatrix}
y_0 \\
y_1 \\
y_2 \\
y_n \\
\end{bmatrix}
$$

and can be solved by Gaussian-Elimination and Backward-Substitution.

**Example** Let $x_0 = 0$, $x_1 = 0.6$, $x_2 = 0.9$, $f(x) = \cos(x^2)$. Find $P_2(x) = a_0 + a_1 x + a_2 x^2$. Sketch both $y = f(x)$ and $y = P_2(x)$ for $x$ in $[0, 1]$.

Solve

$$
\begin{bmatrix}
a_0 \\
a_1 \\
a_2 \\
\end{bmatrix} = 
\begin{bmatrix}
1 & 0 & 0 \\
1 & 0.6 & 0.36 \\
1 & 0.9 & 0.81 \\
\end{bmatrix}^{-1} \begin{bmatrix}
a_0 \\
a_1 \\
a_2 \\
\end{bmatrix} = 
\begin{bmatrix}
1 & \cos(0.36) \\
\cos(0.36) & \cos(0.81) \\
\end{bmatrix} = 
\begin{bmatrix}
1.0 \\
0.369487601 & -0.793877047 \\
\end{bmatrix}.
$$

Hence, $P_2(x) = 1 + 0.369487601x - 0.793877047x^2$.

In MatLab, we can complete this example as follows.

```matlab
>> xv=[0;0.6;0.9];
>> A=[ones(3,1) xv xv.^2];
```
Now we sketch the graphs of $y = \cos(x^2)$ and $y = P_2(x) = 1 + 0.369487601x - 0.793877047x^2$ for $x$ in $[0, 1]$.

```matlab
>> xnew = 0:0.01:1;
>> y1 = cos(xnew.^2);
>> y2 = coeffv(1) + coeffv(2)*xnew + coeffv(3)*xnew.^2;
>> plot(xnew, y1, 'r-', xnew, y2, 'b-.');
```

If we want to add a pair of data $(0.3, \cos(0.09))$, then we can go over the following steps.

```matlab
>> x = [0; 0.3; 0.6; 0.9];
>> A = [ones(4,1) x x.^2 x.^3];
>> y = cos(x.^2);
>> coeffv = inv(A)*y
coeffv =
1.000000000000000
-0.064958529434456
0.412917759444563
-0.804529870654783
Hence, $P_3(x) = 1 - 0.064958529434456x + 0.412917759444563x^2 - 0.804529870654783x^3$. Now we sketch the graphs of $y = \cos(x^2)$ and $y = P_3(x)$ as follows. Remember that xnew and y1 are still there. We need to recompute y2.

```matlab
>> y2 = coeffv(1) + coeffv(2)*xnew + coeffv(3)*xnew.^2 + coeffv(4)*xnew.^3;
>> plot(xnew, y1, 'r-', xnew, y2, 'b-.');
>> title('red - y=cos(x^2), blue -. y=P_3(x) ')
```

3
Clearly, \( P_3(x) \) fits \( y = \cos(x^2) \) much better than \( P_2(x) \) does.

Note that the system (*) of linear equations are getting more difficult to solve when \( n \) is large. Can \( P_k(x) \) be formed with less computation?

2. Lagrange Interpolating Polynomials:
   a. Lagrange Polynomials:
      For \( k = 0, 1, \ldots, n \), define
      \[
      L_{n,k}(x) = \prod_{i=0, i \neq k}^{n} \frac{x-x_i}{x_k-x_i} = \frac{(x-x_0)\ldots(x-x_{k-1})(x-x_{k+1})\ldots(x-x_n)}{(x_k-x_0)\ldots(x_k-x_{k-1})(x_k-x_{k+1})\ldots(x_k-x_n)}.
      \]
      \( L_{n,k}(x) \) are called Lagrange polynomials. For example, let \( x_i = i, \ i = 0, 1, 2, 3. \) Then \( x_0 = 0, x_1 = 1, x_2 = 2 \) and \( x_3 = 3 \), and
      \[
      L_{3,1}(x) = \frac{(x-0)(x-2)(x-3)}{(1-0)(1-2)(1-3)} = \frac{1}{2}x(x-2)(x-3)
      \]
      \[
      L_{3,2}(x) = \frac{x(x-1)(x-3)}{(2-0)(2-1)(2-3)} = -\frac{1}{2}x(x-1)(x-3)
      \]
      Observe that Lagrange Polynomials have the following properties:
      i. \( L_{n,k}(x_k) = 1 \)
      \[
      L_{n,k}(x_k) = \frac{(x_k-x_0)\ldots(x_k-x_{k-1})(x_k-x_{k+1})\ldots(x_k-x_n)}{(x_k-x_0)\ldots(x_k-x_{k-1})(x_k-x_{k+1})\ldots(x_k-x_n)} = 1
      \]
      ii. \( L_{n,k}(x_i) = 0 \) for \( i \neq k \).
      \[
      L_{n,k}(x_i) = \frac{(x_i-x_0)\ldots(x_i-x_{i-1})(x_i-x_{i+1})\ldots(x_i-x_n)}{(x_k-x_0)\ldots(x_k-x_{k-1})(x_k-x_{k+1})\ldots(x_k-x_n)} = 0
      \]
      For example,
      \[
      L_{3,2}(2) = \frac{2(2-1)(2-3)}{(2-0)(2-1)(2-3)} = 1 \quad \text{and} \quad L_{3,2}(1) = \frac{1(1-1)(1-3)}{(2-0)(2-1)(2-3)} = 0.
      \]

b. Lagrange Interpolating Polynomials:
   The polynomial
\( P_n(x) = \sum_{k=0}^{n} y_k L_{n,k}(x) = y_0 L_{n,0}(x) + y_1 L_{n,1}(x) + \ldots + y_n L_{n,n}(x) \)

is called the \( n \)th Lagrange interpolating polynomial. Observe that for \( i = 0, 1, \ldots, n \)

\[ P_n(x_i) = \sum_{k=0}^{n} y_k L_{n,k}(x_i) = y_i L_{n,i}(x_i) = y_i. \]

\( P_n(x) \) is an \( n \)th polynomial that agrees with \( f(x) \) at \( x_0, x_1, \ldots, x_n \).

**Theorem** Suppose that \( x_0, \ldots, x_n \) are distinct numbers in the interval \([a, b]\) and \( f^{(n+1)} \) is continuous in \([a, b]\). For each \( x \) in \([a, b]\), there exists a number \( c(x) \) in \((a, b)\) such that

\[ f(x) = P_n(x) + \frac{f^{(n+1)}(c(x))}{(n+1)!} (x-x_0)(x-x_1)\ldots(x-x_n). \]

**Proof** For \( x \neq x_i \) and \( x \) in \([a, b]\), define \( h(t) = f(t) - P_n(t) - (f(x) - P_n(x)) \prod_{k=0}^{n} \frac{(t-x_k)}{x-x_k} \). Clearly, \( h(x_i) = 0 \), \( i = 0, 1, \ldots, n \). Observe that \( h(x) = 0 \). Since \( x \neq x_i \),

\[ h(t) = 0 \text{ at } t = x_0, x_1, \ldots, x_n, x. \]

Since \( h(t) \) has \( n + 2 \) distinct zeros in \([a, b]\), by Rolle’s Theorem, we know there exists a constant \( c \) in \((a, b)\) such that \( h^{(n+1)}(c) = 0 \).

\[
\begin{align*}
  h^{(n+1)}(t) &= f^{(n+1)}(t) - 0 - (f(x) - P_n(x)) \frac{(n+1)!}{(x-x_0)(x-x_1)\ldots(x-x_n)} \\
  h^{(n+1)}(c) &= f^{(n+1)}(c) - 0 - (f(x) - P_n(x)) \frac{(n+1)!}{(x-x_0)(x-x_1)\ldots(x-x_n)} = 0 \\
  R_n(x) &= f(x) - P_n(x) = \frac{f^{(n+1)}(c)}{(n+1)!} (x-x_0)(x-x_1)\ldots(x-x_n).
\end{align*}
\]

Recall in Calculus II, we studied \( n \)th degree Taylor polynomial at \( x = c \):

\[ P_{Taylor}(x) = f(c) + \frac{f'(c)}{1!}(x-c) + \frac{f''(c)}{2!}(x-c)^2 + \ldots + \frac{f^{(n)}(c)}{n!}(x-c)^n \]

and its error

\[ R_{Taylor}(x) = \frac{f^{(n+1)}(\beta)}{(n+1)!}(x-c)^{n+1} \text{ where } \beta \text{ is between } x \text{ and } c. \]

Note that the differences between a \( k \)th degree Taylor polynomial and a \( k \)th degree interpolating polynomial are:

a. \( P_{Taylor}(x) = f(x) \) at only \( x = x_0 \) and \( P_{interpolating}(x) = f(x) \), at \( x = x_0, x_1, \ldots, x_n \).

b. \( P_{Taylor} \) requires knowledge of \( f', f'', \ldots \) but \( P_{interpolating} \) requires \( f(x_0), f(x_1), \ldots, f(x_n) \). Recall in Calculus II, we studied \( n \)th degree Taylor polynomial at \( x = c \):

\[ P_{Taylor}(x) = f(c) + \frac{f'(c)}{1!}(x-c) + \frac{f''(c)}{2!}(x-c)^2 + \ldots + \frac{f^{(n)}(c)}{n!}(x-c)^n \]

and its error

\[ R_{Taylor}(x) = \frac{f^{(n+1)}(\beta)}{(n+1)!}(x-c)^{n+1} \text{ where } \beta \text{ is between } x \text{ and } c. \]

Note that the differences between a \( k \)th degree Taylor polynomial and a \( k \)th degree interpolating polynomial are:
polynomial are:

a. $P_{\text{Taylor}}(x) = f(x)$ at only $x = x_0$ and $P_{\text{interpolating}}(x) = f(x)$, at $x = x_0, x_1, \ldots, x_n$.

b. $P_{\text{Taylor}}$ requires knowledge of $f'$, $f''$, but $P_{\text{interpolating}}$ requires $f(x_0), f(x_1), \ldots, f(x_n)$.

Example Let $x_0 = 0$, $x_1 = 0.6$, $x_2 = 0.9$, $f(x) = \cos(x^2)$.

(1) Find the Lagrange interpolating polynomial $P_2(x)$ and $R_2(x)$.

(2) Use $P_2(x)$ to approximate $f(0.45)$ and estimate the approximation error.

(3) Approximate $\int_0^1 \cos(x^2) \, dx$ by $\int_0^1 P_2(x) \, dx$.

(1)

\[
P_2(x) = f(0) \frac{(x-0.6)(x-0.9)}{(-0.6)(-0.9)} + f(0.6) \frac{x(x-0.9)}{0.6(-0.3)} + f(0.9) \frac{x(x-0.6)}{0.9(0.3)}
\]

\[
P_2(x) = \frac{1}{0.54} (x-0.6)(x-0.9) - \frac{\cos(0.36)}{0.18} x(x-0.9) + \frac{\cos(0.81)}{0.27} x(x-0.6)
\]

\[\begin{align*}
\text{--- } y &= \cos(x^2), \quad \text{--- } y = P_2(x) \\
\text{\hspace{1cm}} f'(x) &= -2x \sin(x^2), \quad f''(x) = -2(\sin(x^2) + 2x^2 \cos(x^2)), \\
\text{\hspace{1cm}} f'''(x) &= -2(2x \cos(x^2) + 4x \sin(x^2) - 2x^3 \sin(x^2)) = -4(3x \cos(x^2) - 2x^3 \sin(x^2))
\end{align*}\]

\[
| R_2(x) | = \left| -\frac{4(3c \cos(c^2) - 2c^3 \sin(c^2))}{6} x(x-0.6)(x-0.9) \right|
\]

where $c$ is in $(0, 1)$.

(2)

\[
f(0.45) \approx P_2(0.45) = \frac{1}{0.54} (-0.15)(-0.45) - \frac{\cos(0.36)}{0.18} (0.45)(-0.45) + \frac{\cos(0.81)}{0.27} (0.45)(-0.15)
\]

\[
= 1.005509
\]

True error $\approx | \cos((0.45)^2) - P_2(0.45) | = | 0.9795668 - 1.005509 | = 0.0259422$

Find the approximation error:

\[
\left| f'''(x) \right| \leq \max_{x \in (0,1)} \left| -4(3x \cos(x^2) - 2x^3 \sin(x^2)) \right| \leq 4(3 + 2) = 20
\]
or check out the graph of \( |f'''(x)| \) to have a better estimate of an upper bound of \( |f'''(x)| \):

\[
y = |4(3x \cos(x^2) - 2x^3 \sin(x^2))|
\]

(3)

\[
\int_0^1 P_2(x)dx = \int_0^1 \left( \frac{1}{0.54} (x - 0.6)(x - 0.9) - \frac{\cos(0.36)}{0.18}x(x - 0.9) + \frac{\cos(0.81)}{0.27}x(x - 0.6) \right)dx
\]

\[
= \frac{1}{0.54} \int_0^1 ((x - 0.6)^2 - 0.3(x - 0.6))dx - \frac{\cos(0.36)}{0.18} \int_0^1 (x^2 - 0.9x)dx
\]

\[
+ \frac{\cos(0.81)}{0.27} \int_0^1 (x^2 - 0.6x)dx
\]

\[
= \frac{1}{0.54} (0.123333333) - \frac{\cos(0.36)}{0.18} (-0.116666667) + \frac{\cos(0.81)}{0.27} (3.3333333 \times 10^{-2})
\]

\[
= 0.920118119
\]

3. Neville’s Iterated Interpolation:

If we need to add more points \( x_{n+1}, \ldots \) from \( P_n(x) \) to construct an interpolation polynomial with larger degree, can we construct an interpolation polynomial iteratively? The answer is yes. The method is called the Neville Method.

Let \( m_1, m_2, \ldots, m_k \) be \( k \) distinct integers where \( 0 \leq m_i \leq n \) for each \( i \). Let \( P_{m_1, m_2, \ldots, m_k}(x) \) be the Lagrange polynomial that agrees with \( f \) at the \( k \) points: \( x_{m_1}, x_{m_2}, \ldots, x_{m_k} \). For example,

\[
P_{1,2,4}(x) = f(x_1) \frac{(x - x_2)(x - x_4)}{(x_1 - x_2)(x_1 - x_4)} + f(x_2) \frac{(x - x_1)(x - x_4)}{(x_2 - x_1)(x_2 - x_4)} + f(x_4) \frac{(x - x_1)(x - x_2)}{(x_4 - x_1)(x_4 - x_2)}
\]

is the Lagrange polynomial that agrees with \( f(x) \) at 3 points: \( x_1, x_2, x_4 \):

\[
P_{1,2,4}(x_1) = f(x_1), \quad P_{1,2,4}(x_2) = f(x_2), \quad P_{1,2,4}(x_3) = f(x_3).
\]

Theorem Let \( f \) be defined at \( x_0, x_1, \ldots, x_k \), and \( x_j \) and \( x_k \) be two distinct numbers. Then

\[
P_{0,1,\ldots,k}(x) = \frac{(x - x_j)P_{0,1,\ldots,j-1,j+1,\ldots,k}(x) - (x - x_i)P_{0,1,\ldots,i-1,i+1,\ldots,k}(x)}{x_i - x_j}.
\]

Proof We know that both polynomials \( P_{0,1,\ldots,j-1,j+1,\ldots,k}(x) \) and \( P_{0,1,\ldots,i-1,i+1,\ldots,k}(x) \) are degree \( k - 1 \). So, the degree of \( P_{0,1,\ldots,k}(x) \) is \( k \). Check if \( P_{0,1,\ldots,k}(x_i) = f(x_i) \) for \( l = 0, 1, \ldots, k \) since

\[
P_{0,1,\ldots,j-1,j+1,\ldots,k}(x_i) = f(x_i) \quad \text{if} \ l \neq j \quad \text{and} \quad P_{0,1,\ldots,i-1,i+1,\ldots,k}(x_i) = f(x_i) \quad \text{if} \ l \neq i
\]

For \( l = 0, 1, \ldots, k \), but \( l \neq i \) and \( l \neq j \),
\[ P_{0,1,\ldots,k}(x_l) = \frac{(x_l - x_j)P_{0,1,\ldots,j-1,j+1,\ldots,k}(x_l) - (x_l - x_i)P_{0,1,\ldots,i-1,i+1,\ldots,k}(x_l)}{x_l - x_j} \]

For \( l = i \):
\[ P_{0,1,\ldots,k}(x_i) = \frac{(x_i - x_j)P_{0,1,\ldots,j-1,j+1,\ldots,k}(x_i) - (x_i - x_i)P_{0,1,\ldots,i-1,i+1,\ldots,k}(x_i)}{x_i - x_j} = P_{0,1,\ldots,j-1,j+1,\ldots,k}(x_i) = f(x_i) \]

For \( l = j \):
\[ P_{0,1,\ldots,k}(x_j) = \frac{(x_j - x_i)P_{0,1,\ldots,j-1,j+1,\ldots,k}(x_j) - (x_j - x_i)P_{0,1,\ldots,i-1,i+1,\ldots,k}(x_j)}{x_i - x_j} = P_{0,1,\ldots,j-1,j+1,\ldots,k}(x_j) = f(x_j) \]

So, \( P_{0,1,\ldots,k}(x) \) the \( k \)th degree interpolating polynomial.

**Example** Consider \( x_0, x_1, \ldots, x_n \). Then we can form
\[ P_{1,2}(x) = f(x_1) \frac{(x - x_2)}{(x_1 - x_2)} + f(x_2) \frac{(x - x_1)}{(x_2 - x_1)} \text{ and } P_{2,4}(x) = f(x_2) \frac{(x - x_4)}{(x_1 - x_4)} + f(x_4) \frac{(x - x_4)}{(x_2 - x_4)} \]
using \( x_1, x_2 \) and \( x_4 \). Then we can combine \( P_{1,2}(x) \) and \( P_{2,4}(x) \) to form \( P_{1,2,4}(x) \):
\[ P_{1,2,4}(x) = \frac{(x - x_4)P_{1,2}(x) - (x - x_1)P_{2,4}(x)}{(x_1 - x_4)} \]
which is the Lagrange polynomial that agrees with \( f(x) \) at 3 points: \( x_1, x_2, x_4 \).

Now given \( (0, 1), (1, e^{-1}), (2, e^{-4}), \) and \( (3, e^{-9}) \), find \( P_{0,1,2,3}(x) \) using \( P_{0,1,2}(x) \) and \( P_{1,2,3}(x) \).

\[ P_{0,1,2}(x) = \frac{(x - 1)(x - 2)}{(0 - 1)(0 - 2)} + e^{-1} \frac{(x - 0)(x - 2)}{(1 - 0)(1 - 2)} + e^{-4} \frac{(x - 0)(x - 1)}{(2 - 0)(2 - 1)} \]
\[ = \frac{1}{2}(x - 1)(x - 2) - e^{-1}x(x - 2) + \frac{e^{-4}}{2}x(x - 1) \]

\[ P_{1,2,3}(x) = e^{-1} \frac{(x - 2)(x - 3)}{(1 - 2)(1 - 3)} + e^{-4} \frac{(x - 1)(x - 3)}{(2 - 1)(2 - 3)} + e^{-9} \frac{(x - 1)(x - 2)}{(3 - 1)(3 - 2)} \]
\[ = \frac{1}{2}e^{-1}(x - 2)(x - 3) - e^{-4}(x - 1)(x - 3) + \frac{e^{-9}}{2}(x - 1)(x - 2) \]
\[ P_{0,1,2,3}(x) = \frac{(x - 0)P_{1,2,3}(x) - (x - 3)P_{0,1,2}(x)}{(3 - 0)} = \frac{1}{2}(xP_{1,2,3}(x) - (x - 3)P_{0,1,2}(x)) \]

**Algorithm** *Neville’s Iterated Interpolation*:  **Steps of Computing** \( P_{0,1,2,\ldots,k}(\bar{x}) \) for a given \( \bar{x} : \) *(neville.m)*

Given \( (x_i, f(x_i)) \) for \( i = 0, 1, \ldots, k \) and \( \epsilon \), let \( P_i = f(x_i) \)
\[ x_i \quad P_i(\bar{x}) \quad P_{i,i+1}(\bar{x}) \quad P_{i,i+1,i+2}(\bar{x}) \quad P_{i,i+1,i+2,i+3}(\bar{x}) \quad P_{i,i+1,i+2,i+3,i+4}(\bar{x}) \]

\[
\begin{array}{cccccc}
  x_0 & P_0 & & & & \\
  x_1 & P_1 & P_{0,1}(\bar{x}) & & & \\
  x_2 & P_2 & P_{1,2}(\bar{x}) & P_{0,1,2}(\bar{x}) & & \\
  x_3 & P_3 & P_{2,3}(\bar{x}) & P_{1,2,3}(\bar{x}) & P_{0,1,2,3}(\bar{x}) & \\
  x_4 & P_4 & P_{3,4}(\bar{x}) & P_{2,3,4}(\bar{x}) & P_{1,2,3,4}(\bar{x}) & P_{0,1,2,3,4}(\bar{x}) \\
  \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
\end{array}
\]

The algorithm stops and \( f(\bar{x}) \approx P_{0,1,2,\ldots,k}(\bar{x}) \) whenever
\[
| P_{0,1,2,\ldots,k}(\bar{x}) - P_{0,1,2,\ldots,k-1}(\bar{x}) | < \varepsilon.
\]

An alternative way:

\[
\begin{array}{cccccc}
  x_i & P_i(\bar{x}) & P_{i,i+1}(\bar{x}) & P_{i,i+1,i+2}(\bar{x}) & P_{i,i+1,i+2,i+3}(\bar{x}) & P_{i,i+1,i+2,i+3,i+4}(\bar{x}) \\
  x_0 & P_0 & & & & \\
  x_1 & P_1 & P_{0,1}(\bar{x}) & & & \\
  x_2 & P_2 & P_{0,2}(\bar{x}) & P_{0,1,2}(\bar{x}) & & \\
  x_3 & P_3 & P_{0,3}(\bar{x}) & P_{0,2,3}(\bar{x}) & P_{0,1,2,3}(\bar{x}) & \\
  x_4 & P_4 & P_{0,4}(\bar{x}) & P_{0,3,4}(\bar{x}) & P_{0,2,3,4}(\bar{x}) & P_{0,1,2,3,4}(\bar{x}) \\
  \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
\end{array}
\]

**Example** Suppose that \( x_j = j \) for \( j = 0, 1, 2, 3 \) and it is known that
\[
P_{0,1}(x) = x + 1, \quad P_{1,2}(x) = 3x - 1, \quad \text{and} \quad P_{1,2,3}(1.5) = 4.
\]

Find \( P_{1,2}(1.5), P_{0,1,2}(x) \) and \( P_{0,1,2,3}(1.5) \).

\[
P_{0,1,2}(x) = \frac{(x-x_2)P_{0,1}(x) - (x-x_0)P_{1,2}(x)}{x_0-x_2} = \frac{(x-2)(x+1) - x(3x-1)}{-2} = x^2 + 1
\]

\[
\begin{array}{cccc}
  x_i & P_i(\bar{x}) & P_{i,i+1}(\bar{x}) & P_{i,i+1,i+2}(\bar{x}) \\
  0 & P_0 & & \\
  1 & P_1 & P_{0,1}(1.5) = 2.5 & \\
  2 & P_2 & P_{1,2}(1.5) = 3.5 & P_{0,1,2}(1.5) = 3.25 \\
  3 & P_3 & P_{2,3}(1.5) & P_{1,2,3}(1.5) = 4 & P_{0,1,2,3}(1.5) = 5.4375 \\
\end{array}
\]

\[
P_{1,2,3}(x) = \frac{(x-x_1)P_{2,3}(x) - (x-x_3)P_{1,2}(x)}{x_3-x_1} = \frac{(x-1)P_{2,3}(x) - (x-3)(3x-1)}{3-1} = 4
\]

\[
P_{2,3}(1.5) = \frac{8 + (1.5 - 3)(3.5)}{1.5 - 1} = 5.5
\]

\[
P_{0,1,2}(1.5) = \frac{(1.5 - 2)P_{0,1}(1.5) - (1.5 - 0)P_{1,2}(1.5)}{0-2} = \frac{(-0.5)(2.5) - (1.5)(3.5)}{-2} = 3.25
\]
Example  Neville’s method is used to approximate \( f(0.4) \) as follows. Complete the table.

\[
\begin{array}{|c|c|c|c|c|}
\hline
x_i & P_i(x) & P_{i,i+1}(\tilde{x}) & P_{i,i+1,i+2}(\tilde{x}) & P_{i,i+1,i+2,i+3}(\tilde{x}) \\
\hline
0 & 1 & & & \\
0.25 & 2 & P_{0,1}(0.4) = 2.6 & & \\
0.5 & P_2 & P_{1,2}(0.4) & P_{0,1,2}(0.4) & \\
0.75 & 8 & P_{2,3}(0.4) = 2.4 & P_{1,2,3}(0.4) = 2.96 & P_{0,1,2,3}(0.4) = 3.016 \\
\hline
\end{array}
\]

\[
P_{2,3}(0.4) = \frac{(0.4-0.75)P_2 - (0.4-0.5)P_3}{0.5-0.75} = \frac{(2.4)(-0.25) - 0.1(8)}{-0.35} = 4
\]

\[
P_{1,2}(0.4) = \frac{(0.4-0.5)P_1 - (0.4-0.25)P_2}{0.5-0.5} = \frac{(2.4)(-0.25) - (0.4-0.25)(4)}{0.5-0.5} = 3.2
\]

\[
P_{0,1,2}(0.4) = \frac{(0.4-0.5)P_{0,1}(0.4) - (0.4-0.0)P_{1,2}(0.4)}{0-0.5} = \frac{(2.4)(-0.25) - (0.4-0.0)(3.2)}{0-0.5} = 3.08
\]

Check:

\[
P_{0,1,2,3}(0.4) = \frac{(0.4-0.75)(3.08) - (0.4-0.0)(2.96)}{0-0.75} = 3.016
\]

Example  Let \( f(x) = \sin(\ln x) \), \( x_0 = 2.0 \), \( x_1 = 2.4 \), \( x_2 = 2.6 \).

1. Estimate the approximation error \( R_2(x) \) of an interpolating polynomial \( P_2(x) \) over \([2.0, 2.6]\).

2. Approximate \( f(2.2) \) and \( f(2.5) \), and estimate their errors.

(1)

\[
| R_2(x) | = \left| \frac{f'''(c)}{3!}(x-2)(x-2.4)(x-2.6) \right|, \text{ where } c \text{ is in } [2, 2.6]
\]

\[
f'(x) = \frac{\cos(\ln x)}{x}, \quad f''(x) = \frac{-\sin(\ln x)(\frac{1}{x})(x) - \cos(\ln x)}{x^2} = \frac{-\sin(\ln x) - \cos(\ln x)}{x^2}
\]

\[
f'''(x) = \frac{[-\cos(\ln x)(\frac{1}{x}) + \sin(\ln x)(\frac{1}{x})]x^2 + 2x[\sin(\ln x) + \cos(\ln x)]}{x^3}
\]

\[
= \frac{[-\cos(\ln x) + \sin(\ln x)] + 2[\sin(\ln x) + \cos(\ln x)]}{x^3} = \frac{3\sin(\ln x) + \cos(\ln x)}{x^3}
\]
\[ |f'''(x)| \leq |f'''(2)| = \frac{3\sin(\ln 2) + \cos(\ln 2)}{2^3} = 0.336 \]

\[ |R_2(x)| = \left| \frac{f'''(c)}{3!} (x-2)(x-2.4)(x-2.6) \right| \leq \frac{0.336}{6} (0.02) = 0.00112 \]

(2) >> xv=[2;2.4;2.6];
>> yv=sin(log(xv));
>> [yout,yall]=neville(xv,yv,2.1,2)
yout = 0.67510095866951
yall =
0.63896127631363 0.67118192650420 0.67510095866951
0.76784387707588 0.69469611949605 0
0.8166094879577 0 0
>> [yout,yall]=neville(xv,yv,2.5,2)
yout = 0.79353280699093
yall =
0.63896127631363 0.80006452726644 0.79353280699093
0.76784387707588 0.7922646293583 0
0.8166094879577 0 0

true error: \(|\sin(\log(2.1)) - 0.67510095866951| = 6.1635483 \times 10^{-4} \]
true error: \(|\sin(\log(2.5)) - 0.79353280699093| = 1.83804402 \times 10^{-4} \]

\[ |R_2(2.1)| = \left| \frac{f'''(c)}{3!} (2.1-2)(2.1-2.4)(2.1-2.6) \right| = \frac{f'''(c)}{6} |(2.1-2)(2.1-2.4)(2.1-2.6)| \]
\[ \leq \frac{0.336}{6} (0.015) = 0.00084 \]

\[ |R_2(2.5)| = \left| \frac{f'''(c)}{3!} (2.5-2)(2.5-2.4)(2.5-2.6) \right| = \frac{f'''(c)}{6} |(2.5-2)(2.5-2.4)(2.5-2.6)| \]
\[ \leq \frac{0.336}{6} (0.005) = 0.00028 \]
Exercises:

1. Given \((x_i, y_i) = \{(0, -1), \left(\frac{1}{2}, 1\right), (1, 2)\}\) where \(y_i = f(x_i)\) for some \(f(x)\), construct the polynomial \(P_2(x)\) that agrees with \(f(x)\) at \(x_0\), \(x_1\) and \(x_2\) in the following two ways.
   (1) \(P_2(x) = a_0 + a_1x + a_2x^2\) a standard form.
   (2) \(P_2(x)\) is a Lagrange interpolating polynomial.

2. Let \(x_0 = -1\), \(x_1 = 0\), \(x_2 = 1\), and \(f(x) = e^x\).
   (1) Find algebraically (keep \(e\) in \(P_2(x)\)) the Lagrange interpolating polynomial \(P_2(x)\) and its approximation error \(R_2(x)\).
   (2) Use \(P_2(x)\) to approximate \(f\left(\frac{1}{2}\right)\) and \(f\left(-\frac{1}{3}\right)\).
   (3) Compute the true approximation errors \(|f\left(\frac{1}{2}\right) - P_2\left(\frac{1}{2}\right)|\) and \(|f\left(-\frac{1}{3}\right) - P_2\left(-\frac{1}{3}\right)|\).
   (4) Estimate the approximation errors using \(|R_2\left(\frac{1}{2}\right)|\) and \(|R_2\left(-\frac{1}{3}\right)|\).
   (5) Approximate \(\int_{-1}^{1} e^x \, dx\) by \(\int_{-1}^{1} P_2(x) \, dx\) and find the true approximation error \(\left|\int_{-1}^{1} e^x \, dx - \int_{0}^{1} P_2(x) \, dx\right|\).

3. Let \(x_0 = 0\), \(x_1 = 0.6\), \(x_2 = 0.9\), and \(f(x) = \sqrt{1 + x}\).
   (1) Find the Lagrange interpolating polynomial \(P_2(x)\) and its approximation error \(R_2(x)\).
   (2) Use \(P_2(x)\) to approximate \(f(0.45)\) and \(|R_2(0.45)|\) to estimate the approximation error.

4. Use Neville’s method to approximate \(f(0.5)\), giving the following table. Determine \(P_2 = f(0.7)\).

   \[
   \begin{array}{|c|c|c|c|}
   \hline
   i & x_i & P_i & P_{i,i+1} \, \, P_{i,i+1,i+2} \\
   \hline
   0 & 0 & 0 & \\
   1 & 0.4 & 2.8 & 3.5 \\
   2 & 0.7 & \frac{27}{7} & \\
   \hline
   \end{array}
   \]

5. Suppose we know that \(f(0) = 1, f(0.25) = 1.64872, f(0.5) = 2.71828, f(0.75) = 4.48169\). Use given data to approximate \(f(0.43)\) by a Lagrange interpolating polynomial using Neville’s method (MatLab program neville.m).

6. A census of the population of the US is taken every 10 years. The population, in thousands, from 1940 to 2000 is given below.

   \[
   \begin{array}{|c|c|c|c|c|c|c|}
   \hline
   \hline
   \text{t} & 0 & 10 & 20 & 30 & 40 & 50 & 60 \\
   \hline
   \text{population} & 132165 & 151326 & 179323 & 203302 & 226542 & 249633 & 251432 \\
   \hline
   \end{array}
   \]

   Use Neville’s method (MatLab program neville.m) to estimate the populations in 1975 and 2005 and predict the population in 2010.

7. Let \(f(x) = 2^x\). Approximate \(f(0.5) = \sqrt{2}\) by \(P(0.5)\) where \(P(x)\) is a Lagrangian interpolating polynomial. Suppose that Neville’s Method is used to generate \(P(0.5)\) iteratively as follows.
Let the stopping criterion be set as $|P_{01\ldots k}(0.5) - P_{01\ldots (k-1)}(0.5)| < \epsilon$.

a. Determine the degree ($k$) of the polynomial used to approximate $\sqrt{2}$ with $\epsilon = 0.0001$.

b. If $P_{0,1,2,3,4,5,6}(0.5)$ is chosen from above sequence to approximate $\sqrt{2}$, what is $\epsilon$?