4. Interpolation

Given a set of data points \((x_i, f_i), \ i = 0, 1, ..., n\), our concern here is to find a function \(g\) that interpolates these points, i.e., \(g(x_i) = f_i, \ i = 0, 1, ..., n\). \(g\) is called an interpolating function. The following figure shows an interpolating function that interpolates 8 points.

The need to construct an interpolating function occurs in several important applications. One application is to predict the behavior of an unknown function. For instance, if the values of a function \(f\) at \(x = x_0, x_1, ..., x_n\) are given, but \(f\) is unknown, then how should we estimate the value of \(f\) at an intermediate point? One approach is to construct a function \(g\) that interpolates \(f\) at \(x = x_0, x_1, ..., x_n\) (i.e., \(g(x_i) = f(x_i)\)), and use \(g\) to approximate the behavior of \(f\) at the intermediate point.

Another application is to compute the integral of a given function \(f(x)\).

\[
\int_{a}^{b} f(x) \, dx \tag{4.1}
\]

When the given function \(f(x)\) is simple, finding the integral (4.1) usually is not a problem. For instance, we know that

\[
\int_{a}^{b} \sin(x) \, dx = -\cos(x) \bigg|_{a}^{b}
\]

But what if \(f(x)\) is a complicated function, such as \(e^{-x^2}\)? The integral of this function has no closed form.

\[
\int_{a}^{b} e^{-x^2} \, dx \tag{4.2}
\]
In this case, one may only estimate the integral using some approximation methods. One such method is to find a function \( g \) of which we know how to find its integral and which agrees with \( e^{-x^2} \) at \( n+1 \) points

\[
g(x_i) = e^{-x^2_i}, \quad i = 0, 1, ..., n
\]

and then use the integral of \( g(x) \) over the interval \([a, b]\) to approximate (4.2).

Both approaches are based on the assumption that \( g \) would be close to \( f \) if \( g \) interpolates \( f \) at enough points. Three things have to be studied when constructing an interpolating function:

1. The cost of constructing \( g \).
2. The cost of evaluating \( g \) (to predict the behavior of \( f \)).
3. The accuracy problem (how close is \( g \) to \( f \)?).

To minimize the cost of the first two processes, \( g \) is usually chosen from the following classes of functions:

1. The set of polynomials of degree \( n \) or less.
2. The set of spline functions (piecewise continuous polynomials).

The second set of functions, in general, provides better accuracy.

**4.1 Polynomial Interpolation**

The concern here is, for a set of given points \((x_i, f_i), i = 0, 1, ..., n \) (\( x_i \neq x_j \) if \( i \neq j \)), find a polynomial \( p(x) \) such that \( p(x_i) = f_i, \quad i = 0, 1, ..., n \). It can be shown that such a polynomial \( p(x) \) always exists and its degree can be as low as \( n \). For this reason, \( p(x) \) is also denoted \( p_n(x) \).

For instance, for the following given points:

\[
(x_0, f_0), \quad (x_1, f_1), \quad (x_2, f_2)
\]

the existence of such a \( p(x) \) can be shown as follows. First, we set \( p(x) \) as follows

\[
p(x) = a_0 + a_1x + a_2x^2
\]

with \( a_0, \ a_1, \) and \( a_2 \) to be determined. Since \( p(x) \) must satisfy the condition: \( p(x_i) = f_i, \quad i = 0, 1, 2 \), we have

\[
\begin{align*}
a_0 + a_1x_0 + a_2x_0^2 &= f_0 \\
a_0 + a_1x_1 + a_2x_1^2 &= f_1 \\
a_0 + a_1x_2 + a_2x_2^2 &= f_2
\end{align*}
\]
or
\[
\begin{bmatrix}
1 & x_0 & x_0^2 \\
1 & x_1 & x_1^2 \\
1 & x_2 & x_2^2
\end{bmatrix}
\begin{bmatrix}
a_0 \\
a_1 \\
a_2
\end{bmatrix}
= 
\begin{bmatrix}
f_0 \\
f_1 \\
f_2
\end{bmatrix}
\tag{4.3}
\]

The coefficient matrix in the above equation is called a Vander Monte matrix. The determinant of this matrix is non-zero.

\[
\text{det}
\begin{bmatrix}
1 & x_0 & x_0^2 \\
1 & x_1 & x_1^2 \\
1 & x_2 & x_2^2
\end{bmatrix}
= (x_2 - x_1)(x_2 - x_0)(x_1 - x_0) \neq 0
\]

Hence, (4.3) has a unique solution.

\[
\begin{bmatrix}
a_0 \\
a_1 \\
a_2
\end{bmatrix}
= 
\begin{bmatrix}
1 & x_0 & x_0^2 \\
1 & x_1 & x_1^2 \\
1 & x_2 & x_2^2
\end{bmatrix}^{-1}
\begin{bmatrix}
f_0 \\
f_1 \\
f_2
\end{bmatrix}
\]

This shows that \( p(x) \) exists and its degree is 2. The degree of this polynomial can not be smaller than 2 (Why?).

Computationally, however, this is not a good approach to construct \( p(x) \) since the Vander Monte matrix is very ill-conditioned. If \( n+1 \) points are given, this approach needs \( O(n^3) \) operations to find all the coefficients of \( p(x) \). Usually, Lagrange form or Newton form are used in the construction of \( p(x) \).

### 4.1.1 Lagrange Form of \( p_n(x) \)

This approach is technically intuitive, but computationally more expensive than Newton form. We will show the idea of this approach by an example first.

If the following four points are given:

\[
(x_0, f_0) = (-1, -7), \quad (x_1, f_1) = (1, 7), \quad (x_2, f_2) = (2, 4), \quad (x_3, f_3) = (5, 35)
\]

first define \( q_i(x) \), \( i = 0, 1, 2, 3 \), as follows,

\[
q_0(x) = (x - x_1)(x - x_2)(x - x_3) = (x+1)(x-2)(x-5)
\]

\[
q_1(x) = (x - x_0)(x - x_2)(x - x_3) = (x+1)(x-2)(x-5)
\]
\[ q_2(x) = (x - x_0)(x - x_1)(x - x_3) = (x+1)(x-1)(x-5) \]
\[ q_3(x) = (x - x_0)(x - x_1)(x - x_2) = (x+1)(x-1)(x-2) \]

then define \( l_i(x) \), \( i = 0, 1, 2, 3 \), as follows.

\[
\begin{align*}
  l_0(x) &= \frac{q_0(x)}{q_0(x_0)} = \frac{(x - 1)(x - 2)(x - 5)}{(-1-1)(-1-2)(-1-5)} = \frac{1}{36}(x-1)(x-2)(x-5) \\
  l_1(x) &= \frac{q_1(x)}{q_1(x_1)} = \frac{(x + 1)(x - 2)(x - 5)}{(1+1)(1-2)(1-5)} = \frac{1}{8}(x+1)(x-2)(x-5) \\
  l_2(x) &= \frac{q_2(x)}{q_2(x_2)} = \frac{(x + 1)(x - 1)(x - 5)}{(2+1)(2-1)(2-5)} = \frac{1}{9}(x+1)(x-1)(x-5) \\
  l_3(x) &= \frac{q_3(x)}{q_3(x_3)} = \frac{(x + 1)(x - 1)(x - 2)}{(5+1)(5-1)(5-2)} = \frac{1}{72}(x+1)(x-1)(x-2)
\end{align*}
\]

Each \( l_i(x) \) is a polynomial of degree 3 and satisfies the following condition.

\[
l_i(x_j) = \begin{cases} 
1, & \text{if } i = j \\
0, & \text{otherwise.} 
\end{cases} \tag{4.4}
\]

The Lagrange form of \( p_3(x) \) is

\[
p_3(x) = f_0 \cdot l_0(x) + f_1 \cdot l_1(x) + f_2 \cdot l_2(x) + f_3 \cdot l_3(x) \\
= \frac{10}{9} x^3 - \frac{50}{9} x^2 + \frac{53}{9} x + \frac{50}{9}
\]

It is easy to see that \( p_3(x_j) = f_j, j = 0, 1, 2, 3 \). Why?

In general, if \((x_0, f_0), (x_1, f_1), \ldots, (x_n, f_n)\) are given, first define \( q_k(x) \), \( k = 0, 1, \ldots, n \), as follows,

\[
q_k(x) = (x - x_0) \cdots (x - x_{k-1})(x - x_{k+1}) \cdots (x - x_n) \\
= \prod_{i=0}^{n} (x - x_i)_{i \neq k}
\tag{4.5}
\]

then define \( l_k(x) \), \( k = 0, 1, \ldots, n \), as follows.

\[
l_k(x) = q_k(x) / q_k(x_k) \tag{4.6}
\]

\( q_k(x) \) and \( l_k \) are polynomials of degree \( n \). \( l_k(x) \) are called Lagrange polynomials for the points
The Lagrange form of \( p_n(x) \) is defined as follows.

\[
p_n(x) = f_0 \cdot l_0(x) + f_1 \cdot l_1(x) + f_2 \cdot l_2(x) + \cdots + f_n \cdot l_n(x) \quad \text{(4.7)}
\]

\[
= \sum_{k=0}^{n} f_k \cdot l_k(x)
\]

Since \( l_k(x) \) satisfies a condition similar to (4.4),

\[
l_k(x_j) = \begin{cases} 
1, & \text{if } k = j \\
0, & \text{otherwise}
\end{cases}
\]

It is easy to see that \( p_n(x_j) = f_j, j = 0, 1, \ldots, n \). (Question: can you prove that \( \sum_{k=0}^{n} l_k(x) = 1 \)?)

In the following, we will study the cost of evaluating the Lagrange form at a given point. The following notations will be used.

\( A: \) denoting an addition or subtraction, and
\( M: \) denoting a multiplication or division.

From (4.5) and (4.6), we see that straight evaluation of the Lagrange polynomials takes \( 2nA + (2n-1)M \) operations for each of \( l_k(x), k = 0, 1, \ldots, n \). So the total cost of evaluating all the \( l_k(x) \) is \( 2n(n+1)A + (2n-1)(n+1)M \) operations. It takes \( nA + (n+1)M \) operations to evaluate (4.7). Therefore, totally, one needs

\[
(2n + 3)nA + 2n(n + 1)M
\]

operations to evaluate the Lagrange form at a given point.

However, if one is clever enough, he should be able to see that \( l_k(x) \) can be written as

\[
l_k(x) = \frac{\prod_{i=0}^{n} (x-x_i)}{(x-x_k)q_k(x_k)}, \quad k = 0, 1, \ldots, n
\]

and, consequently, \( p_n(x) \) can be written as

\[
p_n(x) = \sum_{k=0}^{n} f_k \cdot l_k(x) = \left( \prod_{i=0}^{n} (x-x_i) \right) \sum_{k=0}^{n} \frac{f_k/q_k(x_k)}{(x-x_k)} \quad \text{(4.8)}
\]

Note that each \( f_k/q_k(x_k) \) is independent of \( x \) and can be computed once and for all. Hence, by assuming that the values of \( f_k/q_k(x_k), k = 0, 1, \ldots, n \), are known, the computational cost of evaluating \( p_n(x) \) at a point \( x \) is only
operations. Note that the values of \((x - x_k)\) can be used both in the product and the summation in (4.8).

**4.1.2 Newton Form of \(p_n(x)\)**

This approach is not so intuitive, but is computationally efficient. Given a set of \(k+1\) data points: \((x_0, f_0)\), \((x_1, f_1)\), ..., \((x_k, f_k)\), the basic idea of this approach is to use \(p_{k-1}(x)\), the polynomial which interpolates \((x_0, f_0)\), \((x_1, f_1)\), ..., \((x_{k-1}, f_{k-1})\), as a building block to construct \(p_k(x)\). The idea is illustrated below.

To find the polynomial \(p_2(x)\) which interpolates \((1, 3)\), \((3, 11)\), and \((4, -6)\), we need to find the polynomial \(p_1(x)\) which interpolates \((1, 3)\) and \((3, 11)\) first. It is easy to see that \(p_1(x)\) can be expressed as follows.

\[
p_1(x) = 3 + 4(x - 1)
\]  
(4.9)

Then we set \(p_2(x)\) as the sum of \(p_1(x)\) and a polynomial of degree 2, \(h_2(x)\), as follows.

\[
p_2(x) = p_1(x) + h_2(x)
\]  
(4.10)

\(h_2(x)\) is to be determined. Since \(p_1(x)\) and \(p_2(x)\) both interpolate \((1, 3)\) and \((3, 11)\), it follows that

\[
h_2(1) = p_2(1) - p_1(1) = 0 \quad \text{and} \quad h_2(3) = p_2(3) - p_1(3) = 0.
\]

Hence, 1 and 3 are roots of \(h_2(x)\). Consequently, \(h_2(x)\) can be expressed as

\[
h_2(x) = A_2(x - 1)(x - 3)
\]  
(4.11)

where \(A_2\) is a constant. To find \(A_2\), note that \(p_2(x)\) interpolates \((4, -6)\), i.e., \(p_2(4) = -6\). On the other hand, from (4.9), we see that \(p_1(4) = 15\). Therefore, by substituting 4 for \(x\) in (4.10) and (4.11), we have

\[
A_2(4 - 1)(4 - 3) = p_2(4) - p_1(4) = -6 - 15 = -21.
\]

or

\[
A_2 = -7.
\]

Hence, the expression of \(p_2(x)\) is

\[
p_2(x) = 3 + 4(x - 1) - 7(x - 1)(x - 3)
\]
In general, if the polynomial $p_{k-1}(x)$ which interpolates $k$ points $(x_0, f_0), (x_1, f_1), ..., (x_{k-1}, f_{k-1})$ is known to us, we can find the polynomial $p_k(x)$ which interpolates $(x_0, f_0), (x_1, f_1), ..., (x_{k-1}, f_{k-1})$, and $(x_k, f_k)$ as follows. First, we set $p_k(x)$ as the sum of $p_{k-1}(x)$ and a degree $k$ polynomial, $h_k(x)$, as follows

$$p_k(x) = p_{k-1}(x) + h_k(x) \quad (4.12)$$

with $h_k(x)$ to be determined. Since $p_{k-1}(x)$ and $p_k(x)$ both interpolate the first $k$ data points: $(x_0, f_0), (x_1, f_1), ..., (x_{k-1}, f_{k-1})$, i.e.,

$$p_{k-1}(x_i) = f_i = p_k(x_i), \quad i = 0, 1, ..., k-1,$$

it follows that

$$h_k(x_i) = p_k(x_i) - p_{k-1}(x_i) = 0$$

for $i = 0, 1, ..., k-1$. Hence, $x_0, x_1, ..., x_{k-1}$ are roots of $h_k(x)$. Consequently, $h_k(x)$ can be expressed as

$$h_k(x) = A_k (x - x_0)(x - x_1) \cdots (x - x_{k-1}) \quad (4.13)$$

where $A_k$ is a constant. From (4.12) and (4.13), we have the following recurrence relation for $p_k(x)$.

$$p_k(x) = p_{k-1}(x) + A_k (x - x_0)(x - x_1) \cdots (x - x_{k-1}) \quad (4.14)$$

$A_k$ is the leading coefficient of $p_k(x)$. $A_k$ can be computed using the facts that $p_k(x)$ interpolates $(x_k, f_k)$ and $p_{k-1}(x)$ is known to us.

By recursively applying the recurrence relation (4.14) to $p_{k-1}(x), p_{k-2}(x), p_{k-3}(x), ..., we get the following expression for $p_k(x)$

$$p_k(x) = p_0(x) + A_1(x - x_0) + A_2(x - x_0)(x - x_1) + \cdots + A_k(x - x_0)(x - x_1) \cdots (x - x_{k-1}) \quad (4.15)$$

where $A_i$ is the leading coefficient of $p_i(x), i = 1, 2, ..., k$, and $p_0(x)$ is a polynomial of degree zero which interpolates $(x_0, f_0)$. Obviously,

$$p_0(x) = f_0 \quad (4.16)$$

$A_i$ can be computed using the facts that $p_i(x)$ interpolates $(x_i, f_i)$ and $p_{i-1}(x)$ is known to us. An efficient algorithm for the computation of $A_i$ will be shown below.

We need some notation and terminology first. $A_i, i = 1, ..., k$, are usually denoted $f [x_0, x_1, ..., x_i]$ and are called the $i$th divided difference (of $f$) at $x_0, x_1, ..., x_i$. With this
notation, \( p_k(x) \) can be expressed as

\[
p_k(x) = f_0 + f_0[x_0, x_1](x - x_0) + f_0[x_0, x_1, x_2](x - x_0)(x - x_1) + \cdots \\
+ f_0[x_0, x_1, \ldots, x_k](x - x_0)(x - x_1) \cdots (x - x_{k-1})
\]  
(4.17)

\( f[x_0, x_1, \ldots, x_i] \) satisfies a recurrence relation.

**Theorem.** If we define \( f[i] \equiv f_i \), then \( f[x_0, x_1, \ldots, x_i] \) satisfies the following formula

\[
f[x_0, x_1, \ldots, x_i] = \frac{f[x_1, x_2, \ldots, x_i] - f[x_0, x_1, \ldots, x_{i-1}]}{x_i - x_0}
\]  
(4.18)

where \( f[x_1, x_2, \ldots, x_i] \) is the leading coefficient of the Newton interpolating polynomial which interpolates \( x_1, x_2, \ldots, x_i \).

**Proof.** \( f[x_0, x_1, \ldots, x_i] \) is the leading coefficient of the Newton interpolating polynomial \( p_i(x) \) which interpolates \( x_0, f_0 \), \( x_1, f_1 \), \ldots, \( x_i, f_i \). \( f[x_0, x_1, \ldots, x_{i-1}] \) is the leading coefficient of the Newton interpolating polynomial \( p_{i-1}(x) \) which interpolates \( x_0, f_0 \), \( x_1, f_1 \), \ldots, \( x_{i-1}, f_{i-1} \). If the Newton interpolating polynomial which interpolates \( x_1, f_1 \), \( x_2, f_2 \), \ldots, \( x_{i-1}, f_{i-1} \) is called \( q(x) \), then \( f[x_1, x_2, \ldots, x_i] \) is the leading coefficient of \( q(x) \) and the polynomials \( p_{i-1}(x), p_i(x), \) and \( q(x) \) satisfy the following relation.

\[
p_i(x) = q(x) + \frac{x-x_i}{x_i-x_0} \left[ q(x) - p_{i-1}(x) \right]
\]  
(4.19)

(to prove (4.19), observe that the right side is polynomial of degree \( n \) and it interpolates \( (x_0, f_0), (x_1, f_1), \ldots, (x_i, f_i) \) too). The leading coefficient of the right side of (4.19) is

\[
\frac{f[x_1, \ldots, x_{i-1}, x_i] - f[x_0, x_1, \ldots, x_{i-1}]}{x_i - x_0}
\]

and it must be same of the leading coefficient of the left side of (4.19), \( f[x_0, x_1, \ldots, x_i] \). We therefore have (4.18). \( \square \)

Hence, one can compute \( f[x_0, x_1], f[x_0, x_1, x_2], f[x_0, x_1, x_2, x_3], \ldots \) as follows.

\[
f[x_0, x_1] = \frac{f[x_1] - f[x_0]}{x_1 - x_0} = \frac{f_1 - f_0}{x_1 - x_0}
\]

\[
f[x_0, x_1, x_2] = \frac{f[x_1, x_2] - f[x_0, x_1]}{x_2 - x_0}
\]
More specifically, if \((x_0, f_0) = (1, 3), (x_1, f_1) = (3, 11), (x_2, f_2) = (4, -6), (x_3, f_3) = (5, 3)\), we can compute \(f[x_0, x_1], f[x_0, x_1, x_2],\) and \(f[x_0, x_1, x_2, x_3]\) as follows. First, we compute \(f[x_0, x_1], f[x_1, x_2], \) and \(f[x_2, x_3].\)

\[
f[x_0, x_1] = \frac{f[x_1] - f[x_0]}{x_1 - x_0} = \frac{f_1 - f_0}{x_1 - x_0} = \frac{11 - 3}{3 - 1} = 4
\]

\[
f[x_1, x_2] = \frac{f[x_2] - f[x_1]}{x_2 - x_1} = \frac{f_2 - f_1}{x_2 - x_1} = \frac{-6 - 11}{4 - 3} = -17
\]

\[
f[x_2, x_3] = \frac{f[x_3] - f[x_2]}{x_3 - x_2} = \frac{f_3 - f_2}{x_3 - x_2} = \frac{3 + 6}{5 - 4} = 9
\]

Then, we compute \(f[x_0, x_1, x_2] \) and \(f[x_1, x_2, x_3].\)

\[
f[x_0, x_1, x_2] = \frac{f[x_1, x_2] - f[x_0, x_1]}{x_2 - x_0} = \frac{-17 - 4}{4 - 1} = -7
\]

\[
f[x_1, x_2, x_3] = \frac{f[x_2, x_3] - f[x_1, x_2]}{x_3 - x_1} = \frac{9 + 17}{5 - 3} = 13
\]

Finally we compute \(f[x_0, x_1, x_2, x_3].\)

\[
f[x_0, x_1, x_2, x_3] = \frac{f[x_1, x_2, x_3] - f[x_0, x_1, x_2]}{x_3 - x_0} = \frac{13 + 7}{5 - 1} = 5
\]

Hence, the Newton form of \(p_3(x)\) is as follows.

\[
p_3(x) = 3 + 4(x - 1) - 7(x - 1)(x - 3) + 5(x - 1)(x - 3)(x - 4)
\]