

Interpolation Theory

The concept of *interpolation* is to select a function $P(x)$ from a given class of functions in such a way that the graph of $y = P(x)$ passes through the given data points (x_i, y_i) , $i = 1, 2, \dots, n$. The points may arise as measurements in a physical problem, or they may be obtained from a known function. The interpolating function is usually chosen from a restricted class of functions, with polynomials being the most commonly used class.

Theorem 1 (Weierstrass Approximation Theorem). *If $f(x)$ is a continuous function on $[a, b]$ and $\epsilon > 0$ is given, then there exists a polynomial $P(x)$, defined on $[a, b]$, with the property that*

$$|f(x) - P(x)| < \epsilon \quad \text{for all } x \in [a, b].$$

Theorem 2. *Given $n + 1$ distinct points x_0, x_1, \dots, x_n and $n + 1$ ordinates y_0, y_1, \dots, y_n , there is a polynomial $P(x)$ of degree $\leq n$ that interpolates to y_i at x_i , $i = 0, 1, 2, \dots, n$. This polynomial $P(x)$ is unique among the set of all polynomials of degree at most n .*

Proof. Consider the following linear system of $n + 1$ equations and $n + 1$ unknowns

$$a_0 + a_1x_0 + a_2x_0^2 + \dots + a_nx_0^n = y_0$$

\vdots

$$a_0 + a_1x_n + a_2x_n^2 + \dots + a_nx_n^n = y_n .$$

The unknowns are (a_0, a_1, \dots, a_n) . The proof follows from the fact that this system has a unique solution.

♣ **Taylor Polynomials.** Let $f(x)$ be $n + 1$ times differentiable near a point x_0 . Then the n -degree Taylor polynomial of $f(x)$ at x_0 is

$$P_n(x) = f(x_0) + f'(x_0)(x - x_0) + f''(x_0)\frac{(x - x_0)^2}{2!} + \dots + f^{(n)}(x_0)\frac{(x - x_0)^n}{n!}.$$

The function

$$R_n(x) = P_n(x) - f(x) = \frac{f^{(n+1)}(\xi(x))}{(n + 1)!}(x - x_0)^{(n+1)}$$

is called the *Remainder function* or the *Error function*.

Example. Approximate $\ln 1.1$ using a Taylor polynomial of degree 3.

Note that if $f(x) = \frac{1}{1+x}$, then $\int_0^{0.1} f(x)dx = \ln 1.1$. Thus we need to integrate the polynomial $P_3(x) = 1 - x + x^2 - x^3$ from zero to 0.1.

$$\int_0^{0.1} (1 - x + x^2 - x^3)dx = \left[x - \frac{x^2}{2} + \frac{x^3}{3} - \frac{x^4}{4} \right]_0^{0.1} = 0.09533 .$$

The error is

$$\left| \int_0^{0.1} f(x) - P_3(x) \right| \approx \left| \int_0^{0.1} x^4 dx \right| = \left[\frac{x^5}{5} \right]_0^{0.1} = 0.000002.$$

Bernstein Polynomials. Given a function defined on $[0, 1]$, the Bernstein polynomial of degree n for $f(x)$ is given by

$$B_n(x) = \sum_{k=0}^n \binom{n}{k} f\left(\frac{k}{n}\right) x^k (1-x)^{n-k}$$

where $\binom{n}{k}$ denotes $\frac{n!}{k!(n-k)!}$.

It can be shown that, if $f(x)$ is continuous on $[0, 1]$ and $x_0 \in [0, 1]$, then

$$\lim_{n \rightarrow \infty} B_n(x_0) = f(x_0).$$

The polynomial can be used in a constructive proof of Weierstrass Theorem.

♣ **Lagrange Polynomial.** Given two points (x_0, y_0) and (x_1, y_1) with $x_0 \neq x_1$, we can draw a straight line through them. The line is the graph of the linear polynomial

$$P_1(x) = \frac{(x_1 - x)}{x_1 - x_0} y_0 + \frac{(x - x_0)}{x_1 - x_0} y_1.$$

Given $n + 1$ distinct points x_0, x_1, \dots, x_n and $n + 1$ ordinates y_0, y_1, \dots, y_n ; according to Theorem 2, there is a unique polynomial $P_n(x)$ that interpolates y_k at $x_k, k = 0, 1, 2, \dots, n$.

For $k = 1, 2, \dots, n$, define the polynomial

$$L_{n,k}(x) = \frac{(x - x_0) \cdots (x - x_{k-1})(x - x_{k+1}) \cdots (x - x_n)}{(x_k - x_0) \cdots (x_k - x_{k-1})(x_k - x_{k+1}) \cdots (x_k - x_n)} = \prod_{\substack{j=0 \\ k \leq j \leq n}}^n \frac{(x - x_j)}{(x_k - x_j)}.$$

Then the polynomial

$$L_n(x) = \sum_{k=0}^n y_k L_{n,k}(x)$$

is called the *Lagrange Interpolation Polynomial*.

Note that

$$L_{n,k}(x_j) = \delta_{k,j}, \quad 0 \leq k, j \leq n$$

where $\delta_{k,j}$ is called *Kronecker delta function*,

$$\delta_{k,j} = \begin{cases} 1, & j = k; \\ 0, & j \neq k. \end{cases}$$

Theorem 3. If x_0, x_1, \dots, x_n are distinct numbers in the interval $[a, b]$ and if $f(x) \in C^{n+1}[a, b]$, then, for each $x \in [a, b]$, there exist a number $\xi(x) \in (a, b)$ such that

$$f(x) = L_n(x) + \frac{f^{(n+1)}(\xi(x))}{(n+1)!} (x - x_0)(x - x_1) \cdots (x - x_n).$$

Example. Let $f(x) = \frac{1}{1+x}$, the find the Lagrange polynomial of degree two passing through the points $(0, f(0))$, $(1, f(1))$ and $(2, f(2))$.

We have $f(0) = 1$, $f(1) = \frac{1}{2}$, $f(2) = \frac{1}{3}$ and

$$L_{2,0}(x) = \frac{(x-1)(x-2)}{(0-1)(0-2)} = \frac{(x-1)(x-2)}{2} = \frac{x^2 - 3x + 2}{2}$$

$$L_{2,1}(x) = \frac{(x-0)(x-2)}{(1-0)(1-2)} = -(x-0)(x-2) = -x^2 + 2x$$

$$L_{2,2}(x) = \frac{(x-0)(x-1)}{(2-0)(2-1)} = \frac{(x-0)(x-1)}{2} = \frac{x^2 - x}{2}.$$

Hence

$$L_2(x) = \sum_{k=0}^2 L_{2,k}(x)f(k) = \frac{x^2 - 3x + 2}{2}(1) + (-x^2 + 2x)\left(\frac{1}{2}\right) + \frac{x^2 - x}{2}\left(\frac{1}{3}\right) = \frac{x^2}{6} - \frac{2x}{3} + 1.$$

♣ **Newton Divided Differences.** The Lagrange form of the interpolation polynomial can be used for interpolating a function. But there are other methods that are much more convenient and produce the same interpolating polynomial.

Suppose $P_{n-1}(x)$ is a polynomial of degree n that interpolate $f(x)$. We would like to pass from $P_{n-1}(x)$ to another interpolation polynomial of degree one greater; so we need to find

$$P_n(x) = P_{n-1}(x) + C(x) \quad C(x) = \text{correction term}$$

Then in general $C(x)$ is a polynomial of degree n , since usually

$$\text{degree}(P_{n-1}) = n - 1 \text{ and } \text{degree}(P_n) = n.$$

Also we have

$$C(x_k) = P_n(x_k) - P_{n-1}(x_k) = f(x_k) - f(x_k) = 0. \quad k = 0, 1, \dots, n - 1$$

Since $P_n(x) = f(x_n)$, we have

$$a_n = \frac{f(x_n) - P_{n-1}(x_n)}{(x_n - x_0) \cdots (x_n - x_{n-1})}.$$

This coefficient a_n is called the n -th order Newton Divided Difference of $f(x)$, and it is denoted by

$$a_n \equiv f[x_0, x_1, \dots, x_n]$$

Thus our interpolation formula becomes

$$P_n(x) = P_{n-1}(x) + (x - x_0) \cdots (x - x_{n-1})f[x_0, \dots, x_n]$$

Using the Lagrange formula, we obtain

$$f[x_0, x_1, \dots, x_n] = \frac{f[x_1, x_2, \dots, x_n] - f[x_0, x_1, \dots, x_{n-1}]}{x_n - x_0}$$

We have

$$\begin{aligned} P_0(x) &= f(x_0) \\ P_1(x) &= f(x_0) + (x - x_0)f[x_0, x_1] \\ &\vdots = \vdots \\ P_n(x) &= f(x_0) + (x - x_0)f[x_0, x_1] + (x - x_0)(x - x_1)f[x_0, x_1, x_2] + \dots \\ &\quad (x - x_0) \dots (x - x_{n-1})f[x_0, x_1, \dots, x_n] \end{aligned}$$

This is called *Newton's divided differences formula for the interpolation polynomial*. It is much better for computation than the Lagrange formula. To construct the divided differences, we use the format shown in the following table.

x_i	$f(x_i)$	$f[x_i, x_{i+1}]$	$D^2 f[x_i]$	$D^3 f[x_i]$	$D^4 f[x_i]$
x_0	y_0				
x_1	y_1	$f[x_0, x_1]$			
x_2	y_2	$f[x_1, x_2]$	$f[x_0, x_1, x_2]$		
x_3	y_3	$f[x_2, x_3]$	$f[x_1, x_2, x_3]$	$f[x_0, x_1, x_2, x_3]$	
\vdots	\vdots	\vdots	\vdots	\vdots	\vdots

Example. We construct a divided difference table for $f(x) = \sqrt{x}$, shown in the next table.

x_i	$f(x_i)$	$f[x_i, x_{i+1}]$	$D^2 f[x_i]$	$D^3 f[x_i]$	$D^4 f[x_i]$
2.0	1.414214				
		0.34924			
2.1	1.449138		-0.0411		
		0.34102		0.009167	
2.2	1.483240		-0.03835		-0.002084
		0.33335		0.008333	
2.3	1.516575		-0.03585		
		0.32618			
2.4	1.549193				

The polynomial

$$P_4(x) = 1.41214 + (x - 2.0)[(0.34924) + (x - 2.1)[(-0.0411) + (x - 2.2)[(0.009167) + (x - 2.3)[(-0.002084)]]]]$$

is known as the *Newton Forward Divided-Difference form*. The following polynomial is known as the *Newton Backward Divided-Difference form*:

$$Q_4(x) = 1.549193 + (x - 2.4)[(0.32618) + (x - 2.3)[(-0.03585) + (x - 2.2)[(0.008333) + (x - 2.1)[(-0.002084)]]]].$$

Algorithm: DIVDIF(N,X,D)

Remark: On entrance, $X = (X(0), \dots, X(N))$ and $D = (F(X_1) \dots F(X_n))$. On exit, $D(J)$ will contain $F[X_0, X_1, \dots, X_J]$.

Step 1. Input 'N' a positive integer;

Step 2. Input 'X' a vector of points $x(0), \dots, X(N)$;

Step 3. Input 'D' a vector of values of some function evaluated at the nodes in X;

Step 4 FOR $I := 1, N$; FOR $J := N, I, -1$; $D(J) := \frac{D(J) - D(J-1)}{X(J) - X(J-1)}$ RETURN;

Step 5 STOP.

Algorithm: FORWDINTERP(N,X,F)

Remark: On entrance, $X = (X(0), \dots, X(N))$ and $D = (D(0), D(1), \dots, D(n))$. On exit, P will contain the value $P_N(T)$ of the N-th degree polynomial interpolation of $F(X)$.

Step 1. $P := D_N$;

Step 2. for $K := N - 1, 0, -1$; $P := D(K) + (T - X(K))P$ Return;

Step 3. Stop.

♣ **Inverse Interpolation.** Suppose $f(x) \in C^1[a, b]$ and $f'(x) \neq 0$ on (a, b) . If $f(x)$ has a zero x^* in $[a, b]$, then we may use the Divided Differences, to approximate x^* .

We explain the method by the use of an example:

Example. Consider the continuous function $f(x) = x^3 - 3x + 1$ with $f(0) = 1$ and $f(1) = -1$. Then according to the Intermediate-Value-Theorem, it has a zero on $[0, 1]$. Although $f'(1) = 1$, but $f'(x) \neq 0$ on $(0, 1)$. Now, to create an inverse interpolating polynomial $q_2(x)$ of degree 2, we choose the following points:

$$(x_0, y_0) = (0, 1), \quad (x_1, y_1) = \left(\frac{1}{2}, -\frac{3}{8}\right), \quad \text{and} \quad (x_2, y_2) = (1, -1).$$

By interchanging x and y , we obtain

$$(y_0, x_0) = (1, 0), \quad (y_1, x_1) = \left(-\frac{3}{8}, \frac{1}{2}\right), \quad \text{and} \quad (y_2, x_2) = (-1, 1).$$

The Newton Divided Differences Table for the second degree polynomial $q_2(x)$ is as follows:

y_i	x_i	D^1	D^2
1	0		
$-\frac{3}{8}$	$\frac{1}{2}$	$-\frac{4}{11}$	
-1	1	$-\frac{4}{5}$	$\frac{12}{55}$

We have:

$$q_1(y) = 0 + (y - 1) \left[-\frac{4}{11} + (y + \frac{3}{8}) \left[\frac{12}{55} \right] \right].$$

By replacing y by 0 in $q(y)$, we obtain an approximation for x^* .

$$x_1^* \approx q_1(0) = 0 + (0 - 1) \left[-\frac{4}{11} + (0 + \frac{3}{8}) \left[\frac{12}{55} \right] \right] = -\frac{31}{110} = -0.2818 \quad \text{with} \quad f(-0.2818) = -1.4691.$$

To improve our approximation, we may use $(y_3, x_3) = (y_1^*, x_1^*) = (-1.4691, -0.2818)$ to generate $q_2(x)$, using the Divided Differences.

y_i	x_i	D^1	D^2	D^3
1	0			
-0.3750	0.5000	-0.3636		
-1.0000	1.0000	-0.8000	-0.2182	
0.1769	-0.2818	-0.6102	0.3439	-0.1527

$$q_2(y) = 0 + (y - 1) [-0.3636 + (y + 0.3750) [-0.2182 + (y + 1) [-0.1527]]].$$

We have $x_2^* \approx q_2(0) = 0.5027$ with $f(x_2^*) = -0.37112$. We continue this process until we obtain a satisfactory approximation.

♣ Hermite Interpolation. For a variety of applications, it is convenient to pick a polynomial $P(x)$ interpolating the function $f(x)$ at certain points x_0, x_1, \dots, x_n ; and also the derivative of the polynomial $P'(x)$, interpolates the derivative function $f'(x)$. Since the first derivative of $f(x)$ and $P(x)$ agree at the interpolation points, they have basically the same “shape” in the neighborhood of the interpolating points.

To construct the Hermite interpolation polynomial, we use the Newton’s Divided Differences, by repeating the interpolating points and using the derivative of $f(x)$ at

those points. Here is the table:

x_i	$f(x_i)$	$f[x_i, x_{i+1}]$	$D^2 f[x_i]$	$D^3 f[x_i]$	$D^4 f[x_i]$	$D^5 f[x_i]$
x_0	y_0	0				
x_0	y_0	$f'(x_0)$				
x_1	y_1	$f[x_0, x_1]$	$f[x_0, x_0, x_1]$			
x_1	y_1	$f'(x_1)$	$f[x_0, x_1, x_1]$	$f[x_0, x_0, x_1, x_1]$		
x_2	y_2	$f[x_1, x_2]$	$f[x_1, x_1, x_2]$	$f[x_0, x_1, x_1, x_2]$	$f[x_0, x_0, x_1, x_1, x_2]$	
x_2	y_2	$f'(x_2)$	$f[x_1, x_2, x_2]$	$f[x_1, x_1, x_2, x_2]$	$f[x_0, x_1, x_1, x_2, x_2]$	$f[x_0, x_0, x_1, x_1, x_2, x_2]$
\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots

Consider the function $f(x)$ with $n + 1$ given points x_0, x_1, \dots, x_n .

The Hermite interpolating of degree $2n + 1$ is given by

$$H_{2n+1} = f(x_0) + (x - x_0) [f'(x_0) + (x - x_1) [f[x_0, x_0, x_1] + (x - x_2) [f[x_0, x_0, x_1, x_1] + \dots \dots]]] .$$

The most widely used form of Hermite interpolation is the cubic polynomial, which uses

$$P(a) = f(a) \quad P'(a) = f'(a) \quad P(b) = f(b) \quad P'(b) = f'(b).$$

$$H_3(x) = \left[1 + 2 \frac{x-a}{b-a} \right] \left[\frac{b-x}{b-a} \right]^2 f(a) + \left[1 + 2 \frac{b-x}{b-a} \right] \left[\frac{x-a}{b-a} \right]^2 f(b) + \frac{(x-a)(b-x)^2}{(b-a)^2} f'(a) - \frac{(x-a)^2(b-x)}{(b-a)^2} f'(b).$$

♣ Interpolation Using Spline Functions. The simplest method of interpolation is to connect the given points by straight line segments; this type of interpolation is called Piecewise Linear Interpolation, and the associated interpolation function is denoted by $S(x)$. Although $S(x)$ agrees with the original function at the nodes, it has the disadvantage of not having a smooth graph.

Another choice is to connect the data points by using succession of quadratic interpolation polynomials. Although the resulting curve is smoother than the first method, however, the graph can still have some corners. For many applications, this interpolating function may be completely adequate, but we would like to obtain a smooth curve. We shall see that a cubic spline produces the desired curve.

♡ Cubic Spline. Suppose $n + 1$ data points $(x_0, y_0), (x_1, y_1), \dots, (x_n, y_n)$ are given. For simplicity, assume

$$x_0 < x_1 < \dots < x_n$$

and let $a = x_0$, $b = x_n$. We seek a function $S(x)$ defined on $[a, b]$ that interpolates the data

$$S(x_i) = y_i, \quad i = 0, 1, \dots, n.$$

For smoothness of $S(x)$, we need to require that $S'(x)$ and $S''(x)$ be continuous. In addition, we would like to have the curve follow the general shape given by the piecewise linear function connecting the data points (x_i, y_i) . The standard way in which this has been done is to ask that the derivative $S'(x)$ not change too rapidly between node points. This has been carried out by requiring the second derivative $S''(x)$ to be as small as possible and, more precisely, by requiring that

$$\int_a^b [S''(x)]^2 dx$$

be also as small as possible. This may not be a perfect mathematical realization of the idea of a smooth shape-preserving interpolation function for the data $\{(x_i, y_i)\}$, but it usually gives a very good interpolating function.

There is a unique solution $S(x)$ to this problem, and it satisfies the following:

- S1. $S(x)$ is a cubic polynomial, denoted $S_j(x)$ on each $[x_j, x_{j+1}]$, for $j = 0, 1, \dots, n-1$;
- S2. $S(x_j) = f(x_j)$ for each $j = 0, 1, \dots, n$;
- S3. $S_{j+1}(x_{j+1}) = S_j(x_{j+1})$ for each $j = 0, 1, \dots, n-2$;
- S4. $S'_{j+1}(x_{j+1}) = S'_j(x_{j+1})$ for each $j = 0, 1, \dots, n-2$;
- S5. $S''_{j+1}(x_{j+1}) = S''_j(x_{j+1})$ for each $j = 0, 1, \dots, n-2$;
- S6. one of the following set of boundary conditions is satisfied:

$$(i) \quad S''(x_0) = S''(x_n) = 0 \quad (\text{Free boundary})$$

$$(ii) \quad S'(x_0) = f'(x_0) \quad \text{and} \quad S'(x_n) = f'(x_n) \quad (\text{Clamped boundary})$$

To construct the cubic spline interpolant for a given function $f(x)$, the conditions (S_1 through S_6) can be applied to the cubic polynomials

$$S_j(x) = A_j + B_j(x - x_j) + C_j(x - x_j)^2 + D_j(x - x_j)^3 \quad \text{for each } j = 0, 1, \dots, n-1.$$

For $j = 0, 1, \dots, n-1$, define $h_j = x_{j+1} - x_j$, then we have

$$A_j = S_j(x_j) = f(x_j) \quad \text{and} \quad A_{j+1} = A_j + B_j h_j + C_j h_j^2 + D_j h_j^3$$

$$B_0 = f'(x_0), \quad B_j = S'_j(x_j) \quad (j > 0) \quad \text{and} \quad B_{j+1} = B_j + 2C_j h_j + 3D_j h_j^2$$

$$B_j = S'(x_j), \quad C_n = \frac{S''(x_n)}{2} \quad C_{j+1} = C_j + 3D_j h_j.$$

Since h_j 's and A_j 's are given, once the values of C_j 's are known, then it is a simple matter to find B_j 's and D_j 's.

Free Boundary. When $S''(a) = S''(b) = 0$, then the vector $C = (C_0, C_1, \dots, C_n)^t$ is the unique solution to the equation.

$$\begin{pmatrix} 1 & 0 & \dots & \dots & 0 \\ h_0 & 2(h_0 + h_1) & h_1 & & \\ 0 & h_1 & 2(h_1 + h_2) & h_2 & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ \vdots & \vdots & h_{n-2} & 2(h_{n-2} + h_{n-1}) & h_{n-1} \\ 0 & \dots & \dots & 0 & 1 \end{pmatrix} \begin{pmatrix} C_0 \\ C_1 \\ \vdots \\ \vdots \\ C_{n-1} \\ C_n \end{pmatrix} = \begin{pmatrix} 0 \\ \frac{3(A_2 - A_1)}{h_1} - \frac{3(A_1 - A_0)}{h_0} \\ \vdots \\ \frac{3(A_n - A_{n-1})}{h_{n-1}} - \frac{3(A_{n-1} - A_{n-2})}{h_{n-2}} \\ 0 \end{pmatrix}.$$

Clamped Boundary. For $S'(a) = f'(a)$ and $S'(b) = f'(b)$, the vector $C = (C_0, C_1, \dots, C_n)^t$ is the unique solution to the equation

$$\begin{pmatrix} 2h_0 & h_0 & 0 & \dots & 0 \\ h_0 & 2(h_0 + h_1) & h_1 & & \\ 0 & h_1 & 2(h_1 + h_2) & h_2 & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ \vdots & \vdots & h_{n-2} & 2(h_{n-2} + h_{n-1}) & h_{n-1} \\ 0 & \dots & \dots & h_{n-1} & 2h_{n-1} \end{pmatrix} \begin{pmatrix} C_0 \\ C_1 \\ \vdots \\ \vdots \\ C_{n-1} \\ C_n \end{pmatrix} = \begin{pmatrix} \frac{3(A_1 - A_0)}{h_0} - 3f'(a) \\ \frac{3(A_2 - A_1)}{h_1} - \frac{3(A_1 - A_0)}{h_0} \\ \vdots \\ \frac{3(A_n - A_{n-1})}{h_{n-1}} - \frac{3(A_{n-1} - A_{n-2})}{h_{n-2}} \\ 3f'(b) - \frac{3(A_n - A_{n-1})}{h_{n-1}} \end{pmatrix}.$$