

# Chapter Three Numerical Approximation

(MA6624)

In this chapter, we study the approximation to a function, its derivative and integration.

**3.1 Polynomial interpolation.** Assume that we have the following data

Year	1940	1950	1960	1970	1980	1990
population	132165	151326	179323	203302	226542	249633

We need to find an approximate function  $p(x)$  based on the data, particularly a polynomial. An interpolation function for given data is a function which passes through all points given.

(i) *Existence.* We consider a general problem: find a polynomial of degree  $n$  satisfying the data

$x$	$x_0$	$x_1$	.....	$x_n$
$y$	$y_0$	$y_1$	.....	$y_n$

where we assume that  $x_i \neq x_j$  for  $i \neq j$ . Let

$$P_n(x) = a_0 + a_1x + \dots + a_nx^n.$$

Then

$$P_n(x_j) = a_0 + a_1x_j + \dots + a_nx_j^n = y_j \quad j = 0, 1, \dots, n$$

which is a linear system with  $n + 1$  unknowns. In matrix form

$$\begin{bmatrix} 1 & x_0 & x_0^2 & x_0^n \\ 1 & x_1 & x_1^2 & x_1^n \\ \dots & & & \\ 1 & x_n & x_n^2 & x_n^n \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{bmatrix} = \begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_n \end{bmatrix}$$

It has been proved that the system has a unique solution. An interpolation based on the data  $\{x_j, y_j\}$  is called Lagrange interpolation.

(ii) *Lagrange form.* Now we present the Lagrange formulation of interpolation polynomials.

First we consider a simple case:  $n = 1$ , i.e., find a linear function (polynomial) passing through two given points. The equation of the line is given by

$$\frac{x - x_0}{x_1 - x_0} = \frac{y - y_0}{y_1 - y_0}$$

i.e.,

$$y = \frac{x - x_1}{x_0 - x_1}y_0 + \frac{x - x_0}{x_1 - x_0}y_1$$

The linear approximation is defined by

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

which satisfies

$$P_1(x_0) = y_0 \quad P_1(x_1) = y_1.$$

Let

$$L_0(x) = \frac{x - x_1}{x_1 - x_1} \quad L_1(x) = \frac{x - x_0}{x_1 - x_0}$$

Obviously

$$L_0(x_0) = 1, \quad L_0(x_1) = 0 \quad L_1(x_0) = 0 \quad L_1(x_1) = 1$$

i.e.,

$$L_i(x_j) = \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases}$$

In general (given data  $\{x_j, y_j\}_{j=0}^n$ ), we need to construct the function  $L_j(x), j = 0, 1, \dots, n$  (or use the notation  $L_j^n(x)$ ), such that the above equation is satisfied. We have

$$L_j(x) = \frac{(x - x_0) \cdots (x - x_{j-1})(x - x_{j+1}) \cdots (x - x_n)}{(x_j - x_0) \cdots (x_j - x_{j-1})(x_j - x_{j+1}) \cdots (x_j - x_n)}$$

and the Lagrange form of interpolation polynomial is given by

$$P_n(x) = \sum_{j=0}^n L_j(x) y_j.$$

*Example 1.* Find a quadratic interpolation polynomial for the given data

x	2	2.5	4
y	0.5	0.4	0.25

We have

$$\begin{aligned} L_0(x) &= \frac{(x - x_1)(x - x_2)}{(x_0 - x_1)(x_0 - x_2)} = \frac{(x - 2.5)(x - 4)}{(2 - 2.5)(2 - 4)} = (x - 2.5)(x - 4) \\ L_1(x) &= \frac{(x - x_0)(x - x_2)}{(x_1 - x_0)(x_1 - x_2)} = \frac{(x - 2)(x - 4)}{(2.5 - 2)(2.5 - 4)} = -\frac{1}{0.75}(x - 2)(x - 4) \\ L_2(x) &= \frac{(x - x_0)(x - x_1)}{(x_2 - x_0)(x_2 - x_1)} = \frac{(x - 2)(x - 2.5)}{(4 - 2)(4 - 2.5)} = (x - 2)(x - 2.5) \end{aligned}$$

The quadratic Lagrange interpolation polynomial is

$$P_2(x) = (x - 2.5)(x - 4) \frac{1}{2} - \frac{1}{0.75}(x - 2)(x - 4) \cdot 0.4 + (x - 2)(x - 2.5) \cdot 0.25.$$

(iii) *Newton form.* For given data  $\{x_j, y_j\}_{j=0}^n$ , we can define a sequence of polynomials  $P_0, P_1, \dots, P_n$ , where  $P_k$  is an interpolation polynomial of degree  $k$  based on the data  $\{x_j, y_j\}_{j=0}^k$  with the form

$$P_k = c_0 + c_1(x - x_0) + c_2(x - x_0)(x - x_1) + \dots + c_k(x - x_0) \cdots (x - x_{k-1}) \quad (1)$$

and

$$P_k(x) = P_{k-1}(x) + c_k(x - x_0) \cdots (x - x_{k-1}).$$

We need to find  $c_j$  such that

$$P_k(x_j) = y_j, \quad j = 0, 1, 2, \dots, k.$$

The first few cases are

$$\begin{aligned} P_0(x) &= c_0 = y_0 \\ P_1(x) &= c_0 + c_1(x - x_0) \\ P_2(x) &= c_0 + c_1(x - x_0) + c_2(x - x_0)(x - x_1). \end{aligned} \tag{2}$$

Thus

$$P_k(x_k) = P_{k-1}(x_k) + c_k(x_k - x_0)(x_k - x_1) \cdots (x_k - x_{k-1})$$

Solving the above equation gives

$$c_k = \frac{y_k - P_{k-1}(x_k)}{(x_k - x_0)(x_k - x_1) \cdots (x_k - x_{k-1})}, \quad c_0 = y_0.$$

*Example 2.* Find an interpolation formula for  $n = 1$  (two points) in Lagrange form and Newton form, respectively.

*Solution* In Lagrange form,

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

In Newton form,

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

**Divided difference.** Let  $y = f(x)$  and  $y_j = f(x_j)$ . We define

$$f[x_0, x_1] = \frac{f(x_1) - f(x_0)}{x_1 - x_0} = \frac{y_1 - y_0}{x_1 - x_0}, \quad f[x_i, x_j] = \frac{f(x_j) - f(x_i)}{x_j - x_i}$$

which is called first order divided difference. The second-order divided difference is defined by

$$f[x_0, x_1, x_2] := \frac{f[x_1, x_2] - f[x_0, x_1]}{x_2 - x_0}$$

and higher-order divided difference be defined analogously.

In Newton form,

$$c_0 = y_0, \quad c_1 = \frac{y_1 - y_0}{x_1 - x_0} = f[x_0, x_1]$$

and

$$c_2 = \frac{y_2 - P_1(x_2)}{(x_2 - x_0)(x_2 - x_1)} = \frac{\frac{y_2 - y_1}{x_2 - x_1} - \frac{y_1 - y_0}{x_1 - x_0}}{x_2 - x_0} = \frac{f[x_1, x_2] - f[x_0, x_1]}{x_2 - x_0} = f[x_0, x_1, x_2]$$

Similarly,

$$c_k = f[x_0, x_1, \dots, c_k].$$

(iv) *Error analysis.*

**Theorem 3.1** Let  $f \in C^{n+1}[a, b]$  and let  $P_n(x)$  be the polynomial of degree  $\leq n$  that interpolates the function  $f(x)$  at  $n + 1$  distinct points  $\{x_j\}_{j=0}^n$  in the interval  $[a, b]$ . Then

$$f(x) - P_n(x) = \frac{f^{(n+1)}(\xi)}{(n+1)!} \prod_{j=0}^n (x - x_j) \quad (3)$$

where  $\xi \in (a, b)$ .

*Proof.* If  $x$  is one of nodes of interpolation  $x_j$ , the assertion is obviously true since both sides of the above equation reduce to 0. Now we only consider  $x \neq x_j, j = 0, 1, \dots, n$ . Let

$$\omega(t) = \prod_{j=0}^n (t - x_j)$$

and for a given  $x \in (a, b)$ ,

$$g(t) = f(t) - P_n(t) - \lambda\omega(t)$$

be a function of  $t$ , where

$$\lambda = \frac{f(x) - P_n(x)}{\omega(x)}$$

It follows that

$$g(x) = 0.$$

Since  $g(x) \in C^{n+1}[a, b]$  and

$$g(x) = g(x_0) = \dots = g(x_n) = 0,$$

By the mean-value theorem,  $g'(t)$  has at least  $n + 1$  distinct zeros in  $(a, b)$ . Similarly,  $g''(t)$  has at least  $n$  distinct zeros in  $(a, b)$  and  $g^{(n+1)}(t)$  has at least one zero in  $(a, b)$ , i.e., there exists  $\xi \in (a, b)$  such that

$$g^{(n+1)}(\xi) = f^{(n+1)}(\xi) - P^{(n+1)}(\xi) - \lambda\omega^{(n+1)}(\xi) = f^{(n+1)}(\xi) - \lambda(n+1)! = 0$$

i.e.,

$$f^{(n+1)}(\xi) - \frac{f(x) - P_n(x)}{\omega(x)}(n+1)! = 0$$

The proof is complete ■

*Example 3.* If  $f = \sin(x)$  is approximated by a polynomial of degree 9 that interpolates  $f$  at ten points in the interval  $[0, 1]$ , how large is the error on this interval ?

*Solution* By using Theorem 3.1,

$$|\sin(x) - P_n(x)| \leq \frac{f^{(10)}}{10!} \prod_{j=0}^9 (x - x_j) \leq \frac{1}{10!} < 2.8 * 10^{-7}$$

(v) *Hermite interpolation.* We have studied Lagrange interpolation. A Lagrange interpolation polynomial  $P_n(x)$  satisfies

$$P_n(x_j) = y_j.$$

Hermite interpolation  $p(x)$  will satisfy

$$p(x_j) = f(x_j) \quad p'(x_j) = f'(x_j), \quad j = 0, 1, \dots, n.$$

First we consider a simple case,  $n = 1$ . To satisfy the above four condition, we assume that

$$p(x) = a_0 + a_1x + a_2x^2 + a_3x^3$$

and satisfies

$$\begin{aligned} a_0 + a_1x_0 + a_2x_0^2 + a_3x_0^3 &= f(x_0) \\ a_1 + 2a_2x_0 + 3a_3x_0^2 &= f'(x_0) \\ a_0 + a_1x_1 + a_2x_1^2 + a_3x_1^3 &= f(x_1) \\ a_1 + 2a_2x_1 + 3a_3x_1^2 &= f'(x_1) \end{aligned}$$

where  $x_0 \neq x_1$ . In matrix form,

$$\begin{bmatrix} 1 & x_0 & x_0^2 & x_0^3 \\ 0 & 1 & 2x_0 & 3x_0^2 \\ 1 & x_1 & x_1^2 & x_1^3 \\ 0 & 1 & 2x_1 & 3x_1^2 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix} = \begin{bmatrix} f(x_0) \\ f'(x_0) \\ f(x_1) \\ f'(x_1) \end{bmatrix}.$$

It is easy to show that the system has a unique solution.

To construct a Hermite interpolation polynomial, we write the polynomial by

$$p(x) = \sum_{j=0}^n f(x_j)A_j(x) + \sum_{j=0}^n f'(x_j)B_j(x)$$

in analogy with the Lagrange formula. Here  $A_j(x)$  and  $B_j(x)$  are polynomials and satisfy the following conditions:

$$\begin{aligned} A_j(x_i) &= \delta_{ij}, & A'_j(x_i) &= 0 \\ B_j(x_i) &= 0, & B'_j(x_i) &= \delta_{ij} \end{aligned}$$

Let

$$L_j(x) = \prod_{i \neq j} \frac{x - x_i}{x_j - x_i}$$

We obtain the  $A_j(x)$  and  $B_j(x)$

$$\begin{aligned} A_j(x_i) &= [1 - 2(x - x_j)L'_j(x_j)]L_j^2(x) \\ B_j(x_i) &= (x - x_j)L_j^2(x) \end{aligned}$$

which can be verified easily.

*Example 4.* In the case  $n = 1$ , the Hermite interpolation polynomial can be given by

$$p(x) = f(x_0)A_0(x) + f'(x_0)B_0(x) + f(x_1)A_1(x) + f'(x_1)B_1(x)$$

where

$$\begin{aligned} A_0(x) &= [1 - 2(x - x_0)L'_0(x_0)]L_0^2(x) \\ A_1(x) &= (1 - 2(x - x_1)L'_1(x_1))L_1^2(x) \\ B_0(x) &= (x - x_0)L_0^2(x) \\ B_1(x) &= (x - x_1)L_1^2(x) \end{aligned}$$

and

$$L_0(x) = \frac{x - x_1}{x_0 - x_1}, \quad L_1(x) = \frac{x - x_0}{x_1 - x_0}, \quad L'_0(x) = \frac{1}{x_0 - x_1}, \quad L'_1(x) = \frac{1}{x_0 - x_1},$$

**Theorem 3.2.** Let  $x_0 < x_1 < \dots < x_n$  be in  $[a, b]$  and let  $f \in C^{2n+2}[a, b]$ . If  $p(x)$  is the Hermite interpolation polynomial of degree at most  $2n + 1$  such that

$$p(x_j) = f(x_j), \quad p'(x_j) = f'(x_j), \quad j = 0, 1, 2, \dots, n$$

then there exists  $\xi \in (a, b)$  such that

$$f(x) - p(x) = \frac{f^{(2n+2)}(\xi)}{(2n+2)!} \prod_{j=0}^n (x - x_j)^2.$$

*Proof.* Similarly to the proof of Theorem 3.1, we define

$$\omega(t) = \prod_{j=0}^n (t - x_j)^2 \quad g(t) = f(t) - p(t) - \lambda\omega(t)$$

where

$$\lambda = \frac{f(x) - p(x)}{\omega(x)}$$

which leads to  $g^{(2n+2)}(\xi) = 0$ . ■

(vi) *Inverse interpolation.* Let  $y = f(x)$  be a function defined by the data

x	0.5	0.8	1.1	1.4
y	-0.0625	-0.1024	0.1331	1.0976

Find a solution of  $f(x) = 0$ .

We may use the Lagrange interpolation to get the interpolation polynomial  $P_3(x)$  and then, solve the equation  $P_3(x) = 0$  by using a certain iterative algorithm, such as Newton's iteration to find an approximate zero of  $f(x)$ . The inverse interpolation provides a simpler way. The basic idea in the inverse is to construct an interpolation polynomial  $x = \bar{P}_3(y)$  based on the data and the approximate solution is

$$x = \bar{P}_3(0)$$

For the above example,

$$\begin{aligned}\bar{P}_3(y) &= \frac{(y + 0.1024)(y - 0.1331)(y - 1.0976)}{(-0.0625 + 0.1024)(-0.0625 - 0.1331)(-0.0625 - 1.0976)}^{0.5} \\ &+ \frac{(y + 0.0625)(y - 0.1331)(y - 1.0976)}{(-0.1024 + 0.0625)(-0.1024 - 0.1331)(-0.1024 - 1.0976)}^{0.8} \\ &+ \frac{(y + 0.0625)(y + 0.1024)(y - 1.0976)}{(0.1331 + 0.0625)(0.1331 + 0.1024)(0.1331 - 1.0976)}^{1.1} \\ &+ \frac{(y + 0.0625)(y + 0.1024)(y - 0.1331)}{(1.0976 + 0.0625)(1.0976 + 0.1024)(1.0976 - 0.1331)}^{1.4}\end{aligned}$$

and the approximate solution is

$$x = P_3(0).$$

### 3.2 Piecewise approximation.

(i) *Piecewise  $C^0$ -approximation.* When data is very large, high-order interpolation polynomials may be not best approximation. In engineering, one often uses *piecewise interpolation polynomials*.

*Example 5.*

x	0	1/4	1/2	3/4	1
y	1	$e^{1/4}$	$e^{1/2}$	$e^{3/4}$	$e$

- (a) Find a global Lagrange interpolation polynomial (degree of 4);
- (b) Find a piecewise linear interpolation polynomial;
- (c) Find a piecewise quadratic interpolation polynomial;

We solve problems below.

- (a). By using Lagrange formulation,

$$\begin{aligned}L_0(x) &= \frac{(x - 1/4)(x - 1/2)(x - 3/4)(x - 1)}{(0 - 1/4)(0 - 1/2)(0 - 3/4)(0 - 1)} \\ L_1(x) &= \frac{(x - 0)(x - 1/2)(x - 3/4)(x - 1)}{(1/4 - 0)(1/4 - 1/2)(1/4 - 3/4)(1/4 - 1)} \\ L_2(x) &= \frac{(x - 0)(x - 1/4)(x - 3/4)(x - 1)}{(1/2 - 0)(1/2 - 1/4)(1/2 - 3/4)(1/2 - 1)} \\ L_3(x) &= \frac{(x - 0)(x - 1/4)(x - 1/2)(x - 1)}{(3/4 - 0)(3/4 - 1/4)(3/4 - 1/2)(3/4 - 1)} \\ L_4(x) &= \frac{(x - 0)(x - 1/4)(x - 1/2)(x - 3/4)}{(1 - 0)(1 - 1/4)(1 - 1/2)(1 - 3/4)}\end{aligned}$$

and

$$P_4(x) = L_0(x) + L_1(x)e^{1/4} + L_2(x)e^{1/2} + L_3(x)e^{3/4} + L_4(x)e.$$

(b). In order to find the piecewise quadratic interpolation polynomial, we need to find two quadratic interpolation polynomial in the subintervals  $(0, 1/2)$  and  $(1/2, 1)$ , respectively. By

Lagrange formulation, we have

$$P_2^p = \begin{cases} \frac{(x-1/4)(x-1/2)}{(0-1/4)(0-1/2)} \cdot 1 + \frac{(x-0)(x-1/2)}{(1/4-0)(1/4-1/2)} e^{1/4} + \frac{(x-0)(x-1/4)}{(1/2-0)(1/2-1/4)} e^{1/2} & x \in [0, 1/2] \\ \frac{(x-3/4)(x-1)}{(1/2-3/4)(1/2-1)} e^{1/2} + \frac{(x-1/2)(x-1)}{(3/4-1/2)(3/4-1)} e^{3/4} + \frac{(x-1/2)(x-3/4)}{(1-1/2)(1-3/4)} e & x \in [1/2, 1] \end{cases}$$

(c) The piecewise linear interpolation polynomial is a linear interpolation at each of these four subintervals:  $(0, 1/4)$ ,  $(1/4, 1/2)$ ,  $(1/2, 3/4)$  and  $(3/4, 1)$  and is given by

$$P_1^p = \begin{cases} \frac{x-1/4}{0-1/4} + \frac{x-0}{1/4-0} e^{1/4} & x \in [0, 1/4] \\ \frac{x-1/2}{1/4-1/2} e^{1/4} + \frac{x-1/4}{1/2-1/4} e^{1/2} & x \in [1/4, 1/2] \\ \frac{x-3/4}{1/2-3/4} e^{1/2} + \frac{x-1/2}{3/4-1/2} e^{3/4} & x \in [1/2, 3/4] \\ \frac{x-1}{3/4-1} e^{3/4} + \frac{x-3/4}{1-3/4} e & x \in [3/4, 1] \end{cases}$$

(ii) *Piecewise  $C^1$ -approximation.* There are two types of  $C^1$ -approximation interpolations. One is based on the data  $\{x_i, y_i, y_i'\}$  and other one is on the data  $\{x_i, y_i\}$ . Here we only study the first case by using piecewise Hermite interpolation.

*Example 6.*

x	0	1/2	1
y	1	e	e <sup>2</sup>
y'	2	2e	2e <sup>2</sup>

The Piecewise Hermite interpolation is defined by

$$p(x) = \begin{cases} \left(1 - \frac{2(x-0)}{-1/2}\right) \left(\frac{x-1/2}{0-1/2}\right)^2 1 + (x-0) \left(\frac{x-0}{0-1/2}\right)^2 2 + \left(1 - \frac{2(x-1/2)}{1/2}\right) \left(\frac{x-1/2}{0-1/2}\right)^2 e \\ \quad + (x-1/2) \left(\frac{x-1/2}{1/2-0}\right)^2 2e & x \in [0, 1/2] \\ \left(1 - \frac{2(x-1/2)}{1/2-1}\right) \left(\frac{x-1}{1/2-1}\right)^2 e + (x-1/2) \left(\frac{x-1/2}{1/2-1}\right)^2 2 + \left(1 - \frac{2(x-1)}{1-1/2}\right) \left(\frac{x-1}{1/2-1}\right)^2 e^2 \\ \quad + (x-1) \left(\frac{x-1}{1-1/2}\right)^2 2e^2 & x \in [1/2, 1] \end{cases}$$

**Remarks:** Interpolation in multi-dimensional spaces.

*Example 6* Approximate the function  $f(x,y)$  in the small square  $\{(x,y) : 0 < x, y < h\}$  by a polynomial  $P(x)$  such that

$$P(0,0) = f(0,0), \quad P(0,h) = f(0,h), \quad P(h,0) = f(h,0), \quad P(h,h) = f(h,h).$$

*Solution* Let  $x_0 = y_0 = 0$ ,  $x_1 = y_1 = h$  and

$$\begin{aligned} L_0(x) &= \frac{x-x_1}{x_0-x_1} & L_1(x) &= \frac{x-x_0}{x_1-x_0} \\ L_0(y) &= \frac{y-y_1}{y_0-y_1} & L_1(y) &= \frac{y-y_0}{y_1-y_0} \end{aligned}$$

We define

$$P(x,y) = L_0(x)L_0(y)f(x_0,y_0) + L_0(x)L_1(y)f(x_0,y_1) + L_1(x)L_0(y)f(x_1,y_0) + L_1(x)L_1(y)f(x_1,y_1).$$

The polynomial  $P$ , which is a bilinear polynomial, satisfies conditions above. The above formula is called tensor product form. In general,

$$P(x, y) = \sum_{i=0}^n \sum_{j=0}^n L_i(x) L_j(y) f(x_i, y_j)$$

**Remarks** Interpolation in multi-dimensional spaces.

**3.3 Least squares (data fitting).** This is a non-interpolation approximation to the given data

x	$x_0$	$x_1$	.....	$x_n$
y	$y_0$	$y_1$	.....	$y_n$

(i) **Linear least squares.** A simple question here is to find a linear function  $p(x) = a + bx$  which is an approximation to the above data. The basic idea is to study the following minimization problem

$$\min_{a,b} \|f(x) - p(x)\|. \quad (4)$$

A special case is

$$\min_{a,b} \sum_{j=0}^n (y_j - (a + bx_j))^2.$$

In terms of theory in calculus, the solution satisfies the system

$$\begin{aligned} 2 \sum_{j=0}^n (y_j - (a + bx_j)) (-1) &= 0 \\ 2 \sum_{j=0}^n (y_j - (ax_j + b)) (-x_j) &= 0 \end{aligned}$$

and in a matrix form,

$$\begin{bmatrix} n+1 & \sum x_j \\ \sum x_j & \sum x_j^2 \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} \sum y_j \\ \sum x_j y_j \end{bmatrix}.$$

It is easy to extend this simple least squares approximation to the problem: find  $p(x) = a + bx + cx^2$  by

$$\min_{a,b,c} \sum_{j=0}^n (y_j - (a + bx_j + cx_j^2))^2.$$

The solution satisfies the system

$$\begin{bmatrix} n+1 & \sum x_j & \sum x_j^2 \\ \sum x_j & \sum x_j^2 & \sum x_j^3 \\ \sum x_j^2 & \sum x_j^3 & \sum x_j^4 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \end{bmatrix} = \begin{bmatrix} \sum y_j \\ \sum x_j y_j \\ \sum x_j^2 y_j \end{bmatrix}.$$

*Example 5.* Fit the following data by  $p_1(x) = a + bx$  and  $p_2(x) = a + bx + cx^2$ , respectively

x	0.5	0.8	1.1	1.4	1.7
y	-0.0625	-0.1024	0.1331	1.0976	1.300

*Solution.* (i) The solution  $(a, b)$  satisfies the system

$$\begin{bmatrix} 5 & 5.5 \\ 5.5 & 6.95 \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} 2.3658 \\ 3.77988 \end{bmatrix}.$$

The solution is

$$a = -0.966, \quad b = 1.308.$$

(ii) The system for  $(a, b, c)$  is

$$\begin{bmatrix} 5 & 5.5 & 6.95 \\ 5.5 & 6.95 & 9.625 \\ 6.95 & 9.625 & 14.1299 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \end{bmatrix} = \begin{bmatrix} 2.3658 \\ 3.77988 \\ 5.988 \end{bmatrix}.$$

The solution is

$$a = 1.3318, \quad b = 1.7387, \quad c = 2.4656$$

A general least squares problem is to find an approximation in a subspace with the basis functions  $\phi_i(x)$ ,  $i = 1, 2, \dots, m$ . The solution in the subspace can be represented by  $p(x) = \sum \alpha_j \phi_j(x)$  and a least squares problem is described by

$$\min_{\alpha} \sum_{j=0}^n \left( y_j - \sum_{i=1}^m \alpha_i \phi_i(x) \right)^2.$$

The solution satisfies the system

$$\begin{bmatrix} \sum_j (\phi_1(x_j))^2 & \sum_j \phi_2(x_j) \phi_1(x_j) & \sum_j \phi_m(x_j) \phi_1(x_j) \\ \sum_j \phi_1(x_j) \phi_2(x_j) & \sum_j (\phi_2(x_j))^2 & \sum_j \phi_m(x_j) \phi_2(x_j) \\ \dots & \dots & \dots \\ \sum_j \phi_1(x_j) \phi_m(x_j) & \sum_j \phi_2(x_j) \phi_m(x_j) & \sum_j (\phi_m(x_j))^2 \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_m \end{bmatrix} = \begin{bmatrix} \sum \phi_1(x_j) y_j \\ \sum \phi_2(x_j) y_j \\ \vdots \\ \sum \phi_m(x_j) y_j \end{bmatrix}.$$

**(ii) Nonlinear least squares.**

*Example 6.* Fit the following data by  $p(x) = ae^{bx}$ .

x	0.5	0.8	1.1	1.4	1.7
y	-0.0625	-0.1024	0.1331	1.0976	1.300

*Solution.* The problem can be described by

$$\min_{a,b} \sum (y_j - ae^{bx_j})^2.$$

The solution satisfies

$$\begin{aligned} 2 \sum (y_j - ae^{bx_j}) e^{bx_j} &= 0 \\ 2 \sum (y_j - ae^{bx_j}) ax_j e^{bx_j} &= 0 \end{aligned} \tag{5}$$

One needs to solve the system of nonlinear equations by a certain numerical method, such as Newton's iteration.

(iii) *Over-determined system.* We consider a simple example:

$$\begin{aligned}2x_1 + x_2 &= 1 \\x_1 + 3x_2 &= 2 \\x_1 - x_2 &= 0.\end{aligned}$$

In this system, there are three equations and two unknowns. The system is over-determined and the system does not have a solution. A least squares problem is described by

$$\min_{x_1, x_2} (1 - 2x_1 - x_2)^2 + (2 - x_1 - 3x_2)^2 + (-x_1 + x_2)^2.$$

The solution satisfies

$$\begin{aligned}-2(1 - 2x_1 - x_2) - (2 - x_1 - 3x_2) - (-x_1 + x_2) &= 0 \\-(1 - 2x_1 - x_2) - 3(2 - x_1 - 3x_2) + (-x_1 + x_2) &= 0\end{aligned}$$

and the solution is

$$x_1 = 8/19 \quad x_2 = 7/19.$$

Finally we consider a general over-determined system

$$Ax = b$$

where  $A$  is an  $n \times m$  matrix with  $n > m$ . The system may not have a solution. The least squares problem is described by

$$\min_x \|Ax - b\|. \tag{6}$$

If we choose 2-norm,

$$\|Ax - b\|_2^2 = (Ax - b)^T(Ax - b) = x^T A^T Ax - b^T Ax - x^T A^T b + b^T b.$$

The solution satisfies the system

$$A^T Ax = A^T b \tag{7}$$

### 3.4 Numerical integration.

In this section, we study numerical methods for the evaluation of the integral

$$\int_a^b f(x)dx.$$

**(i) Numerical integration based on interpolation.** The basic idea is to approximate  $f(x)$  by its interpolation function  $p(x)$  to get the numerical integration

$$\int_a^b f(x)dx \approx \int_a^b p(x)dx$$

We have studied many different interpolations in previous sections.

Let  $P_n(x)$  be the Lagrange interpolation polynomial of degree  $n$  on the nodes  $\{x_j\}_{j=0}^n$ . Then

$$P_n(x) = \sum_{j=0}^n L_j(x)f(x_j)$$

and therefore,

$$\int_a^b f(x)dx \approx \int_a^b P_n(x)dx = \sum_{j=0}^n \alpha_j f(x_j)$$

where

$$\alpha_j = \int_a^b L_j(x)dx .$$

Such a quadrature rule is called *Newton-Cotes quadrature rule*

Case I: we consider a simple case:

$$n = 1, \quad x_0 = a, \quad x_1 = b .$$

In this case

$$L_0(x) = \frac{x_1 - x}{x_1 - x_0}, \quad L_1(x) = \frac{x_0 - x}{x_0 - x_1} .$$

Consequently,

$$\alpha_0 = \int_a^b L_0(x)dx = \frac{b-a}{2} = \int_a^b L_1(x)dx = \alpha_1$$

The corresponding quadrature rule is

$$\int_a^b f(x)dx \approx \frac{b-a}{2}(f(a) + f(b))$$

which is called **trapezoid rule**.

Case II: we consider the case

$$n = 2, \quad x_0 = a, \quad x_1 = \frac{a+b}{2}, \quad x_2 = b .$$

In this case,

$$L_0(x) = \frac{(x-x_1)(x-x_2)}{(x_0-x_1)(x_0-x_2)} \quad L_1(x) = \frac{(x-x_0)(x-x_2)}{(x_1-x_0)(x_1-x_2)} \quad L_2(x) = \frac{(x-x_0)(x-x_1)}{(x_2-x_0)(x_2-x_1)}$$

and

$$\int_a^b L_0(x)dx = \frac{b-a}{6}, \quad \int_a^b L_1(x)dx = \frac{4(b-a)}{6}, \quad \int_a^b L_2(x)dx = \frac{b-a}{6} .$$

The corresponding Newton-Cotes quadrature rule is

$$\int_a^b f(x)dx \approx \frac{b-a}{6} \left( f(a) + 4f\left(\frac{b+a}{2}\right) + f(b) \right) .$$

This is called **Simpson's rule**.

Finally we study the error estimate of the above two quadrature rules. The error for Newton Cotes formulas can be given by

$$\text{error} = \frac{1}{(n+1)!} \int_a^b \prod_{i=0}^n (x-x_i) f^{(n+1)}(\xi) dx$$

For the linear approximation,

$$f(x) = P_1(x) + \frac{(x-a)(x-b)}{2} f''(\xi).$$

Then

$$\int_a^b f(x)dx = \int_a^b P_1(x)dx + \int_a^b \frac{(x-a)(x-b)}{2} f''(\xi)dx$$

By the Mean-Value theory,

$$\text{error} = \int_a^b f(x)dx - \frac{b-a}{2}(f(a) + f(b)) = \frac{-(b-a)^3}{12} f''(\eta)$$

Similarly, for the Simpson's rule

$$\text{error} = \int_a^b f(x)dx - \frac{b-a}{6} \left( f(a) + 4f\left(\frac{b+a}{2}\right) + f(b) \right) = \int_a^b \frac{(x-a)(x-(a+b)/2)(x-b)}{6} f'''(\xi)dx$$

We can prove that

$$\int_a^b \frac{(x-a)(x-(a+b)/2)(x-b)}{6} f'''(\xi)dx = -\frac{1}{90} \left( \frac{b-a}{2} \right)^5 f^{(4)}(\eta).$$

**(ii) Composite Newton-Cotes quadrature rules.** Let

$$a = x_0 < x_1 < \dots < x_n = b$$

and

$$\int_a^b f(x)dx = \sum_{i=1}^n \int_{x_{i-1}}^{x_i} f(x)dx$$

The Composite Newton-Cotes quadrature rule is to use the Newton-Cotes quadrature rule for the integral in each subinterval. The Composite trapezoid rule can be given by

$$\int_a^b f(x)dx \approx \sum_{i=1}^n (x_i - x_{i-1}) \frac{f(x_{i-1}) + f(x_i)}{2}.$$

For a uniform mesh,  $x_i - x_{i-1} = h$ ,

$$\int_a^b f(x)dx \approx \frac{h}{2} \left( f(x_0) + 2 \sum_{i=1}^{n-1} f(x_i) + f(x_n) \right).$$

For the Simpson's rule, we assume  $n = 2m$  and

$$\int_a^b f(x)dx = \sum_{i=1}^m \int_{x_{2i-2}}^{x_{2i}} f(x)dx$$

Applying the Simpson's rule on each subinterval gives the composite Simpson's rule

$$\int_a^b f(x)dx \approx \sum_{i=1}^m \frac{x_{2i} - x_{2i-2}}{6} (f(x_{2i-2}) + 4f(x_{2i-1}) + f(x_{2i}))$$

For a uniform mesh,  $x_i - x_{i-1} = h$ ,

$$\int_a^b f(x)dx \approx \frac{h}{3} \left( f(x_0) + 2 \sum_{i=2}^m f(x_{2i-2}) + 4 \sum_{i=1}^m f(x_{2i-1}) + f(x_n) \right).$$

Error estimates can be obtained from previous analysis at each subinterval. Then the error for composite trapezoid rule is

$$\int_a^b f(x)dx - \frac{h}{2} \left( f(x_0) + 2 \sum_{i=1}^{n-1} f(x_i) + f(x_n) \right) = \sum_{i=1}^n \frac{-h^3}{12} f''(\xi_i) = \frac{-h^2(b-a)}{12} f''(\xi)$$

and the error for Simpson's rule is

$$\begin{aligned} \int_a^b f(x)dx - \frac{h}{3} \left( f(x_0) + 2 \sum_{i=2}^m f(x_{2i-2}) + 4 \sum_{i=1}^m f(x_{2i-1}) + f(x_n) \right) \\ = \sum_{i=1}^m -\frac{1}{90} h^5 f^{(4)}(\eta_i) = -\frac{b-a}{180} h^4 f^{(4)}(\eta). \end{aligned} \quad (8)$$

*Example 7* Calculate the integral

$$\int_0^1 e^{x^2} dx$$

by using trapezoid rule, Simpson's rule, composite trapezoid rule and Simpson's rule with two subintervals.

*Solution* By trapezoid rule

$$\int_0^1 e^{x^2} dx \approx \frac{1}{2}(1 + e) = 1.859140914.$$

By Simpson's rule,

$$\int_0^1 e^{x^2} dx \approx \frac{1}{6}(1 + 4 * e^{1/4} + e) = 1.475730583.$$

By trapezoid rule with two subintervals,  $h = 1/2$

$$\int_0^1 e^{x^2} dx \approx \frac{1/2}{2}(1 + 2e^{1/4} + e) = 1.571583165.$$

By Simpson's rule with two subintervals,  $h = 1/2$ ,

$$\int_0^1 e^{x^2} dx \approx \frac{1/2}{6}(1 + 4e^{1/16} + 2e^{1/4} + 4e^{9/16} + e) = 1.46371076.$$

The exact solution is 1.46265177159149.

**Remarks** A more general formula is

$$\int_a^b \omega(x) f(x) dx = \sum_{i=0}^n \alpha_i f(x_i)$$

where  $\omega(x)$  is weight function and

$$\alpha_i = \int_a^b \omega(x) L_i(x) dx.$$

**(iii) Gaussian quadrature rules.** Based on the quadrature rules in the above section, we can see the general form

$$\int_a^b f(x) dx = \sum_i \alpha_i f(x_i).$$

The question is whether we can choose  $\alpha_i$  and  $x_i$  to get higher-order accuracy. From polynomial interpolation, we have

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

Then the formula is exact when  $f$  is a polynomial of degree  $\leq n$ . Such a formula is said to be of  $n$ -order algebraic accuracy. So Newton-Cotes rule on  $n$  nodes is of at least  $n - 1$ -order algebraic accuracy. Based on the concept, we can choose  $\alpha_i$  and  $x_i$  to create a quadrature rule so that the rule is exact for polynomials of degree as high as possible (or of the highest order algebraic accuracy). Such a quadrature rule is called *Gaussian quadrature rule*

*Example 8* Find  $x_1$  and  $\alpha_1$  such that the following quadrature rules are exact for polynomials of degree as high as possible.

$$\int_{-1}^1 f(x) dx = \alpha_1 f(x_1)$$

and

$$\int_{-1}^1 f(x) dx = \alpha_1 f(x_1) + \alpha_2 f(x_2).$$

*Solution.* (i) Choosing  $f = 1$  and  $f = x$ , we obtain

$$\begin{aligned} 2 &= \alpha_1 \\ 0 &= \alpha_1 x_1 \end{aligned}$$

Solving the system gives

$$\alpha_1 = 2, \quad x_1 = 0.$$

We obtain the quadrature rule

$$\int_{-1}^1 f(x) dx \approx 2f(0)$$

which is called **mid-point rule**. Thus the mid-point rule is of first-order algebraic accuracy.

(ii) Choosing  $f = 1, x, x^2, x^3$ , we have

$$\begin{aligned} 2 &= \alpha_1 + \alpha_2 \\ 0 &= \alpha_1 x_1 + \alpha_2 x_2 \\ \frac{2}{3} &= \alpha_1 x_1^2 + \alpha_2 x_2^2 \\ 0 &= \alpha_1 x_1^3 + \alpha_2 x_2^3 \end{aligned}$$

From the second and fourth equations, we get

$$\det \begin{bmatrix} x_1 & x_2 \\ x_1^3 & x_2^3 \end{bmatrix} = 0$$

which leads to

$$x_1 = \pm x_2.$$

For  $x_1 = -x_2$ , we obtain from the second equation

$$\alpha_1 - \alpha_2 = 0$$

which together with the first equation gives

$$\alpha_1 = \alpha_2 = 1$$

Substituting them into the third equation, we have

$$x_1 = -1/\sqrt{3}, \quad x_2 = 1/\sqrt{3}.$$

Here we call  $x_1$  and  $x_2$  *Gaussian points*. Let  $P_2(x)$  be the Lagrange interpolation polynomial on these two Gaussian points. A special Newton-Cotes rule is defined by

$$\int_{-1}^1 f(x)dx \approx \int_{-1}^1 P_1(x)dx = f(x_1) \int_{-1}^1 L_0(x)dx + f(x_2) \int_{-1}^1 L_1(x)dx$$

A straightforward calculation gives

$$\int_{-1}^1 L_0(x)dx = 1 = \alpha_1, \quad \int_{-1}^1 L_1(x)dx = 1 = \alpha_2.$$

Then the Gaussian quadrature rule is the Newton-Cotes formula on Gaussian points.

We can extend the quadrature rules to general integrals

$$\begin{aligned} \int_a^b f(x)dx &= \int_{-1}^1 f\left(\frac{b-a}{2}t + \frac{a+b}{2}\right) \frac{b-a}{2}dt \\ &\approx \frac{b-a}{2} \left( f\left(\frac{-(b-a)}{2\sqrt{3}} + \frac{a+b}{2}\right) + f\left(\frac{(b-a)}{2\sqrt{3}} + \frac{a+b}{2}\right) \right). \end{aligned}$$

*Example 9* Find  $x_1, x_2$  and  $\alpha_1, \alpha_2$  such that the following quadrature rule has the highest algebraic accuracy

$$\int_{-1}^1 \sqrt{1-x^2} f(x)dx = \alpha_1 f(x_1) + \alpha_2 f(x_2).$$

**Orthogonal polynomials.** Here we shall introduce orthogonal polynomials to generate Gaussian points.

Let  $\omega(x) > 0$  in  $(a, b)$  and  $p(x), q(x)$  be two polynomials. It is said that  $p(x)$  is  $\omega$ -orthogonal to  $q(x)$  (or orthogonal to  $q(x)$  with respect to the weight function  $\omega$ ) if

$$\int_a^b \omega(x)p(x)q(x)dx = 0$$

Mathematically we can prove that

- For a given positive weight function  $\omega(x)$ , there exist a sequence of orthogonal polynomials,  $p_k(x)$ ,  $k = 0, 1, 2, \dots$ , where  $p_k(x)$  is a polynomial of degree  $k$  and

$$\int_a^b \omega(x)p_m(x)p_n(x)dx = 0 \quad n \neq m$$

- $p_k(x)$  has  $k$  simple and real roots.
- If  $x_1, x_2, \dots, x_k$  are roots of  $p_k(x)$ , then the following quadrature rule is exact for any polynomials of degree  $\leq 2k - 1$ .

$$\int_a^b \omega(x)f(x)dx \approx \sum_{i=1}^k \alpha_i f(x_i) \tag{9}$$

where

$$\alpha_i = \int_a^b \omega(x)L_i(x)dx \quad L_i(x) = \prod_{j \neq i} \frac{x - x_j}{x_i - x_j}, \quad i = 1, 2, \dots, k.$$

*Proof.* (only the last assertion) Let  $f$  be a polynomial of degree  $\leq 2k - 1$ . Divide  $f$  by  $p_k$ , obtaining a quotient  $q$  and remainder  $r(x)$ . Then

$$f(x) = p_k(x)q(x) + r(x).$$

where  $r(x)$  is a polynomial of degree  $k - 1$ . Consequently,

$$f(x_i) = p_k(x_i)q(x_i) + r(x_i) = r(x_i).$$

Since the formula (9) is a Newton-Cotes rule on Gaussian points, this formula is exact for any polynomials of degree  $\leq k - 1$ . Then We have

$$\int_a^b \omega(x)f(x)dx = \int_a^b \omega(x)p_k(x)q(x)dx + \int_a^b \omega(x)r(x)dx = \int_a^b \omega(x)r(x)dx$$

and

$$\int_a^b \omega(x)f(x)dx = \int_a^b \omega(x)r(x)dx = \sum_{i=1}^k \alpha_i r(x_i) = \sum_{i=1}^k \alpha_i f(x_i).$$

- If  $f \in C^{2n}[a, b]$ , the error of Gaussian quadrature rule is given by

$$\int_a^b \omega(x)f(x)dx = \sum_{i=1}^n \alpha_i f(x_i) + \frac{f^{(2n)}(\xi)}{(2n)!} \int_a^b \left( \prod_{i=1}^n (x - x_i) \right)^2 \omega(x)dx.$$

*Example 10* Calculate the integral in example 7 by using one-point Gaussian quadrature rule and two-point Gaussian quadrature rule, respectively.

*Solution* By the one-point rule,

$$\int_0^1 e^{x^2} dx \approx e^{1/4} = 1.284025417$$

and by the two-point rule

$$\int_0^1 e^{x^2} dx \approx \frac{1}{2} \left( e^{(0.5-1/2\sqrt{3})^2} + e^{(0.5+1/2\sqrt{3})^2} \right) = 1.454167889$$

It is easy to generate composite Gaussian quadrature rules by combining the concepts of the Gaussian rule and composite rule. Let

$$a = x_0 < x_1 \dots < x_n = b$$

be a uniform partition,  $h = x_j - x_{j-1}$ . Then

$$\int_a^b f(x) dx = \sum_{j=1}^n \int_{x_{j-1}}^{x_j} \omega(x) f(x) dx \approx \sum_{j=1}^n \left( \sum_{i=1}^k \frac{h}{2} \alpha_i f(x_{ij}) \right)$$

*Example 11* Calculate the integral in example 7 by using composite one-point Gaussian rule and composite two-point Gaussian rule with two subintervals, respectively.

*Solution* By the composite one-point rule,

$$\int_0^1 e^{x^2} dx \approx 0.5(e^{1/16} + e^{9/16}) = 1.409774558$$

and by the composite two-point rule,

$$\int_0^1 e^{x^2} dx \approx 0.25 \left( e^{(0.25-0.5/2\sqrt{3})^2} + e^{(0.25+0.5/2\sqrt{3})^2} + e^{(0.75-0.5/2\sqrt{3})^2} + e^{(0.75+0.5/2\sqrt{3})^2} \right) = 1.461950972$$

**3.5 Numerical differentiation.** When a function  $y = f(x)$  is defined by discrete data, the question is how to calculate its derivatives.

Let  $p(x)$  be an approximation to  $f(x)$  based on given data.

$$f(x) \approx p(x)$$

where  $p(x)$  could be Lagrange interpolation polynomial, Hermite interpolation polynomial, piecewise interpolation polynomial, or approximation based on least squares. Then we often approximate its derivatives by

$$f'(x) \approx p'(x), \quad f''(x) \approx p''(x).$$

If  $p(x)$  is the Lagrange interpolation polynomial on the data  $\{x_j, y_j\}_{j=0}^n$ . Then

$$f(x) \approx P_n(x) = \sum_{j=0}^n L_j(x) y_j$$

and

$$f'(x) \approx P'_n(x) = \sum_{j=0}^n L'_j(x) y_j.$$

The error is given by

$$f'(x) = \sum_{j=0}^n L'_j(x)y_j + \frac{1}{(n+1)!} \left( \prod_{i=0}^n (x-x_i) f^{(n+1)}(\xi) \right)'.$$

Here we consider several simple cases.

Case I:  $n = 1$  (linear approximation).

$$L_0(x) = \frac{x-x_1}{x_0-x_1} \quad L_1(x) = \frac{x-x_0}{x_1-x_0}.$$

Then

$$L'_0(x) = \frac{1}{x_0-x_1} \quad L'_1(x) = \frac{1}{x_1-x_0}$$

and

$$f'(x) \approx P'_1(x) = \frac{y_0}{x_0-x_1} + \frac{y_1}{x_1-x_0} = \frac{y_1-y_0}{x_1-x_0}.$$

Case II:  $n = 2$  (quadratic approximation).

$$L_0(x) = \frac{(x-x_1)(x-x_2)}{(x_0-x_1)(x_0-x_2)} \quad L_1(x) = \frac{(x-x_0)(x-x_2)}{(x_1-x_0)(x_1-x_2)} \quad L_2(x) = \frac{(x-x_0)(x-x_1)}{(x_2-x_0)(x_2-x_1)}.$$

Then

$$L'_0(x) = \frac{2x-x_1-x_2}{(x_0-x_1)(x_0-x_2)} \quad L'_1(x) = \frac{2x-x_0-x_2}{(x_1-x_0)(x_1-x_2)} \quad L'_2(x) = \frac{2x-x_0-x_1}{(x_2-x_0)(x_2-x_1)}.$$

For a uniform mesh,  $h = x_1 - x_0 = x_2 - x_1$ ,

$$L'_0(x) = \frac{2(x-x_{3/2})}{2h^2} \quad L'_1(x) = \frac{2(x-x_1)}{-h^2} \quad L'_2(x) = \frac{2(x-x_{1/2})}{2h^2}$$

and

$$f'(x) \approx y_0 \frac{2(x-x_{3/2})}{2h^2} y_1 \frac{2(x-x_1)}{-h^2} + y_2 \frac{2(x-x_{1/2})}{2h^2}$$

where  $x_{1/2} = (x_0 + x_1)/2$  and  $x_{3/2} = (x_1 + x_2)/2$ . Also, we have

$$f''(x) \approx P'_2(x) = y_0 \frac{2}{2h^2} + y_1 \frac{2}{-h^2} + y_2 \frac{2}{2h^2} = \frac{y_0 - 2y_1 + y_2}{h^2}.$$

It is easy to get the general error estimates

$$f'(x) = P'_1(x) + \frac{1}{2} ((x-x_0)(x-x_1)f''(\xi))'$$

and

$$\begin{aligned} f'(x) &= P'_2(x) + \frac{1}{6} ((x-x_0)(x-x_1)(x-x_2)f'''(\xi))' \\ f''(x) &= P''_2(x) + \frac{1}{6} ((x-x_0)(x-x_1)(x-x_2)f'''(\xi))'' \end{aligned} \quad (10)$$

For a uniform mesh and small  $h$ ,

$$f'(x) - P_1'(x) = O(h)$$

$$f'(x) = P_2'(x) + O(h^2) \quad f''(x) = P_2''(x) + O(h).$$

Also we can use Taylor expansion to derive the order of an approximation. For

$$f'(x_0) \approx P_1'(x_0) = \frac{y_1 - y_0}{h} = \frac{f(x_1) - f(x_0)}{h},$$

by using Taylor formula,

$$f(x_1) = f(x_0) + hf'(x_0) + \frac{h^2}{2}f''(\xi)$$

and therefore,

$$\frac{f(x_1) - f(x_0)}{h} = f'(x_0) + \frac{h}{2}f''(\xi).$$

The error of the linear approximation is

$$f'(x_0) - \frac{f(x_1) - f(x_0)}{h} = \frac{h}{2}f''(\xi).$$

This is a first-order approximation. However, it is possible to get a higher-order approximation at certain points. We consider

$$f'(x_{1/2}) \approx P_1'(x_{1/2}) = \frac{y_1 - y_0}{h} = \frac{f(x_1) - f(x_0)}{h}.$$

By using Taylor's expansion,

$$\begin{aligned} f(x_1) &= f(x_{1/2}) + \frac{h}{2}f'(x_{1/2}) + \frac{(h/2)^2}{2}f''(x_{1/2}) + \frac{(h/2)^3}{6}f'''(\xi_1) \\ f(x_0) &= f(x_{1/2}) - \frac{h}{2}f'(x_{1/2}) + \frac{(h/2)^2}{2}f''(x_{1/2}) - \frac{(h/2)^3}{6}f'''(\xi_2). \end{aligned} \quad (11)$$

We obtain

$$\frac{f(x_1) - f(x_0)}{h} = f'(x_{1/2}) + \frac{(h/2)^3}{6}(f'''(\xi_1) + f'''(\xi_2)) = f'(x_{1/2}) + \frac{h^3}{24}f'''(\xi) = f'(x_{1/2}) + O(h^2)$$

which leads to a second-order approximation.

Similarly we can prove that

$$f''(x_1) = \frac{y_0 - 2y_1 + y_2}{h^2} + O(h^2).$$