SCI 4010 COMPUTATIONAL METHODS

MSc Course on Scientific Computing
Hong Kong Baptist University

Contents

Chapter 2

Polynomial Approximation

2.1 Introduction

For many purposes we wish to approximate a given function by a simpler function such as Polynomial. The advantages for using a polynomial are the following:

- (1) Easily evaluated in terms of addition (or substraction) & multiplication;
- (2) Easily integrated or differentiated.

However, there are also disadvantages for the polynomial approximation. They are mainly the following:

- (1) Oscillation when degree of the polynomial is high;
- (2) Always $\lim_{x\to\infty} |P(x)| = \infty$.

Both of the above two difficulties can be overcome by using Low Degree Piecewise polynomial.

2.1.1 Evaluation of Polynomial

We evaluate a polynomial $f(x) = a_0x^n + a_1x^{n-1} + \cdots + a_n$, with

Algorithm I: $x \uparrow k \Rightarrow x_k, \qquad k = 1, \ldots, n;$
$$\sum_{k=0}^{n} a_{n-k} * x_k \Rightarrow f. \qquad \frac{n(n-1)}{2} \text{ multiplications}$$

Algorithm II: $x \Rightarrow x_1; \quad x * x_k \Rightarrow x_{k+1}, \quad k = 1, \ldots, n-1;$
$$\sum_{k=0}^{n} a_{n-k} * x_k \Rightarrow f. \qquad (2n-1) \text{ multiplications}$$

Algorithm III: $a_0 * x \Rightarrow f; \quad (f + a_k) * x \Rightarrow f, \qquad k = 1, \ldots, n-1;$
$f + a_n \Rightarrow f. \qquad n$ multiplications

2.1.2 Taylor Expansion

$$
\begin{aligned}
f(x) &= T_n(x) + R_n(x) \\
T_n(x) &= f(x_0) + f'(x_0)(x-x_0) + \frac{1}{2!}f''(x_0)(x-x_0)^2 \\
&\quad + \cdots + \frac{1}{n!}f^{(n)}(x_0)(x-x_0)^n \\
R_n(x) &= \frac{f^{(n+1)}(\xi_x)}{(n+1)!}(x-x_0)^{n+1}, \quad \xi_x \text{ an unknown point between } x_0 \ \& \ x
\end{aligned}
$$

Although there are many advantages for using Taylor expansion, the following are the main difficulties for its application to practical problems:

- (1) High smoothness requirement
- (2) No convergent guarantee even if $f(x)$ infinitely differentiable
- (3) Can not be used when only discrete function values given, which is often the case in application.

2.2 Interpolation

2.2.1 Definition

A Taylor series approximation is found by utilizing the vaules of f and its derivatives at some point x_0. Another approach to finding a polynomial approximation uses only values of f.

Definition 2.1 Assume that we have $(n+1)$ points (x_i, y_i), where x_i are distinct real number and

$$y_i = f(x_i), \qquad i = 0, 1, \ldots, n$$

We wish to find a polynomial $p_n(x)$ such that

$$p_n(x_i) = y_i, \qquad i = 0, 1, \ldots, n$$

$p_n(x)$ is called the Interpolating Polynomial with respect to data $\{(x_i, y_i)\}_{i=0}^n$.

Example : If $n = 1$, $\quad p_1(x) = y_1 + \dfrac{y_1 - y_0}{x_1 - x_0}(x - x_1)$

2.2.2 Existence

Let $p_n(x) = a_n + a_{n-1}x + a_{n-2}x^2 + \cdots + a_0 x^n$. From the definition of interpolation, the coefficients $a_n, a_{n-1}, \ldots, a_0$ satisfy the following linear system of equations:

$$\begin{array}{rcccl} p_n(x_0) & = & a_n + a_{n-1}x_0 + \cdots + a_0 x_0^n & = & y_0 \\ p_n(x_1) & = & a_n + a_{n-1}x_1 + \cdots + a_0 x_1^n & = & y_1 \\ \vdots & & \vdots & & \vdots \\ p_n(x_n) & = & a_n + a_{n-1}x_n + \cdots + a_0 x_n^n & = & y_n \end{array}$$

The coefficient matrix (the a_i are unknowns)

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

is called Vandermonde matrix and

$$\det A = \prod_{i<j} (x_j - x_i)$$

Because $x_0, \ldots, x_n$ are distinct, $\det A \neq 0$. Hence the solution $\{a_0, \ldots, a_n\}$ is uniquely existent. However, we never construct the interpolating polynomial by using this linear system.

2.2.3 Lagrange Form

Given $(n+1)$ distinct real numbers: $x_0, x_1, \ldots, x_n$, let

$$l_i(x) = \frac{(x - x_0)\cdots(x - x_{i-1})(x - x_{i+1})\cdots(x - x_n)}{(x_i - x_0)\cdots(x_i - x_{i-1})(x_i - x_{i+1})\cdots(x_i - x_n)} \qquad i = 0, 1, \ldots, n$$

It is easy to check that

$$l_i(x_j) = \delta_{ij} = \begin{cases} 1, & i = j \\ 0, & i \neq j \end{cases}$$

Thus, by the definition, we obtain the interpolating polynomial as following

$$p_n(x) = \sum_{i=0}^{n} f(x_i) l_i(x)$$

which is called Lagrange form of interpolation or Lagrange polynomial.

2.2.4 Convergence and Stability

Theorem 2.1 Assume that $f(x)$ have $(n+1)$ continouous derivatives on $[a,b]$ and $x_0, x_1, \ldots, x_n$ be distinct nodes on $[a,b]$. Then the remainder of the Lagrange interpolation polynomial p_n is given by

$$R_n(x) = f(x) - p_n(x) = \frac{f^{(n+1)}(\xi_x)}{(n+1)!}(x-x_0)(x-x_1)\cdots(x-x_n)$$

for $a \le x \le b$, where ξ_x is an unknown point between the minimum and maximum of $x_0, x_1, \ldots, x_n$ and x.

In general, the remainder can not be used to give a very precise bound on the error. However, it can be useful in providing various insights into the errors that are produced.

Behaviour of the Error: When we consider the remainder, the polynomial

$$\Psi_n(x) = (x-x_0)(x-x_1)\cdots(x-x_n)$$

is the most important quantity in determining the behaviour when the nodes are evenly spaced. For larger values of n, say $n \ge 5$, the vaules of $\Psi_n(x)$ change rapidly in the interval $[x_0, x_n]$. The values in $[x_0, x_1]$ and $[x_{n-1}, x_n]$ become much larger than the values in the middle of $[x_0, x_n]$. The graph of $\Psi_n(x)$, $n = 6$, is given as following:

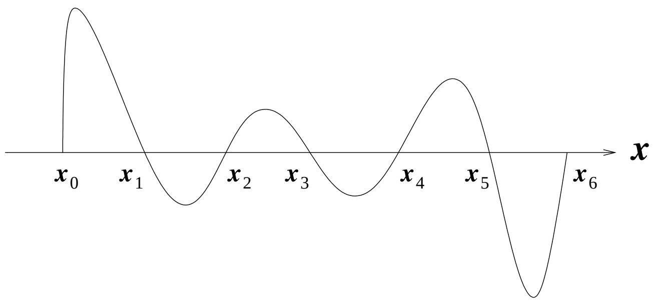


Figure: y=Ψ_b (x)

As n increases, this disparity also increases. Moreover, the interpolation error in some intervals near the end points is usually very large for large n.

It can be proved by using Theorem 2.1 that $\lim_{n\to\infty} p_n(x) = f(x)$ uniformly on some interval $[a,b]$ when $f(z)$ is analytic in a sufficient large domain in the complex plane. Otherwise there is no guarentee of convergence even if $f(x)$ is infinitely differentiable.

Example : (Bernstein):

$$f(x) = |x|, \quad -1 \le x \le 1$$

Take equidistant interpolating nodes, namely,

$$x_j = 2\left(\frac{j}{n} - \frac{1}{2}\right), \qquad j = 0, 1, \ldots, n$$

to construct the interpolation $p_n(x)$. Then the sequence $\{p_n(x)\}$ converges only at three points $x = -1,\ 0,\ 1$ and $\lim_{n\to\infty} p_n(x) = \infty$, at other points.

Example : (Runge):

$$f(x) = \frac{1}{1+x^2}$$

$f(x)$ is infinitely differentiable on $(-\infty, +\infty)$ but not analytic in $|z| < r$ when $r > 1$.
Take equidistant interpolating nodes on $[-5, 5]$, namely,

$$x_j = 10\left(\frac{j}{n} - \frac{1}{2}\right), \qquad j = 0, 1, \ldots, n$$

Then the interpolation sequence $p_n(x)$ is divergent outside the interval $|x| \leq 3.63$.

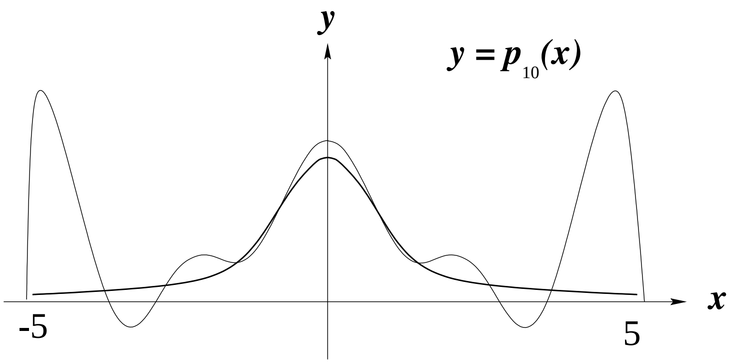


Figure: The interpolartion to $\frac{1}{(1+x^2)}$

2.3 Piecewise Polynomials

2.3.1 Piecewise Polynomials

Due to numerical instability (mainly due to rounding errors) and no guarentee of convergence, high order interpolation is not appropriate for function approximation. Since the early sixties, the subject of piecewise polynomial functions has become increasingly popular, especially spline functions. They are used in computer graphics, data fitting, numerical integration and numerical solution for differential and integral equations.

The advantages of Piecewise Polynomial Interpolation are the following: stable, convergent and easy programing.

Definition 2.2 A function $q(x)$ is called a piecewise polynomial function on $[a, b]$, if there is a subdivision on $[a, b]$

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

such that $q(x)$ is a polynomial on each subinterval $[x_i, x_{i+1}]$, $i = 0, 1, \dots, n-1$, i.e.,

$$q(x) = \begin{cases} p_1(x), & x_0 < x < x_1 \\ p_2(x), & x_1 < x < x_2 \\ \vdots & \vdots \\ p_n(x), & x_{n-1} < x < x_n \end{cases}$$

Step function is a piecewise polynomial of zero degree.

2.3.2 Piecewise Linear Interploation

Given interpolating nodes:

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

and data

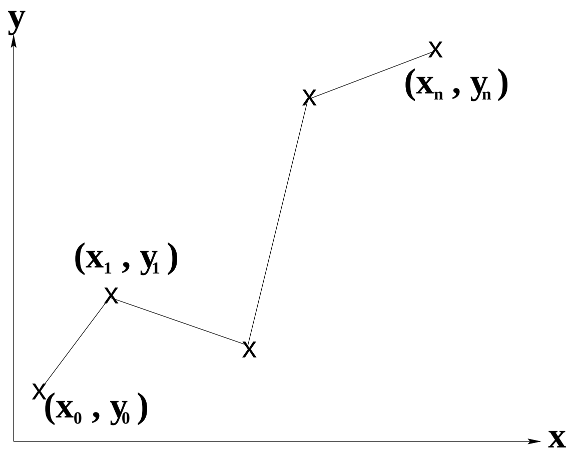
$$y_j = f(x_j), \qquad j = 0, 1, \dots, n.$$

Let $h_j = x_{j+1} - x_j$ and $h = \max_{1 \le j \le n-1} h_j$. Construct the interpolation $I_h(x)$ which is a linear polynomial at each subinterval $[x_j, x_{j+1}]$ and satisfies

$$I_h(x_j) = y_j, \qquad j = 0, 1, \dots, n$$

Obviously,

$$I_h(x) = \frac{x - x_j}{x_{j+1} - x_j} y_{j+1} + \frac{x - x_{j+1}}{x_j - x_{j+1}} y_j, \quad x_j \le x \le x_{j+1}, \quad j = 0, 1, \dots, n-1$$



Convergence Let $f(x) \in C[a, b]$. Since

$$\frac{x - x_j}{x_{j+1} - x_j} + \frac{x - x_{j+1}}{x_j - x_{j+1}} \equiv 1$$

we obtain

$$f(x) = \frac{x - x_j}{x_{j+1} - x_j} f(x) + \frac{x - x_{j+1}}{x_j - x_{j+1}} f(x)$$

Hence on $[x_j, x_{j+1}]$

$$I_h(x) - f(x) = \frac{x - x_j}{x_{j+1} - x_j}(f(x_{j+1}) - f(x)) + \frac{x - x_{j+1}}{x_j - x_{j+1}}(f(x_j) - f(x)).$$

and therefore

$$|I_h(x) - f(x)| \le \left| \frac{x - x_j}{x_{j+1} - x_j} \right| |f(x) - f(x_{j+1})| + \left| \frac{x - x_{j+1}}{x_j - x_{j+1}} \right| |f(x) - f(x_j)|.$$

Since $f(x)$ is continuous on $[a, b]$, $I_h(x)$ converges to $f(x)$ uniformly on $[a, b]$ as $h \to 0$.

Stability

Assume that a perturbation on data $\{y_j\}_{j=0}^n$ is given, i.e.,

$$\tilde{y}_j = y_j + \delta y_j \,.$$

For data $\tilde{y}_j$, the interpolation is

$$\tilde{I}_h(x) = \frac{x - x_j}{x_{j+1} - x_j} \tilde{y}_{j+1} + \frac{x - x_{j+1}}{x_j - x_{j+1}} \tilde{y}_j \qquad x_j \le x \le x_{j+1} \,.$$

The error $|\tilde{I}_h(x) - I_h(x)|$ due to the data perturbation is

$$\begin{aligned} |\tilde{I}_h(x) - I_h(x)| &= \left| \frac{x - x_j}{x_{j+1} - x_j} \delta y_j + \frac{x - x_{j+1}}{x_j - x_{j+1}} \delta y_{j+1} \right| \\ &\le \left| \frac{x - x_j}{x_{j+1} - x_j} \right| |\delta y_j| + \left| \frac{x - x_{j+1}}{x_j - x_{j+1}} \right| |\delta y_{j+1}|, \qquad x_j \le x \le x_{j+1} \end{aligned}$$

Since

$$\left| \frac{x - x_j}{x_{j+1} - x_j} \right| + \left| \frac{x - x_{j+1}}{x_j - x_{j+1}} \right| = 1, \qquad x_j \le x \le x_{j+1}$$

we obtain that

$$|\tilde{I}_h(x) - I_h(x)| \le \max_{0 \le j \le n} |\delta y_j|.$$

Conclusing: The interpolation error does not enlarge the initial perturbation error.

2.3.3 Piecewise Quadratic Interpolation

Given interpolating nodes of odd number:

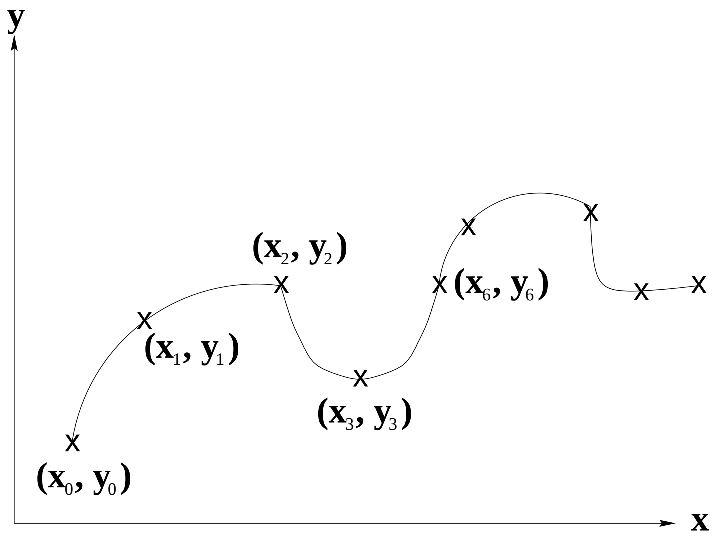
$$a = x_0 < x_1 < \cdots < x_{2k} = b$$

and data

$$f(x_j) = y_j, \qquad j = 0, 1, \ldots, 2k$$

For every three nodes $x_{2j-2}, x_{2j-1}, x_{2j}$ we may construct an interpolating polynomial of second degree on $[x_{2j-2}, x_{2j}]$ by using the data $y_{2j-2}, y_{2j-1}, y_{2j}$, i.e.,

$$\begin{aligned} I_h(x) &= \frac{(x-x_{2j-1})(x-x_{2j})}{(x_{2j-2}-x_{2j-1})(x_{2j-2}-x_{2j})}y_{2j-2} + \frac{(x-x_{2j-2})(x-x_{2j})}{(x_{2j-1}-x_{2j-2})(x_{2j-1}-x_{2j})}y_{2j-1} \\ &\quad + \frac{(x-x_{2j-2})(x-x_{2j-1})}{(x_{2j}-x_{2j-2})(x_{2j}-x_{2j-1})}y_{2j}, \qquad x_{2j-2} \le x \le x_{2j} \end{aligned}$$



Convergence Assume that $f(x) \in C[a,b]$ and x_j are evenly spaced, $h = x_{j+1} - x_j$. Then, in subinterval $[x_{2j-2}, x_{2j}]$, let $s = x - x_{2j-2}$, we have

$$\begin{aligned} I_h(x) &= \frac{(s-h)(s-2h)}{2h^2}y_{2j-2} + \frac{s(2h-s)}{h^2}y_{2j-1} \\ &\quad + \frac{s(s-h)}{2h^2}y_{2j}, \qquad 0 \le s \le 2h. \end{aligned}$$

Since

$$\frac{(s-h)(s-2h)}{2h^2} + \frac{s(2h-s)}{h^2} + \frac{s(s-h)}{2h^2} \equiv 1$$

and the absolute vaule of each term is not greater than 1, we obtain

$$\begin{aligned} |I_h(x) - f(x)| &= \left| \frac{(y-h)(y-2h)}{2h^2}[f(x_{2j-2}) - f(x)] + \right. \\ &\quad \left. \frac{y(2h-y)}{h^2}[f(x_{2j-1}) - f(x)] + \frac{y(y-h)}{2h^2}[f(x_{2j}) - f(x)] \right| \\ &\le 3\max\{\, |f(x_{2j-2}) - f(x)|, |f(x_{2j-1}) - f(x)|, |f(x_{2j}) - f(x)| \,\} \end{aligned}$$

If $f(x)$ is continuous, $I_h(x)$ uniformly converge to $f(x)$ as $h \to 0$.

Stability For data perturbation, we have

$$\begin{aligned}\tilde{I}_h(x) - I_h(x) &= \frac{(y-h)(y-2h)}{2h^2}(\tilde{y}_{2j-2} - y_{2j-2}) + \frac{y(2h-y)}{h^2}(\tilde{y}_{2j-1} - y_{2j-1}) \\ &\quad + \frac{y(y-h)}{2h^2}(\tilde{y}_{2j} - y_{2j}), \qquad 0 \le y \le 2h.\end{aligned}$$

The above result leads to

$$\max_{a\le x\le b} |\tilde{I}_h(x) - I_h(x)| \le 3 \max_{0\le j\le 2h} |\tilde{y}_j - y_j| \le \delta\,.$$

The above piecewise polynomial interpolations are convergent and stable, but of low smoothness (no continuous derivatives). Sometimes they do not meet the smoothness requirement in applications.

2.4 Splines

2.4.1 Definition

Definition 2.3 *A piecewise m^{th} degree polynomial which has continuous derivatives up to order $(m-1)$ is called a* Spline of Degree m .

Assume that there is a subdivision on $[a, b]$

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

and $S(x)$ is a m^{th} degree spline with respect to the subdivision. Then $S(x)$ satisfies

- 1.

$$S(x) = \begin{cases} q_1(x), & x_0 \le x \le x_1 \\ q_2(x), & x_1 \le x \le x_2 \\ \vdots & \vdots \\ q_n(x), & x_{n-1} \le x \le x_n \end{cases}$$

- 2.

$$q_i^{(k)}(x_i) = q_{i+1}^{(k)}(x_i), \quad i = 1, \ldots, n-1, \quad k = 0, \ldots, m-1$$

2.4.2 Cubic Spline for Interpolation

In application, cubic spline is the most useful one. Let $\sigma_3[x_0, x_1, \ldots, x_n]$ denote the set of all cubic splines with respect to the subdivision $x_0 < x_1 < \cdots < x_n$. Then $\sigma_3[x_0, \ldots, x_n]$ is a linear space.

Dimension of $\boldsymbol{\sigma_3[x_0, \ldots, x_n]}$. On each subinterval $[x_{j-1}, x_j]$, the polynomial $q_j(x)$ has 4 coefficients. Hence, there are total $4n$ coefficients of $S(x)$ defined on n subinterval. At the same time, there are 3 constrained conditions on each interval node x_j, $j = 1, \ldots, n-1$. So that, the dimension of $\sigma_3[x_0, \ldots, x_n]$ should be

$$\mathbf{4n - 3(n-1) = n + 3}$$

Given data

$$f(x_j) = y_j, \qquad j = 0, 1, \ldots, n.$$

Assume $h = \max\limits_{0 \le j \le n-1} (x_{j+1} - x_j)$. Find $S_h(x) \in \sigma_3[x_0, \ldots, x_n]$ such that

$$S_h(x_j) = y_j, \qquad j = 0, 1, \ldots, n$$

Because $S_h(x)$ has $(n+3)$ freedoms, it needs two more conditions to make the spline unique. Usually, these conditions are boundary conditions such as

$$S_h'(x_0) = f'(x_0), \quad S_h'(x_n) = f'(x_n) \qquad \text{(clamped boundary)}$$

or

$$S_h''(x_0) = 0, \quad S_h''(x_n) = 0 \qquad \text{(natural boundary)}$$

2.4.3 Construction of $S_h(x)$

Introduce the variable $M_0, M_1, \ldots, M_n$ with

$$M_i = S_h''(x_i), \qquad i = 0, \ldots, n.$$

Then, it can be checked that on subinterval $[x_{j-1}, x_j]$

$$\begin{aligned} S_h(x) \quad = \quad & \frac{(x_j - x)^3 M_{j-1} + (x - x_{j-1})^3 M_j}{6(x_j - x_{j-1})} + \frac{(x_j - x) y_{j-1} + (x - x_{j-1}) y_j}{x_j - x_{j-1}} \\ & - \frac{1}{6}(x_j - x_{j-1})[(x_j - x) M_{j-1} + (x - x_{j-1}) M_j] \end{aligned}$$

To ensure the continuity of $S_h'(x)$ over $[a, b]$, the formulas for $S_h'(x)$ on $[x_{j-1}, x_j]$ and $[x_j, x_{j+1}]$ are required to give a same value at their common point $x = x_j$, $j = 1, \ldots, n-1$. After a great deal of simplification, this leads to the following system of linear equations:

$$\begin{aligned} & \frac{x_j - x_{j-1}}{6} M_{j-1} + \frac{x_{j+1} - x_{j-1}}{3} M_j + \frac{x_{j+1} - x_j}{6} M_{j+1} \\ & = \frac{y_{j+1} - y_j}{x_{j+1} - x_j} - \frac{y_j - y_{j-1}}{x_j - x_{j-1}}, \qquad j = 1, \ldots, n-1 \end{aligned}$$

If M_0, M_n are given as boundary conditions, then these $(n-1)$ equations are just uniquely to determine $M_1, \ldots, M_{n-1}$. If the boundary conditions are given $S_h'(x_0) = f'(x_0)$ and $S_h'(x_n) = f'(x_n)$, then the following 2 more equations will be added:

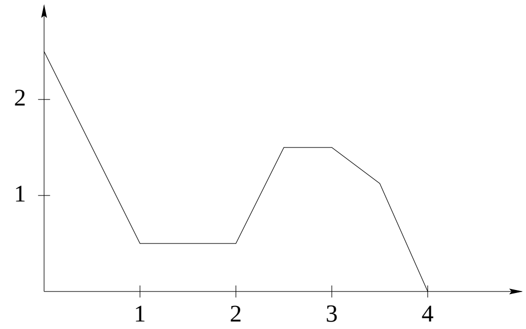
$$\begin{aligned} & \frac{x_1 - x_0}{3} M_0 + \frac{x_1 - x_0}{6} M_1 = \frac{y_1 - y_0}{x_1 - x_0} - f'(x_0), \\ & \frac{x_n - x_{n-1}}{6} M_{n-1} + \frac{x_n - x_{n-1}}{3} M_n = f'(x_n) - \frac{y_n - y_{n-1}}{x_n - x_{n-1}}. \end{aligned}$$

2.4.4 Comparision

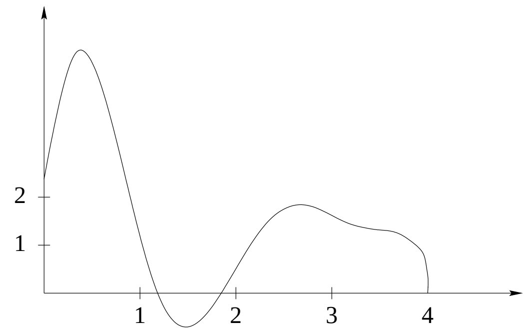
We compare several kinds of interpolation by graph for the following interpolation data points

x	0	1	2	2.5	3	3.5	4
y	2.5	0.5	0.5	1.5	1.5	1.125	0

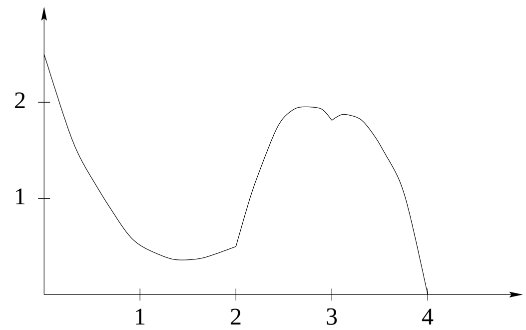
piecewise linear interpolation



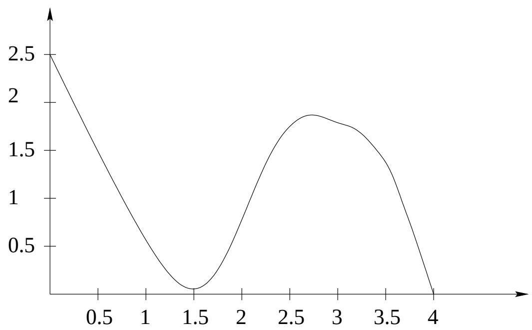
high order interpolation $y \approx p_6(x)$



piecewise quadratic interpolation



natural cubic spline interpolation



2.4.5 B-Splines

Assume that the nodes

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

are equally spaced with spacing h. In this case, a Basis for vector space $\sigma_3[x_0, x_1, \ldots, x_n]$ can be constructed easily and this basis is very useful in applications.

Let $B(x)$ be an **Even function** defined in $(-\infty, +\infty)$

$$B(x) = \begin{cases} 0, & -\infty < x \leq -2h \\ \dfrac{1}{4h^3}(x+2h)^3, & -2h \leq x \leq -h \\ \dfrac{1}{4} + \dfrac{3}{4h}(x+h) + \dfrac{3}{4h^2}(x+h)^2 - \dfrac{3}{4h^3}(x+h)^3, & -h \leq x \leq 0 \end{cases}$$

$$B(x) = B(-x), \qquad 0 \leq x < +\infty$$

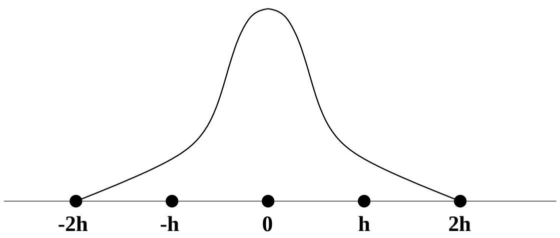


Figure of function *B(x)*

Now define

$$B_i(x) = B(x - x_i), \qquad i = -1, 0, 1, \ldots, n, n+1$$

Then $B_i(x) \in \sigma_3[x_0, x_1, \ldots, x_n]$ and $B_{-1}(x), B_0(x), \ldots, B_n(x), B_{n+1}(x)$ are linearly independent. So that they compose a Basis of $\sigma_3[x_0, x_1, \ldots, x_n]$ and are called B-spline. The B-spline $B_i(x)$ centered at x_i and

$$\begin{aligned} B_i(x_i) &= B(0) &&= 1 \\ B_i(x_{i\pm1}) &= B(\pm h) &&= \tfrac{1}{4} \\ B_i(x_{i\pm2}) &= B(\pm 2h) &&= 0 \\ B_i(x_j) &= 0, && |i-j| \geq 2 \end{aligned}$$

For any cubic spline $S_h(x) \in \sigma_3[x_0, x_1, \ldots, x_n]$,

$$S_h(x) = \sum_{j=-1}^{n+1} \alpha_j B_j(x)$$

2.4.6 Interpolation using B-splines

Given data $y_i = f(x_i),\ i = 0, 1, \dots, n$ and boundary conditions

$$\begin{cases} f'(x_0) = y'_0 \\ f'(x_n) = y'_n \end{cases}$$

find $S_h(x) \in \sigma_3[x_0, x_1, \dots, x_n]$ such that

$$\begin{aligned} S_h(x_i) &= y_i, \qquad i = 0, 1, \dots, n \\ S'_h(x_0) &= y'_0 \\ S'_h(x_n) &= y'_n \end{aligned}$$

2.4.7 Formulation

Let $S_h(x) = \sum_{j=-1}^{n+1} \alpha_j B_j(x)$. Since $B_i(x_j) = 0$ whenever $|i - j| \geq 2$, the conditions $S_h(x_i) = y_i$ implies

$$\begin{aligned} \sum_{j=-1}^{n+1} \alpha_j B_j(x_i) &= \alpha_{i-1} B_{i-1}(x_i) + \alpha_i B_i(x_i) + \alpha_{i+1} B_{i+1}(x_i) = y_i, \\ &\qquad i = 0, \dots, n \end{aligned}$$

Because $B_i(x_i) = 1$ and $B_i(x_{i\pm1}) = \frac{1}{4}$, we have

$$\frac{1}{4}\alpha_{i-1} + \alpha_i + \frac{1}{4}\alpha_{i+1} = y_i, \qquad i = 0, \dots, n$$

The boundary conditions $S'_h(x_0) = y'_0$ and $S'_h(x_n) = y'_n$ become

$$\begin{cases} \alpha_{-1} B'_{-1}(x_0) + \alpha_0 B'_0(x_0) + \alpha_1 B'_1(x_0) & = y'_0 \\ \alpha_{n-1} B'_{n-1}(x_n) + \alpha_n B'_n(x_n) + \alpha_{n+1} B'_{n+1}(x_n) & = y'_n \end{cases}$$

It is easy to check

$$\begin{aligned} B'_0(x_0) &= B'_n(x_n) = B'(0) = 0 \\ B'_{-1}(x_0) &= B'_{n-1}(x_n) = B'(h) = -\frac{3}{4}h^{-1} \\ B'_1(x_0) &= B'_{n+1}(x_n) = B'(-h) = \frac{3}{4}h^{-1} \end{aligned}$$

Hence we have

$$\begin{cases} -\frac{3}{4}\alpha_{-1} + \frac{3}{4}\alpha_1 & = hy'_0 \\ -\frac{3}{4}\alpha_{n-1} + \frac{3}{4}\alpha_{n+1} & = hy'_n \end{cases}$$

If no conditions are posed on the boundary, we can take $\alpha_{-1} = \alpha_{n+1} = 0$ to simplify the problem and get the following linear system of equations to determine $\alpha_0, \alpha_1, \ldots, \alpha_n$:

$$\begin{bmatrix} 1 & \frac{1}{4} & & & \\ \frac{1}{4} & 1 & \frac{1}{4} & & \\ & \ddots & \ddots & \ddots & \\ & & \ddots & \ddots & \ddots \\ & & & \frac{1}{4} & 1 \end{bmatrix} \begin{bmatrix} \alpha_0 \\ \alpha_1 \\ \vdots \\ \vdots \\ \alpha_n \end{bmatrix} = \begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ \vdots \\ y_n \end{bmatrix}$$

2.4.8 Solution of a Tridiagonal System

In the spline interpolation problem, we have to solve a linear system of equations

$$Ax = B$$

with a tridiagonal coefficient matrix

$$A = \begin{bmatrix} a_{11} & a_{12} & & & \\ a_{21} & a_{22} & a_{23} & & \\ & \ddots & \ddots & \ddots & \\ & & \ddots & \ddots & \ddots \\ & & & a_{nn-1} & a_{nn} \end{bmatrix}$$

For solving this tridiagonal system, the procedure is composed by 3 steps:

Step I	Decompose:	$A = LU$
Step II	Solve :	$Lz = b$
Step III	Solve :	$Ux = z$

- Step I: Decompose $A = LU$, where L and U are lower and upper triangular matrix with only two diagonals.

$$\begin{bmatrix} a_{11} & a_{12} & & & & \\ a_{21} & a_{22} & a_{23} & & & \\ & \ddots & \ddots & \ddots & & \\ & a_{ii-1} & a_{ii} & a_{ii+1} & & \\ & & \ddots & \ddots & \ddots & \\ & & & & a_{nn-1} & a_{nn} \end{bmatrix} =$$

$$\begin{bmatrix} l_{11} & & & & & \\ l_{21} & l_{22} & & & & \\ & \ddots & \ddots & & & \\ & & l_{ii-1} & l_{ii} & & \\ & & & \ddots & \ddots & \\ & & & & l_{nn-1} & l_{nn} \end{bmatrix} \begin{bmatrix} 1 & u_{12} & & & & \\ & 1 & u_{23} & & & \\ & & \ddots & \ddots & & \\ & & & 1 & u_{ii+1} & \\ & & & & \ddots & \ddots \\ & & & & & 1 \end{bmatrix}$$

By the multiplication of matrices, we have

$$
\begin{array}{llll}
l_{11} = a_{11}, & l_{11}u_{12} = a_{12} & & \\
l_{21} = a_{21}, & l_{21}u_{12} + l_{22} = a_{22}, & l_{22}u_{23} = a_{23} & \\
\ddots & \ddots & \ddots & \\
l_{ii-1} = a_{ii-1}, & l_{ii-1}u_{i-1i} + l_{ii} = a_{ii}, & l_{ii}u_{ii+1} = a_{ii+1} & \\
\ddots & \ddots & \ddots & \\
l_{nn-1} = a_{nn-1}, & l_{nn-1}u_{n-1n} + l_{nn} = a_n & &
\end{array}
$$

Therefore, we obtain the following:

$$
\begin{array}{lll}
l_{11} = a_{11}, & u_{12} = a_{12}/l_{11} & \\
l_{21} = a_{21}, & l_{22} = a_{22} - l_{21}u_{12}, & u_{23} = a_{23}/l_{22} \\
\ddots & \ddots & \ddots \\
l_{ii-1} = a_{ii-1}, & l_{ii} = a_{ii} - l_{ii-1}u_{i-1i}, & u_{ii+1} = a_{ii+1}/l_{ii}
\end{array}
$$

In summary, we have the following alogrithm to compute l_{ij} and u_{ij}:

$$
\boxed{
\begin{array}{l}
a_{11} \Rightarrow l_{11}; \qquad a_{12}/l_{11} \Rightarrow u_{12}; \\
\text{For } j = 2 \text{ to } n-1 \\
\quad a_{j,j-1} \Rightarrow l_{j,j-1}; \\
\quad a_{j,j} - l_{j,j-1} * u_{j-1,j} \Rightarrow l_{j,j}; \\
\quad a_{j,j+1}/l_{j,j} \Rightarrow u_{j,j+1}; \\
a_{n,n-1} \Rightarrow l_{n,n-1}; \qquad a_{n,n} - l_{n,n-1} * u_{n-1,n} \Rightarrow l_{n,n}
\end{array}
}
$$

- **Step II:** Solve: $Lz = b$

$$
\boxed{
\left\{
\begin{array}{lcl}
l_{11}z_1 & = & b_1 \\
l_{21}z_1 + l_{22}z_2 & = & b_2 \\
\vdots & & \vdots \\
l_{i,i-1}z_{i-1} + l_{ii}z_i & = & b_i \\
\vdots & & \vdots \\
l_{n,n-1}z_{n-1} + l_{nn}z_n & = & b_n
\end{array}
\right.
\Rightarrow
\left\{
\begin{array}{lcl}
z_1 & = & b_1/l_{11} \\
z_2 & = & (b_2 - l_{21}z_1)/l_{22} \\
\vdots & & \vdots \\
z_i & = & (b_i - l_{i,i-1}z_{i-1}/l_{ii} \\
\vdots & & \vdots \\
z_n & = & (b_n - l_{n,n-1}z_{n-1}/l_{nn}
\end{array}
\right.
}
$$

- **Step III:** Solve: $Ux = z$

$$\begin{cases} x_1 + u_{12}x_2 & = & z_1 \\ x_2 + u_{23}x_3 & = & z_2 \\ \vdots & & \vdots \\ \vdots & & \vdots \\ x_{n-1} + u_{n-1,n}x_n & = & z_{n-1} \\ x_n & = & z_n \end{cases} \Rightarrow \begin{cases} x_n & = & z_n \\ x_{n-1} & = & z_{n-1} - u_{n-1,n}x_n \\ \vdots & = & \vdots \\ x_i & = & z_i - u_{i,i+1}x_{i+1} \\ \vdots & = & \vdots \\ x_1 & = & z_1 - u_{12}x_2 \end{cases}$$

2.4.9 Diagonally Dominant Matrix

Matrix A is said strictly diagonally dominant if its elments satisfy

$$a_{ii} > \sum_{j \neq i} |a_{ij}|, \qquad i = 1, \ldots, n$$

A strictly diagonally dominant matrix is non-singular. For a tridiagonal system with strictly diagonally dominant matrix, the above algorithm will not break down and will be numerically stable.

In spline interpolation problem, the tridiagonal matrix is always strictly diagonally dominant.

Homework 2.1

- 2.1 Assume that f is a given function for which the following values are known: $f(1) = 2, f(2) = 3, f(3) = 5, f(4) = 3$. For these data find the interpolation polynomial of degree 3 and write it in the form $a_0 + a_1 + a_2x^2 + a_3x^3$.
- 2.2 In the last question, find the quadratic spline function that satisfies the condition $q'(1) = 0$. (Hint: Start from the left)
- 2.3 Let f be sufficiently differentiable. Show that the error in the linear piecewise function is $O(h^2)$.
- 2.4 Approximate the function $f(x) = (1 + 25x^2)^{-1}$ on the interval $[-1, 1]$ by:
  - **a.** Write a program for computing the interpolating polynomial p_n for the points $x_i = -1 + (2i)/n, i = 0, 1, \cdots, n$, for $n = 5, 10, 25$.
  - **b.** Do the same but use the points $x_i = \cos(2i\pi/n)$.

  Plot the error function $f(x) - p_n(x)$ in each case and comment on the difference in the two approaches.

2.5 Least Square Approximation

Suppose that $p(x)$ is a polynomial of degree n, and that we wish to determine its coefficients. By interpolation, it suffices to know $p(x)$ at $(n+1)$ distinct points. In many situations, however, the values of $p(x)$ can be found only by measurements and may be in error. In this case, it is common to take more than $(n+1)$ measurements in the hope that these measurement errors will *average out*.

The way in which these errors average out is determined by the method used to combined the measurements to obtain the coefficients of $p(x)$. For both computational and statistical reasons, the method of choice is often called *least squares* and this method also enjoys an elegant mathematical simplicity.

Example : *(Spring Constant)* Hooke's law states that when a force is applied to a spring constructed of uniform material, the length of the spring is a linear function of the force applied:

$$F(l) = k(l - E)$$

where F represents the force, constant E represents the length of the spring with no force applied, and the constant k is called the spring constant. We want to determine the spring constant k and have the following data from measurement:

F	0	2	4	6
l	5.3	7.0	9.4	12.3

A quick examination shows that the points $(0, 5.3), (2, 7.0), (4, 9.4), (6, 12.3)$ do not lie on a straight line. Although we could simply use one random pair of these data points to approximate the spring constant, it would seem more reasonable to find the line that best approximates all the data points to determine the constant. That is, to find k such that

$$\sum_{i=1}^{4} \Big(k(l_i - E) - F_i\Big)^2 = \min$$

In other words, we want to fund the minimum value of the following function:

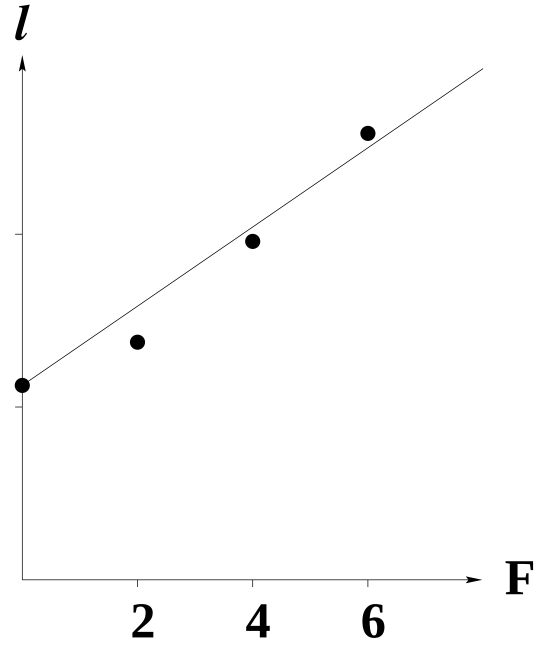
$$g(k) = \sum_{i=1}^{4} \Big(k(l_i - E) - F_i\Big)^2.$$

calculus, we know that g takes on a local minimum at a point k_0 which satisfies $g'(k_0) = 0,\ g''(k_0) \geq 0$. Now

$$\begin{aligned} g'(k) &= 2\sum_{i=1}^{4} \Big(k(l_i - E) - F_i\Big)(l_i - E) \\ g''(k) &= 2\sum_{i=1}^{4} (l_i - E)^2 > 0\,. \end{aligned}$$

We obtain

$$k_0 = \frac{\sum_{i=1}^{4} F_i(l_i - E)}{\sum_{i=1}^{4}(l_i - E)^2} = 0.90$$



2.5.1 Weighting

Assume that we have m pairs of data:

$$(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m),$$

where $m \geq n+1$ and at least $(n+1)$ of x_i are distinct. Then we wish to find a polynomial $p(x) = a_0 + a_1 x + \cdots + a_n x^n$ such that

$$\sum_{i=1}^{m}[y_i - p(x_i)]^2 = \min$$

It means, we wish to find $a_0, a_1, \ldots, a_n$ so that the sum of the squares of the errors $|y_i - p(x_i)|$ is minimized. That is why we call the approximation *Least Square*. More generally, because the y_i's are measurements and we have different confidence in them, we would find $a_0, a_1, \ldots, a_n$ such that

$$\sum_{i=1}^{m} w_i[y_i - p(x_i)]^2 = \min$$

where w_i are given positive constants, called weights.

For example, if we have good confidence in $y_1, \ldots, y_{10}$ and little confidence in the rest, we might set $w_1 = w_2 = \cdots = w_{10} = 5$ and the rest equal 1. For some problems arising in signal processing, it is common to take $w_j = \alpha^j$ for some α between 0 and 1.

2.5.2 Normal Equations

Let

$$g(a_0, a_1, \ldots, a_n) = \sum_{i=1}^{m} w_i (p(x_i) - y_i)^2, \qquad m \geq n+1$$

From the calculus, a necessary condition for g to be minimized is that

$$\frac{\partial g}{\partial a_j}(a_0, a_1, \ldots, a_n) = 0, \qquad j = 0, \ldots, n$$

$$\Rightarrow \sum_{i=1}^{m} w_i x_i^j (a_0 + a_1 x_i + \cdots a_n x_i^n - y_i) = 0, \qquad j = 0, 1, \ldots, n$$

Collecting the coefficients of the a_j and rewriting the equations in matrix form gives the system

$$\begin{bmatrix} s_0 & s_1 & s_2 & \cdots & s_n \\ s_1 & s_2 & s_3 & \cdots & s_{n+1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ s_n & s_{n+2} & s_{n+3} & \cdots & s_{2n} \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{bmatrix} = \begin{bmatrix} c_0 \\ c_1 \\ \vdots \\ c_n \end{bmatrix} \tag{2.1}$$

where

$$s_j = \sum_{i=1}^{m} w_i x_i^j, \qquad c_j = \sum_{i=1}^{m} w_i x_i^j y_i\,. \tag{2.2}$$

These equations are known as Normal Equations.

Note that the coefficnent matrix is symmetric and determined by only $(2n+1)$ quantities $s_0, \ldots, s_{2n}$. Such a matrix is called a Hankel matrix and has many interesting properties. The quantities s_j are known as moments.

2.5.3 General Linear Least Squares Problem

Consider more general least-squares problems in which the approximating function is not necessarily a polynomial but is linear combination of given functions $\phi_0, \phi_1, \ldots, \phi_n$

$$\phi(x) = \sum_{i=0}^{n} a_i \phi_i(x).$$

If $\phi_j(x) = x^j$, then $\phi(x)$ is the polynomial considered previously. Other common choices for the basis functions ϕ_j are:

$$\phi_j(x) = \cos j\pi x; \qquad \phi_j(x) = e^{\alpha_j x}, \quad \alpha_j \text{ real numbers.}$$

We could also use piecewise polynomials or splines.

Problem: To find $a_0, a_1, \ldots, a_n$ such that

$$\sum_{i=1}^{m} w_i [\phi(x_i) - y_i]^2 = \min$$

where (x_i, y_i) are given data.

Let $g(a_0, a_1, \ldots, a_n) = \sum_{i=1}^{m} w_i[\phi(x_i) - y_i]^2$. Straightforward computation gives

$$\frac{\partial g}{\partial a_j} = 2\sum_{k=1}^{m} w_k \phi_j(x_k)[a_0\phi_0(x_k) + \cdots + a_n\phi_n(x_k) - y_k]$$

Setting these equal to zero, collecting coefficients of the a_i, we obtain the following linear system

$$\begin{bmatrix} s_{00} & s_{01} & \cdots & s_{0n} \\ s_{10} & s_{11} & \cdots & s_{1n} \\ \vdots & \vdots & \ddots & \vdots \\ s_{n0} & s_{n1} & \cdots & s_{nn} \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{bmatrix} = \begin{bmatrix} c_0 \\ c_1 \\ \vdots \\ c_n \end{bmatrix}$$

where

$$s_{ji} = \sum_{k=1}^{m} w_k \phi_j(x_k)\phi_i(x_k), \qquad c_j = \sum_{k=1}^{m} w_k \phi_j(x_k) y_k$$

The coefficient matrix is again symmetric but in order to conclude that it is positive definite, suitable conditions must be imposed on $\{\phi_j\}_{j=0}^{n}$ and $\{x_i\}_{i=1}^{m}$.

2.5.4 Orthogonal Polynomials

The normal equtaions are useful for theoretical purposes, or for computation when n is small. But they have a tendency to become very ill-coditioned (unstable) as n increases.

Definition 2.4 *Let $q_0(x), q_1(x), \ldots, q_n(x)$ be polynomials of degree $0, 1, \ldots, n$ respectively. Then we will say that $q_0, q_1, \ldots, q_n$ are orthogonal with respect to the points $x_1, \ldots, x_m$ and weights $w_1, \ldots, w_m$ if*

$$\sum_{k=1}^{m} w_k q_i(x_k) q_j(x_k) = 0, \qquad i \neq j\ .$$

If we take $\{q_i(x)\}_{i=0}^{n}$ as basis functions instead of $\{x^i\}_{i=0}^{n}$ for least square approximation, the coefficient matrix S of the normal equations will become a diagonal matrix, i.e.,

$$\begin{bmatrix} s_{00} & & & \\ & s_{11} & & \\ & & \ddots & \\ & & & s_{nn} \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{bmatrix} = \begin{bmatrix} c_0 \\ c_1 \\ \vdots \\ c_n \end{bmatrix}$$

where

$$s_{ii} = \sum_{k=1}^{m} w_k q_i^2(x_k), \qquad c_i = \sum_{k=1}^{m} w_k q_i(x_k) y_k$$

Hence we obtain the least-squares polynomial

$$p(x) = \sum_{i=0}^{n} a_i q_i(x)$$

with coefficients

$$a_i = \frac{\sum_{k=1}^{m} w_k q_i(x_k) y_k}{\sum_{k=1}^{m} w_k q_i^2(x_k)} \tag{2.3}$$

The use of orthogonal basis reduces the normal equations to a diagonal system of equations, which is trivial to solve.

2.5.5 Computation of the $q_i(x)$

Assume all the weights $w_i = 1$ (the general case left as an assignment). Let $q_0(x) \equiv 1$ and $q_1(x) = x - \alpha_1$, where α_1 is to be determined. Since

$$0 = \sum_{k=1}^{m} q_0(x_k) q_1(x_k) = \sum_{k=1}^{m} (x_k - \alpha_1) = \sum_{k=1}^{m} x_k - m\alpha_1$$

$$\Rightarrow \quad \alpha_1 = \frac{\sum_{k=1}^{m} x_k}{m} . \tag{2.4}$$

Now let $q_2(x) = xq_1(x) - \alpha_2 q_1(x) - \beta_1$, where α_2 and β_1 are to be determined. Because

$$\begin{aligned} 0 &= \sum_{k=1}^{m} q_2(x_k) q_0(x_k) &= \sum_{k=1}^{m} [x_k q_1(x_k) - \alpha_2 q_1(x_k) - \beta_1] \\ 0 &= \sum_{k=1}^{m} q_2(x_k) q_1(x_k) &= \sum_{k=1}^{m} [x_k q_1(x_k) - \alpha_2 q_1(x_k) - \beta_1] q_1(x_k) \end{aligned}$$

and noting that $\sum_{k=1}^{m} q_1(x_k) = 0$, we have

$$\sum_{k=1}^{m} x_k q_1(x_k) - m\beta_1 = 0, \qquad \sum_{k=1}^{m} (x_k q_1^2(x_k) - \alpha_2 q_1^2(x_k)) = 0$$

$$\Rightarrow \quad \beta_1 = \frac{\sum_{k=1}^{m} x_k q_1(x_k)}{m}, \qquad \alpha_2 = \frac{\sum_{k=1}^{m} x_k q_1^2(x_k)}{\sum_{k=1}^{m} q_1^2(x_k)}.$$

The computation for the remaining $q_i(x)$ proceeds in an analogous fashion.

Three-Term Recurrence Relation: Assume that we have determined $q_0, q_1, \ldots, q_j$, then we can define q_{j+1} in terms of q_j and q_{j-1} as following so called three-term recurrence formula

$$q_{j+1}(x) = xq_j(x) - \alpha_{j+1}q_j(x) - \beta_j q_{j-1}(x) \tag{2.5}$$

where α_{j+1} and β_j are to be determined from the orthogonality requirements

$$\sum_{k=1}^{m} q_{j+1}(x_k)q_j(x_k) = 0 \tag{2.6}$$

$$\sum_{k=1}^{m} q_{j+1}(x_k)q_{j-1}(x_k) = 0 \tag{2.7}$$

If these two relations are satisfied, then q_{j+1} must be orthogonal to all the previous q_l, $0 \le l \le j-2$. Since by (2.5)

$$\begin{aligned}\sum_{k=1}^{m} q_{j+1}(x_k)q_l(x_k) &= \sum_{k=1}^{m} x_k q_j(x_k)q_l(x_k) \\ &\quad -\alpha_{j+1}\sum_{k=1}^{m} q_j(x_k)q_l(x_k) - \beta_j \sum_{k=1}^{m} q_{j-1}(x_k)q_l(x_k)\end{aligned}$$

The last two terms are zero by assumption, whereas $xq_l(x)$ is a polynomial of degree $(l+1)$ and can be expressed as a linear combination of $q_0, q_1, \ldots, q_{l+1}$. Hence, the first term is zero. Returning to (2.6)(2.7) and putting in q_{j+1} from (2.5) leads to

$$\begin{aligned}\alpha_{j+1} &= \frac{1}{\gamma_j}\sum_{k=1}^{m} x_k q_j^2(x_k), \qquad \gamma_j = \sum_{k=1}^{m} q_j^2(x_k) \\ \beta_j &= \frac{1}{\gamma_{j-1}}\sum_{k=1}^{m} x_k q_j(x_k)q_{j-1}(x_k)\end{aligned} \tag{2.8}$$

The β_j can be computed in a better way if we substitute for $xq_{j-1}(x)$ using (2.5) and then note that

$$\sum_{k=1}^{m} q_j(x_k)[q_j(x_k) + \alpha_j q_{j-1}(x_k) + \beta_{j-1}q_{j-2}(x_k)] = \sum_{k=1}^{m} q_j^2(x_k) = \gamma_j$$

by the orthogonality of the q's. Thus

$$\beta_j = \frac{\gamma_j}{\gamma_{j-1}}. \tag{2.9}$$

Summarize the orthogonal polynomial algorithm as follows:

- (i) Set $q_0(x) \equiv 1, \qquad q_1(x) = x - \frac{1}{m}\sum_{k=1}^{m} x_k, \qquad \gamma_0 = m,$
- (ii) Compute for $j = 1$ to $n-1$

$$\begin{aligned}
\gamma_j &= \sum_{k=1}^{m} q_j^2(x_k) \\
\alpha_{j+1} &= \frac{1}{\gamma_j}\sum_{k=1}^{m} x_k q_j^2(x_k), \qquad \beta_j = \frac{\gamma_j}{\gamma_{j-1}} \\
q_{j+1}(x) &= xq_j(x) - \alpha_{j+1}q_j(x) - \beta_j q_{j-1}(x)
\end{aligned}$$

- (iii) Compute the coefficients $a_0, a_1, \ldots, a_n$ of the least-squares polynomial

$$p(x) = a_0 q_0(x) + a_1 q_1(x) + \cdots + a_n q_n(x).$$

Advantages: Orthogonal system is often used numerically since it avoids solving the possibly ill-conditioned system. Anpther advantage is that we are able to build up the least-squares polynomial degree by degree. For example, if we do not know what degree polynomial we wish to use, we might start with a first-degree polynomial, then a second-degree, and so on, until we obtain a fit that we believe is suitable.

2.6 A Numerical Example

We now give a simple example, using the data

$$\begin{array}{lllll}
x_1 = 0 & x_2 = \dfrac{1}{4} & x_3 = \dfrac{1}{2} & x_4 = \dfrac{3}{4} & x_5 = 1 \\
y1 = 1 & y_2 = 2 & y_3 = 1 & y_4 = 0 & y_5 = 1\,.
\end{array}$$

Since there are five points x_i, these data will uniquely determine a fourth-degree interpolating polynomial. We will compute the linear and quadratic least square polynomials by both the normal equations and the orthorgonal polynomial approach.

2.6.1 Linear polynomial

If without special mention, we take $w_k \equiv 1$. It follows from (2.2) that we will need the following quantities:

$$\sum_{i=1}^{5} x_i = \frac{5}{2}, \quad \sum_{i=1}^{5} x_i^2 = \frac{15}{8}, \quad \sum_{i=1}^{5} y_i = 5, \quad \sum_{i=1}^{5} x_i y_i = 2. \tag{2.10}$$

Then solving the normal equation (2.1) gives

$$a_0 = \frac{7}{5}, \quad a_1 = -\frac{4}{5}.$$

Thus, the linear least-squares polynomial is

$$p_1(x) = \frac{7}{5} - \frac{4}{5}x.$$

If we compute the same polynomial by orthorgonal polynomials, the polynomial is given in the form

$$a_0 q_0(x) + a_1 q_1(x) = a_0 + a_1(x - \alpha_1), \tag{2.11}$$

where a_0 and a_1 are given by (2.3) and (2.10), and α_1 by (2.4):

$$a_0 = 1, \quad a_1 = -\frac{4}{5}, \quad \alpha_1 = \frac{1}{2}.$$

Therefore, the polynomial (2.11) is $1 - \frac{4}{5}(x - \frac{1}{2})$, which is, as expected, the same as the one obtained with the normal equation approach.

2.6.2 Quadratic Polynomial

To compute the least-squares quadratic polynomial by the normal equations, we need to solve the system (2.1) for $n = 2$, which for our data is

$$\begin{bmatrix} 640 & 320 & 240 \\ 320 & 240 & 200 \\ 240 & 200 & 177 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} 640 \\ 256 \\ 176 \end{bmatrix}.$$

The solution of this system is

$$a_0 = \frac{7}{5}, \qquad a_1 = \frac{-4}{5}, \qquad a_2 = 0. \tag{2.12}$$

Thus, the best least-squares polynomial approximation turns out to be the linear least-squares polynomial; that is, no improvement to the linear approximation can be made by adding a quadratic term. That (2.12) is correct is verified by computing the least-squares quadratic by orthogonal polynomials.

The orthogonal polynomial representation will be

$$a_0 q_0(x) + a_1 q_1(x) + a_2 q_2(x) = \frac{7}{5} - \frac{4}{5}x + a_2 \left[x\left(x - \frac{1}{2}\right) - \alpha_2 \left(x - \frac{1}{2}\right) - \beta_1 \right],$$

where α_2 and β_1 are computed from (2.8) and (2.9) as

$$\alpha_2 = \frac{\sum x_i (x_i - \frac{1}{2})^2}{\sum (x_i - \frac{1}{2})^2} = \frac{1}{2}, \qquad \beta_1 = \frac{1}{5} \sum x_i (x_i - \frac{1}{2}) = \frac{1}{8}.$$

Thus, $q_2(x) = x^2 - x - \frac{1}{8}$, and from (2.3), we find that $a_2 = 0$.

Homework 2.2

- 2.5: Assume that f is a given function for which the following values are known: $f(1) = 2, f(2) = 3, f(3) = 5, f(4) = 3$. Find the constant, linear and quadratic least-squares polynomials by both the normal-equation and orthogonal-polynomial approaches.

- 2.6: Write a computer program to obtain the least-squares polynomial of degree n using $m \geq n + 1$ data points by the orthogonal-polynomial approach. Test your program on the polynomials in Question 2.5.