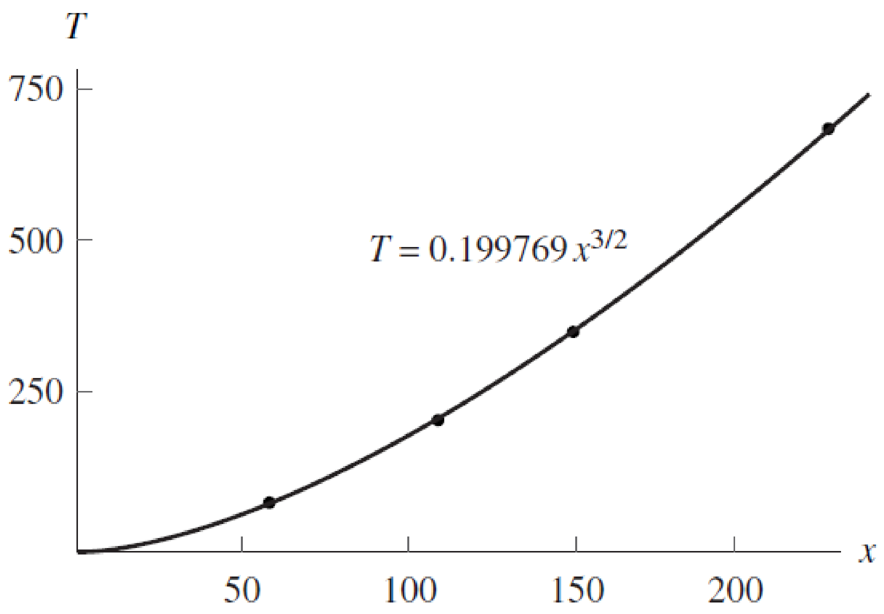
Curve Fitting

Peng Yu

Tel: 0755 8801 8911
Email: yup6@sustech.edu.cn

Introduction

- Interpolation vs Curve Fitting (approximation)
- Why Curve Fitting?



x: distance to the sun
T: orbital period in days

Figure 5.1 The least-squares fit $T = 0.199769x^{3/2}$ for the first four planets using Kepler's third law of planetary motion.

The observed data pairs (x, T) for the first four planets, Mercury, Venus, Earth, and Mars, are (58, 88), (108, 225), (150, 365), and (228, 687),

Curve Fitting

- Least-Squares Line
- Methods of Curve Fitting
- Interpolation by Spline Functions
- Fourier Series and Trigonometric Polynomials
- Bézier Curves

Least-Squares Line

Curve Fitting

- In science and engineering it is often the case that an experiment produces a set of data points $(x_1, y_1), \ldots, (x_N, y_N)$, where the abscissas $\{x_k\}$ are distinct. One goal of numerical methods is to determine a formula $y = f(x)$ that relates these variables.
- Usually, a class of allowable formulas is chosen and then coefficients must be determined.
- There are many different possibilities for the type of function that can be used, depending on the underlying physics.
- The class of linear functions is often used

$$y = f(x) = Ax + B$$

Why curve fitting

- In last chapter, we saw how to construct a polynomial that passes through a set of points.
- If all the numerical values $\{x_k\}$, $\{y_k\}$ are known to several significant digits of accuracy, then polynomial interpolation can be used successfully; otherwise, it cannot.
- Many experiments are done with equipment that is reliable only to three or fewer digits of accuracy.
- Often, there is an experimental error in the measurements, and although three digits are recorded for the values $\{x_k\}$ and $\{y_k\}$, it is realized that the true value $f(x_k)$ satisfies

$$f(x_k) = y_k + e_k ,$$

where e_k is the measurement error

How to find the best linear approximation

- Consider the ***errors*** (or ***deviations*** or ***residuals***)

Maximum error: $$E_\infty(f) = \max_{1\le k\le N}\{|f(x_k) - y_k|\},$$

Average error: $$E_1(f) = \frac{1}{N}\sum_{k=1}^{N}|f(x_k) - y_k|,$$

Root-mean-square error (rms): $$E_2(f) = \left(\frac{1}{N}\sum_{k=1}^{N}|f(x_k) - y_k|^2\right)^{1/2}$$

An example

Example Compare the maximum error, average error, and rms error for the linear approximation $y = f(x) = 8.6 - 1.6x$ to the data points $(-1, 10), (0, 9), (1, 7), (2, 5)$, $(3, 4), (4, 3), (5, 0)$, and $(6, -1)$.

Table 5.1 Calculations for Finding $E_1(f)$ and $E_2(f)$ for Example 5.1

x_k	y_k	$f(x_k) = 8.6 - 1.6x_k$	$\lvert e_k \rvert$	e_k^2
−1	10.0	10.2	0.2	0.04
0	9.0	8.6	0.4	0.16
1	7.0	7.0	0.0	0.00
2	5.0	5.4	0.4	0.16
3	4.0	3.8	0.2	0.04
4	3.0	2.2	0.8	0.64
5	0.0	0.6	0.6	0.36
6	−1.0	−1.0	0.0	0.00
			2.6	1.40

$$E_\infty(f) = \max\{0.2, 0.4, 0.0, 0.4, 0.2, 0.8, 0.6, 0.0\} = 0.8\,,$$

$$E_1(f) = \frac{1}{8}(2.6) = 0.325\,,$$

$$E_2(f) = \left(\frac{1.4}{8}\right)^{1/2} \approx 0.41833\,.$$

Error $E_2(f)$ is often used when the statistical nature of the errors is considered

Finding the Least-Squares Line

- Let $\{(x_k, y_k)\}$ be a set of N points, where the abscissas $\{x_k\}$ are distinct. The least squares line $y = f(x) = Ax + B$ is the line that minimizes the root-mean-square error $E_2(f)$.

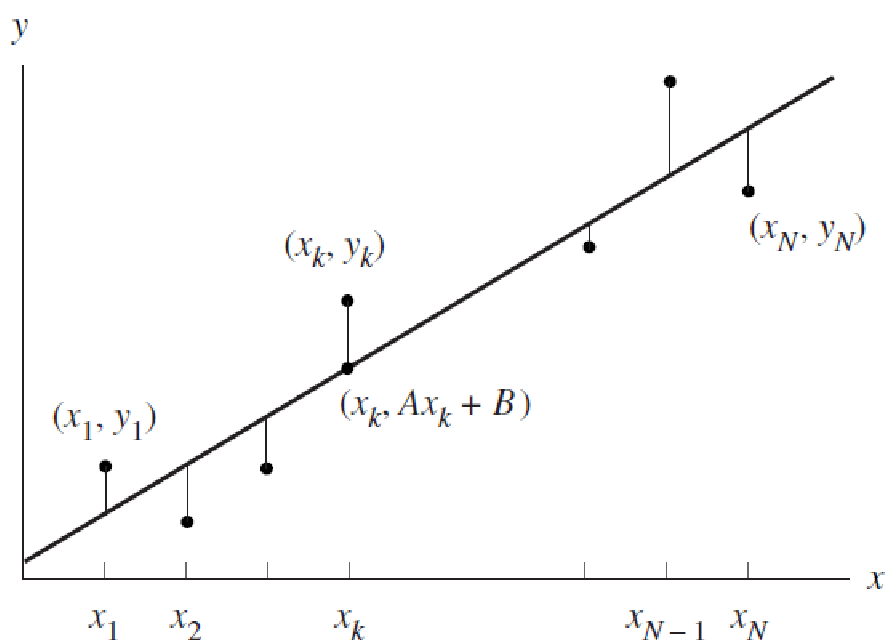


Figure 5.2 The vertical distances between the points $\{(x_k, y_k)\}$ and the least-squares line $y = Ax + B$.

Finding the Least-Squares Line

Theorem (Least-Squares Line). Suppose that $\{(x_k, y_k)\}_{k=1}^{N}$ are N points, where the abscissas $\{x_k\}_{k=1}^{N}$ are distinct. The coefficients of the least-squares line

$$y = Ax + B$$

are the solution to the following linear system, known as the ***normal equations:***

$$\left(\sum_{k=1}^{N} x_k^2\right) A + \left(\sum_{k=1}^{N} x_k\right) B = \sum_{k=1}^{N} x_k y_k\,, \tag{10}$$

$$\left(\sum_{k=1}^{N} x_k\right) A + NB = \sum_{k=1}^{N} y_k\,.$$

Proof: minimize the sum of the squares of the errors (vertical distances d_k)

$$E(A,B) = \sum_{k=1}^{N} (Ax_k + B - y_k)^2 = \sum_{k=1}^{N} d_k^2\,.$$

Finding the Least-Squares Line

The minimum value of $E(A, B)$ is determined by setting the partial derivatives $\partial E/\partial A$ and $\partial E/\partial B$ equal to zero and solving these equations for A and B.

$$\frac{\partial E(A,B)}{\partial A} = \sum_{k=1}^{N} 2(Ax_k + B - y_k)(x_k) = 2\sum_{k=1}^{N}(Ax_k^2 + Bx_k - x_k y_k) \; .$$

$$\frac{\partial E(A,B)}{\partial B} = \sum_{k=1}^{N} 2(Ax_k + B - y_k) = 2\sum_{k=1}^{N}(Ax_k + B - y_k) \; .$$

$$0 = \sum_{k=1}^{N}(A{x_k}^2 + Bx_k - x_k y_k) = A\sum_{k=1}^{N} x_k^2 + B\sum_{k=1}^{N} x_k - \sum_{k=1}^{N} x_k y_k \; .$$

$$0 = \sum_{k=1}^{N}(Ax_k + B - y_k) = A\sum_{k=1}^{N} x_k + NB - \sum_{k=1}^{N} y_k \; .$$

An example

Find the least-squares line for the data points given in last Example

$$92A + 20B = 25$$
$$20A + \quad 8B = 37$$

x_k	y_k
−1	10.0
0	9.0
1	7.0
2	5.0
3	4.0
4	3.0
5	0.0
6	−1.0

$A \approx -1.6071429$ and $B \approx 8.6428571.$

$y = -1.6071429x + 8.6428571$

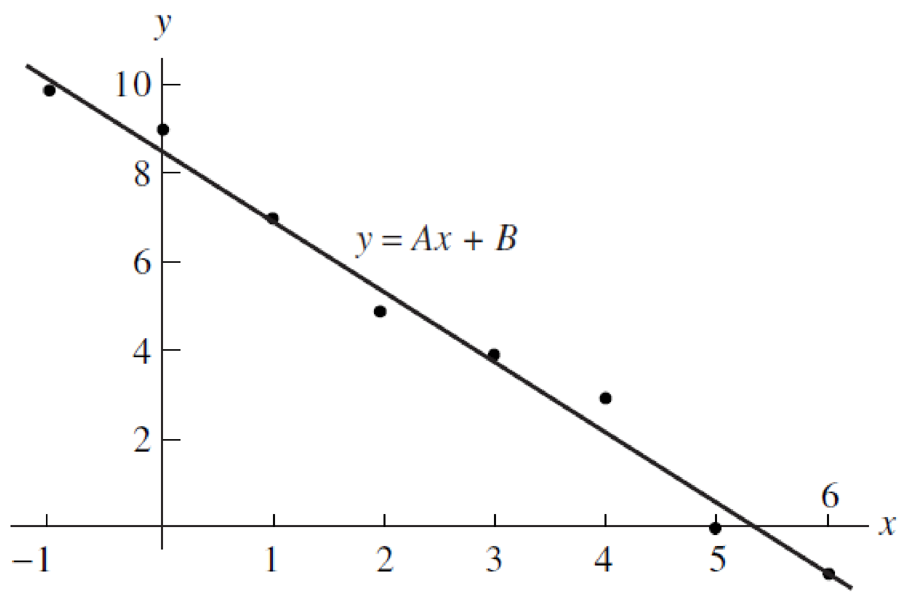


Figure 5.3 The least-squares line $y = -1.6071429x + 8.6428571.$

Power Fit $y = A x^M$

- Some situations involve $f(x) = Ax^M$, where M is a known constant, eg. the example of planetary motion

Theorem (Power Fit). Suppose that $\{(x_k, y_k)\}_{k=1}^{N}$ are N points, where the abscissas are distinct. The coefficient A of the least-squares power curve $y = Ax^M$ is given by

$$A = \left(\sum_{k=1}^{N} x_k^M y_k\right) \Big/ \left(\sum_{k=1}^{N} x_k^{2M}\right).$$

Proof: minimize the following function

$$E(A) = \sum_{k=1}^{N} (Ax_k^M - y_k)^2 .$$

$$\Downarrow$$

$$E'(A) = 2\sum_{k=1}^{N} \left(Ax_k^M - y_k\right)(x_k^M) = 2\sum_{k=1}^{N} (Ax_k^{2M} - x_k^M y_k) .$$

$$0 = A\sum_{k=1}^{N} x_k^{2M} - \sum_{k=1}^{N} x_k^M y_k ,$$

Matlab Code

Program 5.1 (Least-Squares Line). To construct the least-squares line $y = Ax + B$ that fits the N data points $(x_1, y_1), \ldots, (x_N, y_N)$.

```
function [A,B]=lsline(X,Y)

%Input  - X is the 1xn abscissa vector
%       - Y is the 1xn ordinate vector
%Output - A is the coefficient of x in Ax + B
%       - B is the constant coefficient in Ax + B

xmean=mean(X);
ymean=mean(Y);
sumx2=(X-xmean)*(X-xmean)';
sumxy=(Y-ymean)*(X-xmean)';
A=sumxy/sumx2;
B=ymean-A*xmean;
```

Methods of Curve Fitting

Data Linearization Method for $y = Ce^{Ax}$

Suppose that we are given the points $(x_1, y_1), (x_2, y_2), \dots, (x_N, y_N)$ and want to fit an exponential curve of the form

$$y = Ce^{Ax}.$$

The first step is to take the logarithm of both sides:

$$\ln(y) = Ax + \ln(C).$$

Then introduce the change of variables:

$$Y = \ln(y), \qquad X = x, \qquad \text{and} \quad B = \ln(C).$$

This result in a linear relation between the new variables X and Y:

$$Y = Ax + B.$$

The original points (x_k, y_k) in the xy-plane are transformed into the points $(X_k, Y_k) = (x_k, \ln(y_k))$ in the XY-plane. This process is called ***data-linearization.*** Then the least-squares line is fit to the points $\{(X_k, Y_k)\}$.

Data Linearization Method for $y = Ce^{Ax}$

$$\left(\sum_{k=1}^{N} X_k^2\right)A + \left(\sum_{k=1}^{N} X_k\right)B = \sum_{k=1}^{N} X_k Y_k \ ,$$

$$\left(\sum_{k=1}^{N} X_k\right)A + NB = \sum_{k=1}^{N} Y_k \ .$$

After A and B have been found, the parameter C in equation (1) is computed:

$$C = e^{B} \ .$$

An example

Use the data linearization method and find the exponential fit $y = Ce^{Ax}$ for the five data points $(0, 1.5)$, $(1, 2.5)$, $(2, 3.5)$, $(3, 5.0)$, and $(4, 7.5)$.

Apply the transformation to the original points and obtain

$$\{(X_k, Y_k)\} = \{(0, \ln(1.5)), (1, \ln(2.5)), (2, \ln(3.5)), (3, \ln(5.0)), (4, \ln(7.5))\}$$

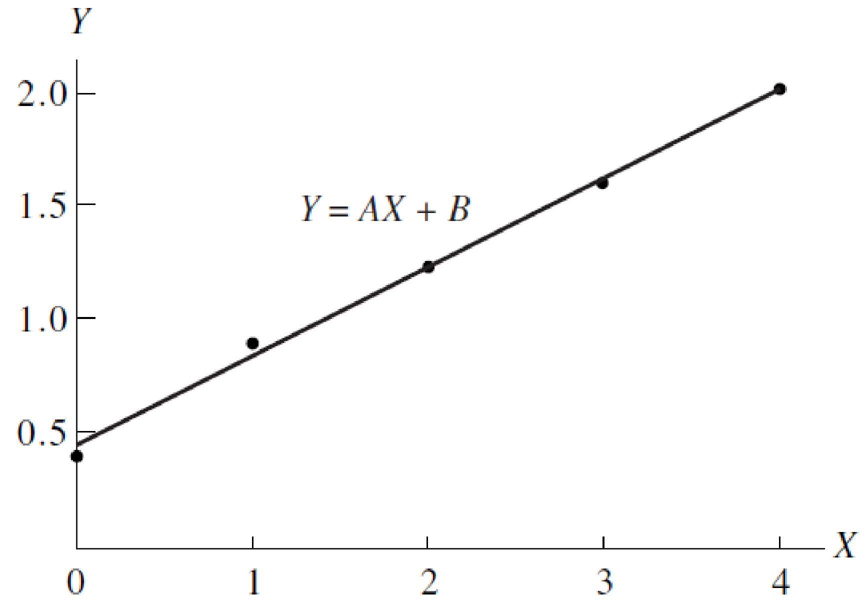
$$= \{(0, 0.40547), (1, 0.91629), (2, 1.25276), (3, 1.60944), (4, 2.01490)\}.$$

Table 5.4 Obtaining Coefficients of the Normal Equations for the Transformed Data Points $\{(X_k, Y_k)\}$

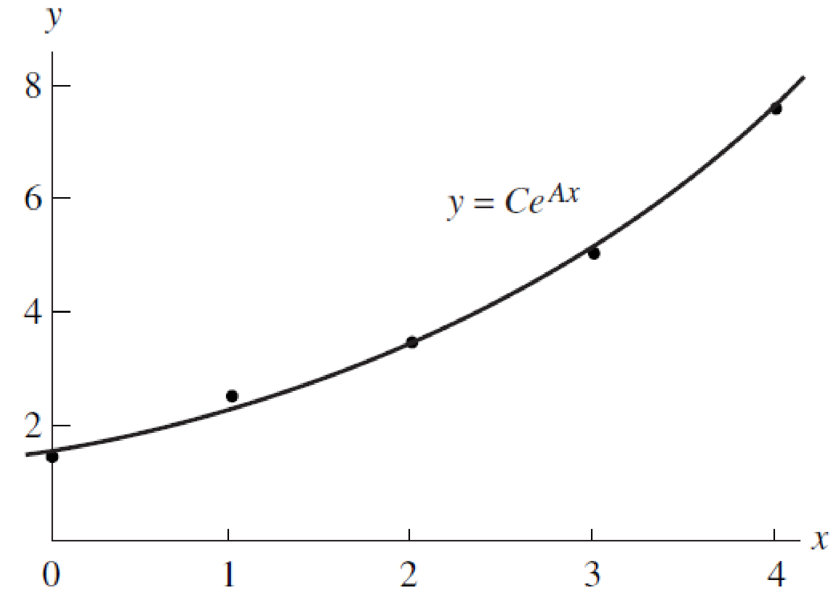
x_k	y_k	X_k	$Y_k = \ln(y_k)$	X_k^2	$X_k Y_k$
0.0	1.5	0.0	0.405465	0.0	0.000000
1.0	2.5	1.0	0.916291	1.0	0.916291
2.0	3.5	2.0	1.252763	4.0	2.505526
3.0	5.0	3.0	1.609438	9.0	4.828314
4.0	7.5	4.0	2.014903	16.0	8.059612
		10.0 $= \sum X_k$	6.198860 $= \sum Y_k$	30.0 $= \sum X_k^2$	16.309743 $= \sum X_k Y_k$

An example

$$Y = 0.391202X + 0.457367.$$



$$y = 1.579910e^{0.3912023x}$$



Nonlinear Least-Squares Method for $y = Ce^{Ax}$

Suppose that we are given the points $(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)$ and want to fit an exponential curve of the form

$$y = Ce^{Ax} .$$

The nonlinear least-squares procedure requires that we find a minimum of

$$E(A, C) = \sum_{k=1}^{N} (Ce^{Ax_k} - y_k)^2 . \tag{12}$$

The partial derivatives of $E(A, C)$ with respect to A and C are

$$\frac{\partial E}{\partial A} = 2 \sum_{k=1}^{N} (Ce^{Ax_k} - y_k)(Cx_k e^{Ax_k}) \tag{13}$$

and

$$\frac{\partial E}{\partial C} = 2 \sum_{k=1}^{N} (Ce^{Ax_k} - y_k)(e^{Ax_k}) . \tag{14}$$

Nonlinear Least-Squares Method for $y = Ce^{Ax}$

When the partial derivatives in (13) and (14) are set equal to zero and then simplified, the resulting normal equations are

$$
\begin{aligned}
&C \sum_{k=1}^{N} x_k e^{2Ax_k} - \sum_{k=1}^{N} x_k y_k e^{Ax_k} = 0\,, \\
(15) \qquad & \\
&C \sum_{k=1}^{N} e^{Ax_k} - \sum_{k=1}^{N} y_k e^{Ax_k} = 0\,.
\end{aligned}
$$

- The equations in (15) are nonlinear in the unknowns *A* and *C* and can be solved using Newton’s method. This is a time-consuming computation and the iteration involved requires good starting values for *A* and *C*.
- Many software packages have a built-in minimization subroutine for functions of several variables that can be used to minimize *E*(*A*,*C*) directly, For example, the Nelder-Mead simplex algorithm can be used to minimize (12) directly and bypass the need for equations (13) through (15).

An example

Example 5.5. Use the least-squares method and determine the exponential fit $y = Ce^{Ax}$ for the five data points $(0, 1.5), (1, 2.5), (2, 3.5), (3, 5.0)$, and $(4, 7.5)$.

$$E(A, C) = (C - 1.5)^2 + (Ce^A - 2.5)^2 + (Ce^{2A} - 3.5)^2 \\ + (Ce^{3A} - 5.0)^2 + (Ce^{4A} - 7.5)^2 .$$

We use the fmins command in MATLAB to approximate the values of A and C that minimize $E(A, C)$. First we define $E(A, C)$ as an M-file in MATLAB.

```
function z=E(u)
A=u(1);
C=u(2);
z=(C-1.5).^2+(C.*exp(A)-2.5).^2+(C.*exp(2*A)-3.5).^2+...
   (C.*exp(3*A)-5.0).^2+(C.*exp(4*A)-7.5).^2;
```

Using the fmins command in the MATLAB Command Window and the initial values $A = 1.0$ and $C = 1.0$, we find

```
>>fmins('E',[1 1])
ans =
   0.38357046980073 1.61089952247928
```

$$y = 1.6108995e^{0.3835705}$$

An example: comparison

Table 5.5 Comparison of the Two Exponential Fits

x_k	y_k	$1.5799e^{0.39120x}$	$1.6109e^{0.38357x}$
0.0	1.5	1.5799	1.6109
1.0	2.5	2.3363	2.3640
2.0	3.5	3.4548	3.4692
3.0	5.0	5.1088	5.0911
4.0	7.5	7.5548	7.4713
5.0		11.1716	10.9644
6.0		16.5202	16.0904
7.0		24.4293	23.6130
8.0		36.1250	34.6527
9.0		53.4202	50.8535
10.0		78.9955	74.6287

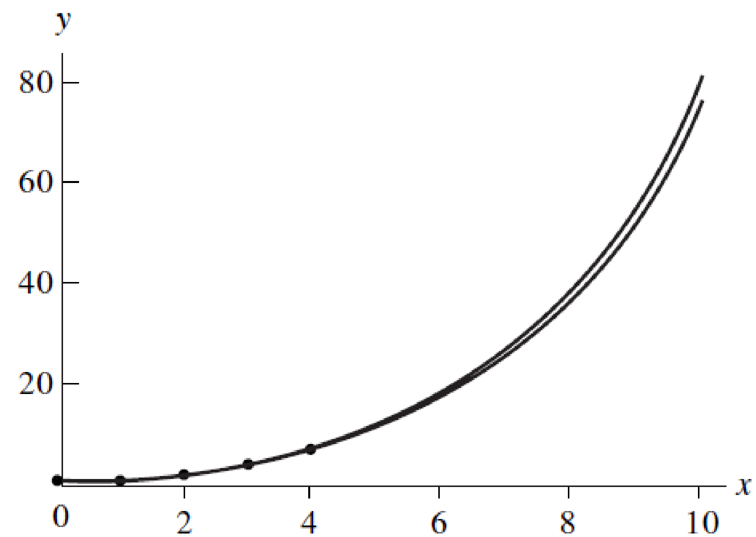


Figure 5.6 A graphical comparison of the two exponential curves

- There is a slight difference in the coefficients. The approximations differ by no more than 2% over the interval [0, 4].
- If there is a normal distribution of the errors in the data, nonlinear least-squares method is usually the preferred choice.
- When extrapolation is made beyond the range of the data, the two solutions will diverge and the discrepancy increases to about 6% when $x = 10$.

Transformations for Data Linearization

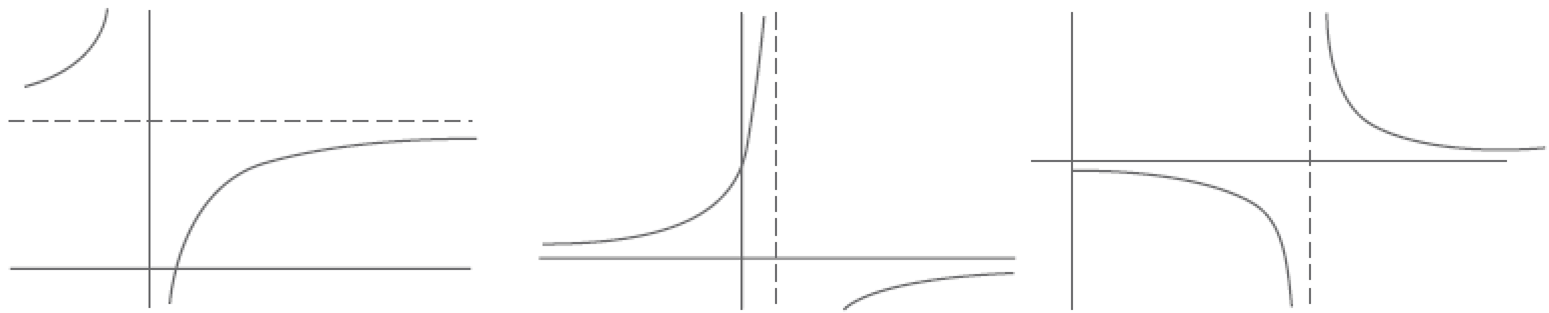
- The technique of data linearization has been used by scientists to fit curves such as y = $Ce^{(Ax)}$, $y = A \ln(x) + B$, and $y = A/x + B$.
- Once the curve has been chosen, a suitable transformation of the variables must be found so that a linear relation is obtained.
- For example, $y = D/(x + C)$ is transformed into a linear problem $Y = AX + B$ by using the change of variables $X = xy$, $Y = y$, $C = -1/A$, and $D = -B/A$.

Transformations for Data Linearization

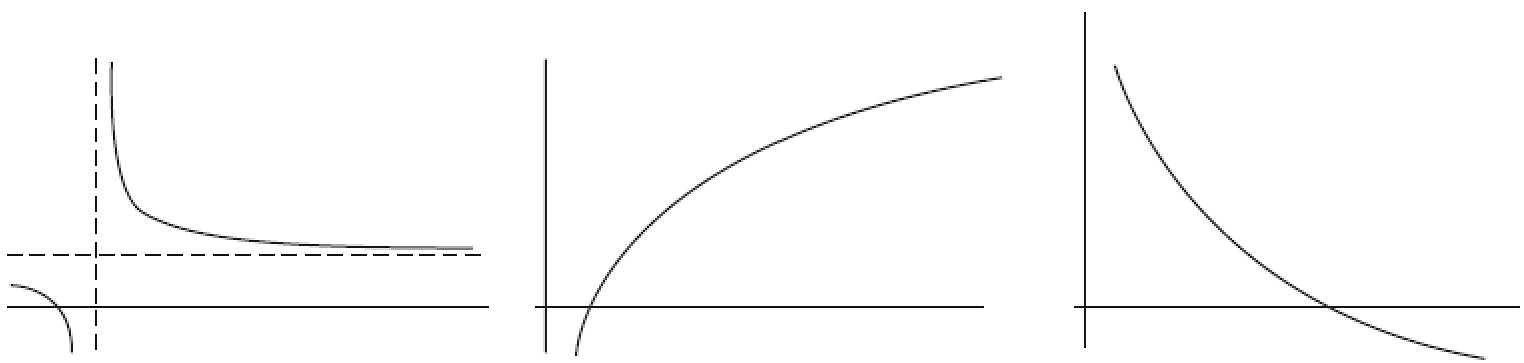
Table 5.6 Change of Variable(s) for Data Linearization

Function, $y = f(x)$	Linearized form, $Y = AX + B$	Change of variable(s) and constants
$y = \frac{A}{x} + B$	$y = A\frac{1}{x} + B$	$X = \frac{1}{x},\ Y = y$
$y = \frac{D}{x + C}$	$y = \frac{-1}{C}(xy) + \frac{D}{C}$	$X = xy,\ Y = y$ $C = \frac{-1}{A},\ D = \frac{-B}{A}$
$y = \frac{1}{Ax + B}$	$\frac{1}{y} = Ax + B$	$X = x,\ Y = \frac{1}{y}$
$y = \frac{x}{Ax + B}$	$\frac{1}{y} = A\frac{1}{x} + B$	$X = \frac{1}{x},\ Y = \frac{1}{y}$
$y = A\ln(x) + B$	$y = A\ln(x) + B$	$X = \ln(x),\ Y = y$
$y = Ce^{Ax}$	$\ln(y) = Ax + \ln(C)$	$X = x,\ Y = \ln(y)$ $C = e^B$
$y = Cx^A$	$\ln(y) = A\ln(x) + \ln(C)$	$X = \ln(x),\ Y = \ln(y)$ $C = e^B$
$y = (Ax + B)^{-2}$	$y^{-1/2} = Ax + B$	$X = x,\ Y = y^{-1/2}$
$y = Cxe^{-Dx}$	$\ln\left(\frac{y}{x}\right) = -Dx + \ln(C)$	$X = x,\ Y = \ln\left(\frac{y}{x}\right)$ $C = e^B,\ D = -A$
$y = \frac{L}{1 + Ce^{Ax}}$	$\ln\left(\frac{L}{y} - 1\right) = Ax + \ln(C)$	$X = x,\ Y = \ln\left(\frac{L}{y} - 1\right)$ $C = e^B$ and L is a constant that must be given

Transformations for Data Linearization



$y = \frac{A}{x} + B;\ A = -3, B = 4$ $y = \frac{D}{x + C};\ D = -1, C = \frac{-1}{4}$ $y = \frac{1}{Ax + B};\ A = 2, B = -3$



$y = \frac{x}{A + Bx};\ A = \frac{-1}{2}, B = 1$ $y = A \ln(x) + B;\ A = 2, B = \frac{3}{2}$ $y = A \ln(x) + B;\ A = -2, B = 2$

Figure 5.7 Possibilities for the curves used in “data linearization.”

Transformations for Data Linearization

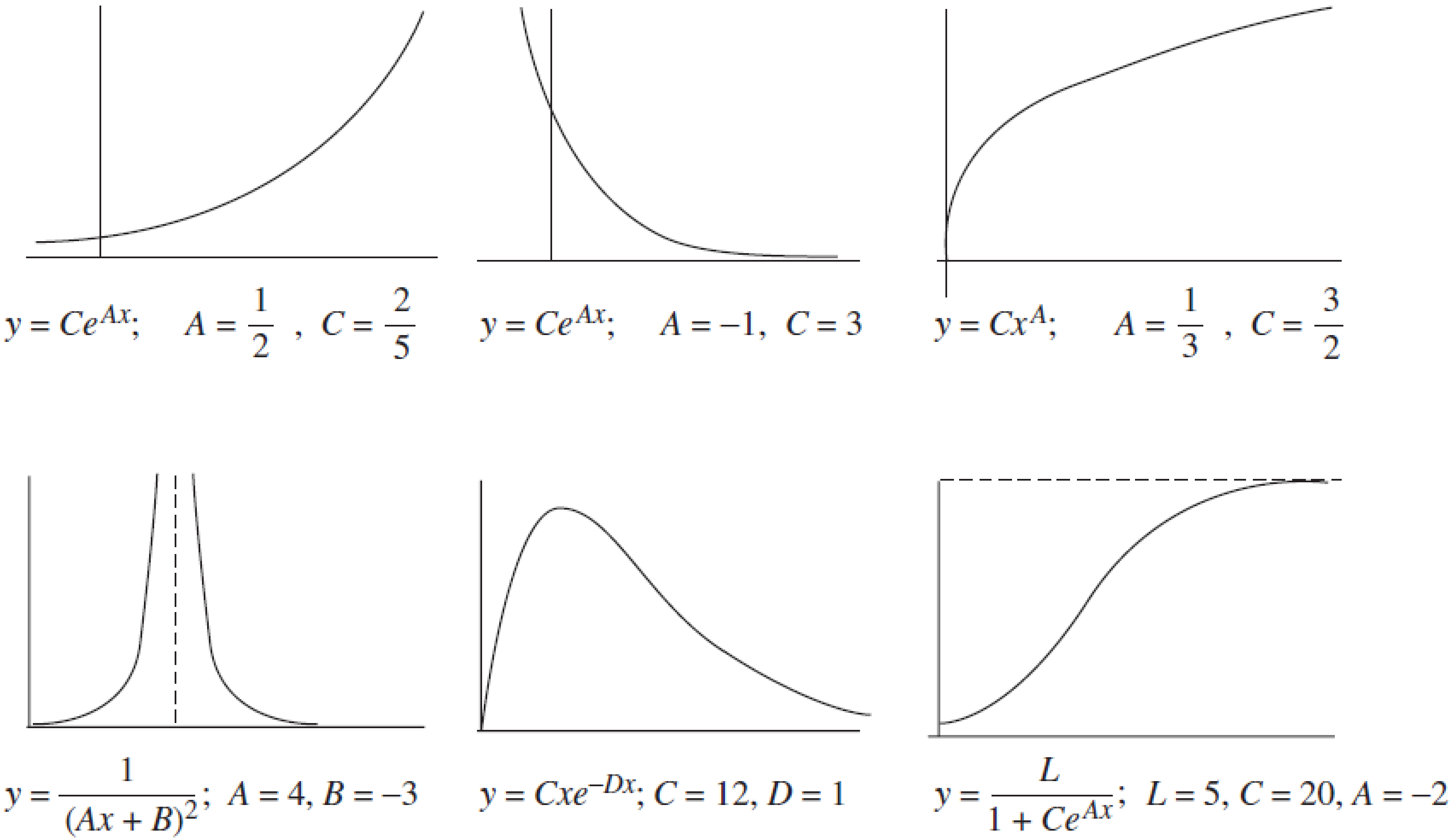


Figure 5.7 Possibilities for the curves used in “data linearization.”

Linear Least Squares

- Suppose that N data points $\{(x_k, y_k)\}$ and a set of M linear independent functions $\{f_j(x)\}$ are given. We want to find M coefficients $\{c_j\}$ so that the function $f(x)$ given by the linear combination

$$f(x) = \sum_{j=1}^{M} c_j f_j(x)$$

will minimize the sum of squares of the errors:

$$E(c_1, c_2, \ldots, c_M) = \sum_{k=1}^{N} (f(x_k) - y_k)^2 = \sum_{k=1}^{N} \left(\left(\sum_{j=1}^{M} c_j f_j(x_k) \right) - y_k \right)^2$$

Linear Least Squares

For E to be minimized it is necessary that each partial derivatives be zero (i.e., $\partial E/\partial c_i = 0$ for $i = 1, 2, \ldots, M$), and this results in the system of equations

$$\sum_{k=1}^{N}\left(\left(\sum_{j=1}^{M} c_j f_j(x_k)\right) - y_k\right)\big(f_i(x_k)\big) = 0 \quad \text{for} \quad i = 1, 2, \ldots, M. \tag{20}$$

Interchanging the order of the summations in (20) will produce an $M \times M$ system of linear equations where the unknowns are the coefficients $\{c_j\}$. They are called the normal equations:

$$\sum_{j=1}^{M}\left(\sum_{k=1}^{N} f_i(x_k) f_j(x_k)\right) c_j = \sum_{k=1}^{N} f_i(x_k)\, y_k \quad \text{for} \quad i = 1, 2, \ldots, M. \tag{21}$$

Matrix Formulation

Although (21) is easily recognized as a system of M linear equations in M unknows, one must be clever so that wasted computations are not performed when writing the system in matrix notation. The key is to write down the matrices F and F' as follows:

$$F = \begin{bmatrix} f_1(x_1) & f_2(x_1) & \cdots & f_M(x_1) \\ f_1(x_2) & f_2(x_2) & \cdots & f_M(x_2) \\ f_1(x_3) & f_2(x_3) & \cdots & f_M(x_3) \\ \vdots & \vdots & & \vdots \\ f_1(x_N) & f_2(x_N) & \cdots & f_M(x_N) \end{bmatrix},$$

$$F' = \begin{bmatrix} f_1(x_1) & f_1(x_2) & f_1(x_3) & \cdots & f_1(x_N) \\ f_2(x_1) & f_2(x_2) & f_2(x_3) & \cdots & f_2(x_N) \\ \vdots & \vdots & \vdots & & \vdots \\ f_M(x_1) & f_M(x_2) & f_M(x_3) & \cdots & f_M(x_N) \end{bmatrix}.$$

Matrix Formulation

Consider the product of F' and the column matrix Y:

$$F'Y = \begin{bmatrix} f_1(x_1) & f_1(x_2) & f_1(x_3) & \cdots & f_1(x_N) \\ f_2(x_1) & f_2(x_2) & f_2(x_3) & \cdots & f_2(x_N) \\ \vdots & \vdots & \vdots & & \vdots \\ f_M(x_1) & f_M(x_2) & f_M(x_3) & \cdots & f_M(x_N) \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix}.$$

(23) $$\sum_{k=1}^{N} f_i(x_k)\, y_k = \mathrm{row_i}\boldsymbol{F}' \cdot [y_1\ y_2\ \cdots y_N]'$$

$$F'F = \begin{bmatrix} f_1(x_1) & f_1(x_2) & f_1(x_3) & \cdots & f_1(x_N) \\ f_2(x_1) & f_2(x_2) & f_2(x_3) & \cdots & f_2(x_N) \\ \vdots & \vdots & \vdots & & \vdots \\ f_M(x_1) & f_M(x_2) & f_M(x_3) & \cdots & f_M(x_N) \end{bmatrix} \begin{bmatrix} f_1(x_1) & f_2(x_1) & \cdots & f_M(x_1) \\ f_1(x_2) & f_2(x_2) & \cdots & f_M(x_2) \\ f_1(x_3) & f_2(x_3) & \cdots & f_M(x_3) \\ \vdots & \vdots & & \vdots \\ f_1(x_N) & f_2(x_N) & \cdots & f_M(x_N) \end{bmatrix}$$

(24) $$\sum_{k=1}^{N} f_i(x_k) f_j(x_k) = f_i(x_1) f_j(x_1) + f_i(x_2) f_j(x_2) + \cdots + f_i(x_N) f_j(x_N)$$

When M is small, a computationally efficient way to calculate the linear least-squares coefficients for (18) is to store the matrix F, compute $F'F$, and $F'Y$ and then solve the linear system

(25) $F'FC = F'Y$ for the coefficient matrix C.

Polynomial Fitting

- When the foregoing method is adapted to using the functions $\{ f_j(x) = x^{j-1}\}$ and the index of summation ranges from $j = 1$ to $j = M + 1$, the function $f(x)$ will be a polynomial of degree M:

$$f(x) = c_1 + c_2 x + c_3 x^2 + \cdots + c_{M+1} x^M.$$

Theorem (Least-Squares Parabola). Suppose that $\{(x_k, y_k)\}_{k=1}^{N}$ are N points, where the abscissas are distinct. The coefficient of the least-squares parabola

$$y = f(x) = Ax^2 + Bx + C \tag{27}$$

are the solution values A, B, and C of the linear system

$$\begin{aligned}
\left(\sum_{k=1}^{N} x_k^4\right) A + \left(\sum_{k=1}^{N} x_k^3\right) B + \left(\sum_{k=1}^{N} x_k^2\right) C &= \sum_{k=1}^{N} y_k x_k^2 \ , \\
\left(\sum_{k=1}^{N} x_k^3\right) A + \left(\sum_{k=1}^{N} x_k^2\right) B + \left(\sum_{k=1}^{N} x_k\right) C &= \sum_{k=1}^{N} y_k x_k \ , \\
\left(\sum_{k=1}^{N} x_k^2\right) A + \left(\sum_{k=1}^{N} x_k\right) B + NC &= \sum_{k=1}^{N} y_k \ .
\end{aligned} \tag{28}$$

Polynomial Fitting

Proof. The coefficients *A*, *B*, and *C* will minimize the quantity:

(29)
$$E(A,B,C) = \sum_{k=1}^{N} (Ax_k^2 + Bx_k + C - y_k)^2 \ .$$

The partial derivatives $\partial E/\partial A, \partial E/\partial B,$ and $\partial E/\partial C$ must all be zero. This results in

(30)
$$0 = \frac{\partial E(A,B,C)}{\partial A} = 2\sum_{k=1}^{N} \left(Ax_k^2 + Bx_k + C - y_k\right)^1 (x_k^2) \ ,$$

$$0 = \frac{\partial E(A,B,C)}{\partial B} = 2\sum_{k=1}^{N} \left(Ax_k^2 + Bx_k + C - y_k\right)^1 (x_k) \ ,$$

$$0 = \frac{\partial E(A,B,C)}{\partial C} = 2\sum_{k=1}^{N} \left(Ax_k^2 + Bx_k + C - y_k\right)^1 (1) \ .$$

An example

Example 5.6. Find the least-squares parabola for the four points $(-3, 3)$, $(0, 1)$, $(2, 1)$, and $(4, 3)$.

Table 5.7 Obtaining the Coefficients for the Least-Squares Parabola of Example 5.6

x_k	y_k	x_k^2	x_k^3	x_k^4	$x_k y_k$	$x_k^2 y_k$
−3	3	9	−27	81	−9	27
0	1	0	0	0	0	0
2	1	4	8	16	2	4
4	3	16	64	256	12	48
3	8	29	45	353	5	79

The linear system (28) for finding A, B, and C becomes

$$\begin{aligned} 353A + 45B + 29C &= 79 \\ 45A + 29B + 3C &= 5 \\ 29A + 3B + 4C &= 8. \end{aligned}$$

$$y = \frac{585}{3278}x^2 - \frac{631}{3278}x + \frac{1394}{1639} = 0.178462x^2 - 0.192495x + 0.850519.$$

An example

Example 5.6. Find the least-squares parabola for the four points $(-3, 3)$, $(0, 1)$, $(2, 1)$, and $(4, 3)$.

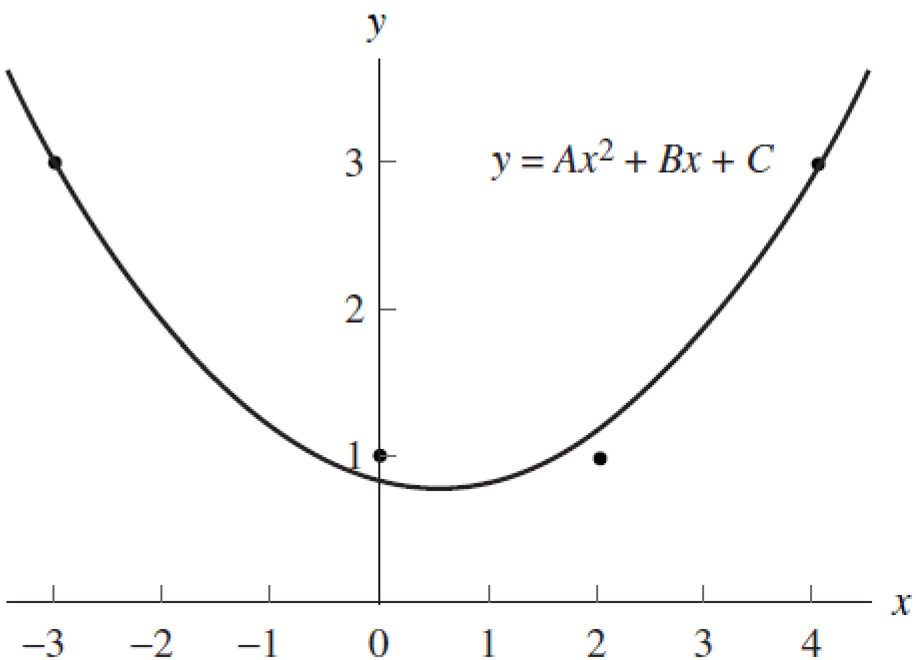


Figure 5.8 The least-squares parabola for Example 5.6.

Polynomial Wiggle

- It is tempting to use a least-squares polynomial to fit data that are nonlinear.
- But if the data do not exhibit a polynomial nature, the resulting curve may exhibit large oscillations. This phenomenon, called polynomial wiggle, becomes more pronounced with higher-degree polynomials.
- For this reason, we seldom use a polynomial of degree 6 or above unless it is known that the true function that we are working with is a polynomial.
- See an example

Polynomial Wiggle

- Consider six data points (0.25, 23.1), (1.0, 1.68), (1.5, 1.0), (2.0, 0.84), (2.4, 0.826), and (5.0, 1.2576), generated by $f(x) = 1.44/x^2 + 0.24x$

- The result of curve fitting with the least-squares polynomials

$$P_2(x) = 22.93 - 16.96x + 2.553x^2 \,,$$

$$P_3(x) = 33.04 - 46.51x + 19.51x^2 - 2.296x^3 \,,$$

$$P_4(x) = 39.92 - 80.93x + 58.39x^2 - 17.15x^3 + 1.680x^4 \,,$$

$$P_5(x) = 46.02 - 118.1x + 119.4x^2 - 57.51x^3 + 13.03x^4 - 1.085x^5$$

Polynomial Wiggle

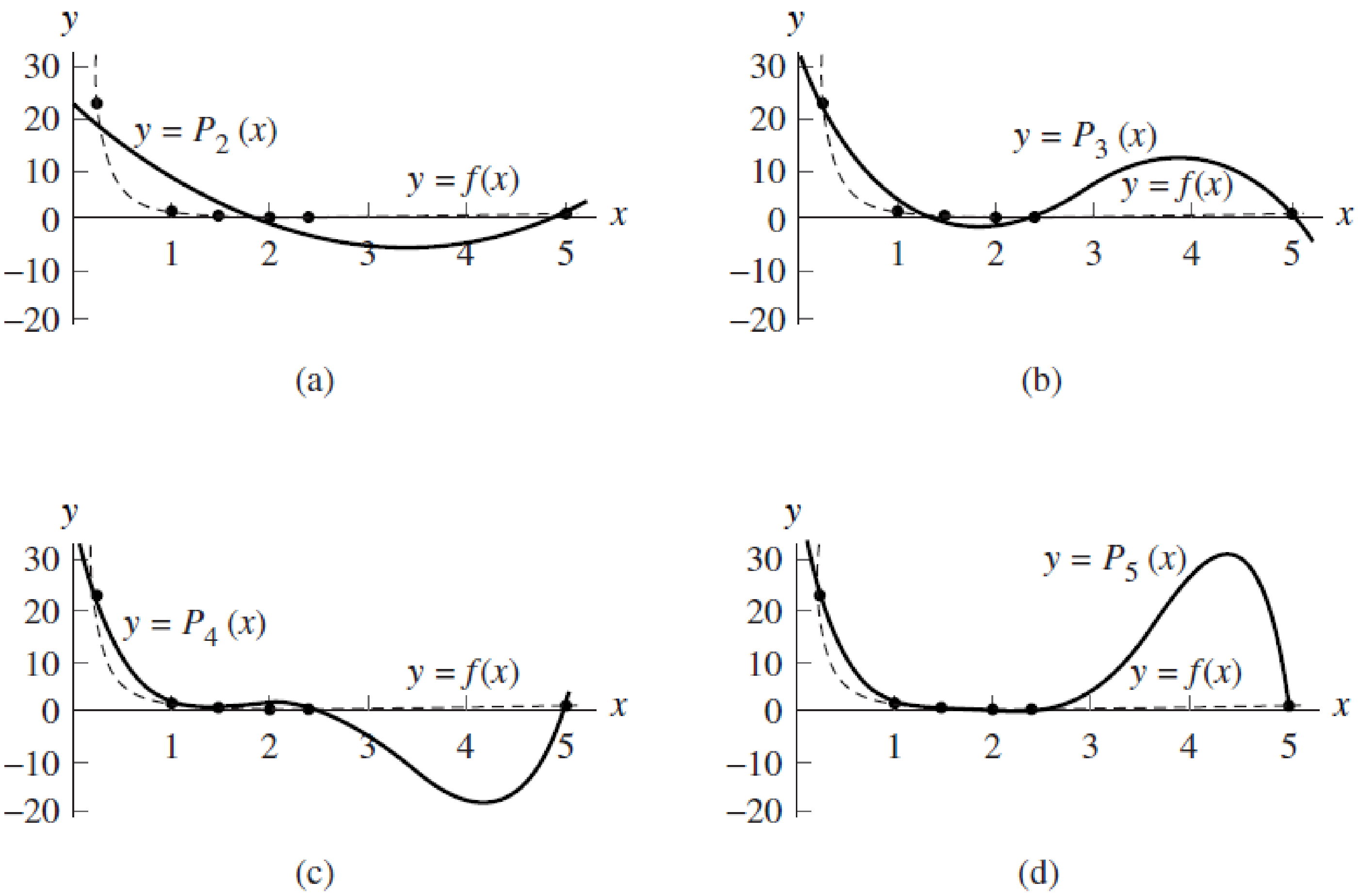


Figure 5.9 (a) Using $P_2(x)$ to fit data. (b) Using $P_3(x)$ to fit data. (c) Using $P_4(x)$ to fit data. (d) Using $P_5(x)$ to fit data.

Matlab Code

Program 5.2 (Least-Squares Polynomial). To construct the least-squares polynomial of degree M of the form

$$P_M(x) = c_1 + c_2 x + c_3 x^2 + \cdots + c_M x^{M-1} + c_{M+1} x^M$$

that fits the N data points $\{(x_k, y_k)\}_{k=1}^{N}$.

```
function C = lspoly(X,Y,M)
%Input    - X is the 1xn abscissa vector
%         - Y is the 1xn ordinate vector
%         - M is the degree of the least-squares polynomial
% Output - C is the coefficient list for the polynomial
n=length(X);
B=zeros(1:M+1);
F=zeros(n,M+1);
%Fill the columns of F with the powers of X
for k=1:M+1
   F(:,k)=X'.^(k-1);
end
%Solve the linear system from (25)
A=F'*F;
B=F'*Y';
C=A\B;
C=flipud(C);
```

Interpolation by Spline Functions

Interpolation by Spline Functions

- Polynomial interpolation for a set of $N + 1$ points $\{(x_k , y_k)\}$ is frequently unsatisfactory. A polynomial of degree N can have $N - 1$ relative maxima and minima, and the graph can **wiggle** in order to pass through the points.

- Another method is to piece together the graphs of lower-degree polynomials $S_k(x)$ and interpolate between the successive nodes (x_k , y_k) and (x_{k+1}, y_{k+1}).

- The two adjacent portions of the curve $y = S_k(x)$ and $y = S_{k+1}(\mathrm{x})$, which lie above $[x_k , x_{k+1}]$ and $[x_{k+1}, x_{k+2}]$, respectively, pass through the common ***knot*** (x_{k+1}, y_{k+1}).

Interpolation by Spline Functions

- The two portions of the graph are tied together at the knot (x_{k+1}, y_{k+1}), and the set of functions $\{S_k(x)\}$ forms a piecewise polynomial curve, which is denoted by $S(x)$.

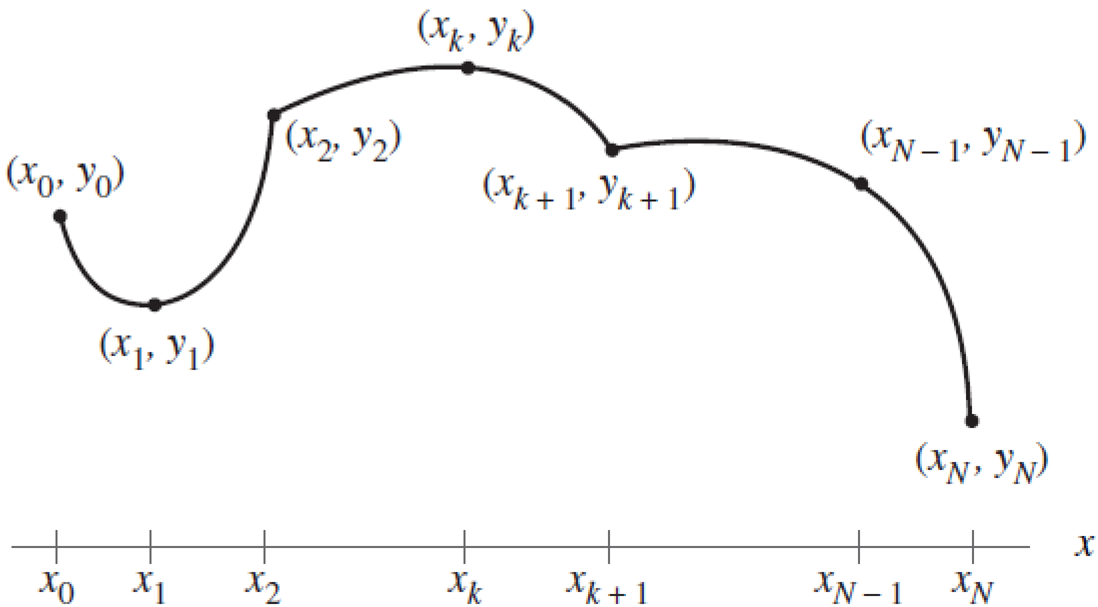
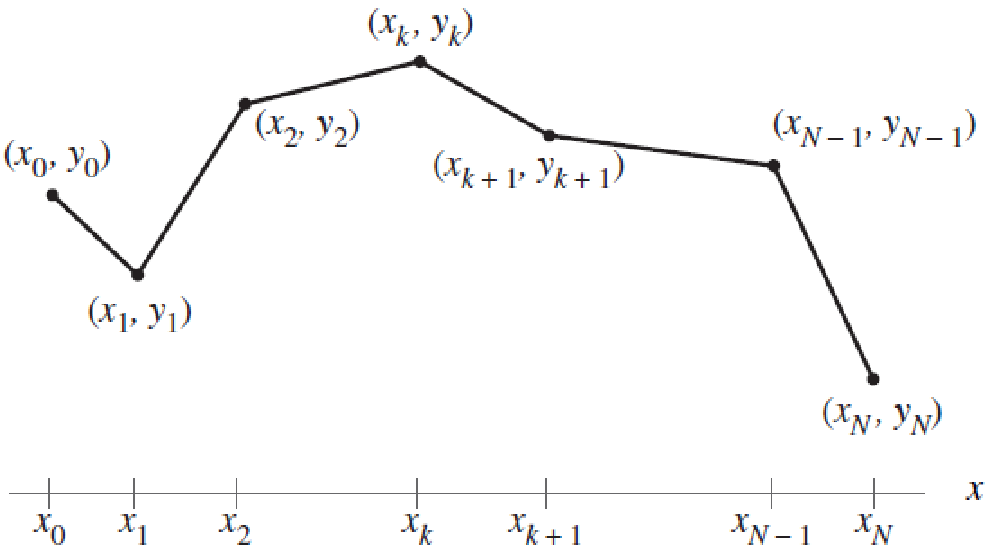


Figure 5.10 Piecewise polynomial interpolation.

Piecewise Linear Interpolation

- The simplest polynomial to use, a polynomial of degree 1, produces a polygonal path that consists of line segments that pass through the points.
- The Lagrange polynomial can be used to represent this piecewise linear curve:

$$S_k(x) = y_k \frac{x - x_{k+1}}{x_k - x_{k+1}} + y_{k+1} \frac{x - x_k}{x_{k+1} - x_k} \quad \text{for} \quad x_k \le x \le x_{k+1}.$$



The resulting curve looks like a broken line!

Figure 5.11 Piecewise linear interpolation (a linear spline).

Piecewise Linear Interpolation

- An equivalent expression can be obtained if we use the point-slope formula for a line segment:

$$S_k(x) = y_k + d_k(x - x_k)\,,$$
$$d_k = (y_{k+1} - y_k)/(x_{k+1} - x_k)\,.$$

$$S(x) = \begin{cases} y_0 + d_0(x - x_0) & \text{for } x \text{ in } [x_0, x_1]\,, \\ y_1 + d_1(x - x_1) & \text{for } x \text{ in } [x_1, x_2]\,, \\ \vdots & \vdots \\ y_k + d_k(x - x_k) & \text{for } x \text{ in } [x_k, x_{k+1}]\,, \\ \vdots & \vdots \\ y_{N-1} + d_{N-1}(x - x_{N-1}) & \text{for } x \text{ in } [x_{N-1}, x_N]\,, \end{cases}$$

- The techniques can be extended to higher-order polynomials.
- Problem with quadratic polynomial (constructed on each subinterval $[x_{2k}, x_{2k+2}]$): the curvature at the even nodes x_{2k} changes abruptly.

Piecewise Cubic Splines

- For piecewise cubic polynomials, both the first and second derivatives can be made continuous.
- The continuity of first derivative means that the graph $y = S(x)$ will not have sharp corners. The continuity of second derivatives means that the radius of curvature is defined at each point.
- Useful in CAD, CAM, and computer graphics systems.

Definition 5.1 Suppose that $\{(x_k, y_k)\}_{k=0}^{N}$ are $N+1$ points, where $a = x_0 < x_1 < \cdots < x_N = b$. The function $S(x)$ is called a ***cubic spline*** if there exist N cubic polynomials $S_k(x)$ with coefficients $s_{k,0}$, $s_{k,1}$, $s_{k,2}$ and $s_{k,3}$ that satisfy the following properties:

I. $S(x) = S_k(x) = s_{k,0} + s_{k,1}(x - x_k) + s_{k,2}(x - x_k)^2 + s_{k,3}(x - x_k)^3$ for $x \in [x_k, x_{k+1}]$ and $k = 0, 1, \ldots, N-1$.

II. $S(x_k) = y_k$ for $k = 0, 1, \ldots, N$.

III. $S_k(x_{k+1}) = S_{k+1}(x_{k+1})$ for $k = 0, 1, \ldots, N-2$.

IV. $S'_k(x_{k+1}) = S'_{k+1}(x_{k+1})$ for $k = 0, 1, \ldots, N-2$.

V. $S''_k(x_{k+1}) = S''_{k+1}(x_{k+1})$ for $k = 0, 1, \ldots, N-2$.

Existence of Cubic Splines

- Each cubic polynomial $S_k(x)$ has four unknown constants ($s_{k,0}$, $s_{k,1}$, $s_{k,2}$, and $s_{k,3}$); hence $4N$ coefficients to be determined, or $4N$ degrees of freedom or conditions that must be specified.

- The data points supply $N + 1$ conditions, and properties III, IV, and V each supply $N - 1$ conditions. Hence, $4N - 2$ conditions are specified.

- This leaves us two additional degrees of freedom, called ***endpoint constraints***: they will involve either first or second derivatives at x_0 and x_N,

Construction of Cubic Splines

- Since $S(x)$ is piecewise cubic, its second derivative is piecewise linear on $[x_0, x_N]$.

$$S_k''(x) = S''(x_k)\frac{x - x_{k+1}}{x_k - x_{k+1}} + S''(x_{k+1})\frac{x - x_k}{x_{k+1} - x_k} .$$

Use $m_k = S''(x_k), m_{k+1} = S''(x_{k+1})$, and $h_k = x_{k+1} - x_k$

$$S_k''(x) = \frac{m_k}{h_k}(x_{k+1} - x) + \frac{m_{k+1}}{h_k}(x - x_k)$$

- Integrating twice results

$$S_k(x) = \frac{m_k}{6h_k}(x_{k+1} - x)^3 + \frac{m_{k+1}}{6h_k}(x - x_k)^3 + p_k(x_{k+1} - x) + q_k(x - x_k).$$

- Now we need to get p_k and q_k

Construction of Cubic Splines

- Substituting x_k and x_{k+1} into last equation, and using $y_k = S_k(x_k)$ and $y_{k+1} = S_k(x_{k+1})$

$$y_k = \frac{m_k}{6} h_k^2 + p_k h_k \quad \text{and} \quad y_{k+1} = \frac{m_{k+1}}{6} h_k^2 + q_k h_k \,.$$

- We can solve for p_k and q_k from last two equations, and get

$$\begin{aligned} S_k(x) = &-\frac{m_k}{6h_k}(x_{k+1} - x)^3 + \frac{m_{k+1}}{6h_k}(x - x_k)^3 \\ &+ \left(\frac{y_k}{h_k} - \frac{m_k h_k}{6}\right)(x_{k+1} - x) + \left(\frac{y_{k+1}}{h_k} - \frac{m_{k+1} h_k}{6}\right)(x - x_k)\,. \end{aligned}$$

- Now the only unknown coefficients are m_k.

- How to find these coefficients?
- We need to use the derivative of the last equation.

Construction of Cubic Splines

$$S'_k(x) = -\frac{m_k}{2h_k}(x_{k+1} - x)^2 + \frac{m_{k+1}}{2h_k}(x - x_k)^2$$

$$-\left(\frac{y_k}{h_k} - \frac{m_k h_k}{6}\right) + \frac{y_{k+1}}{h_k} - \frac{m_{k+1} h_k}{6}\ .$$

- Using the value at x_k, we can get

$$S'_k(x_k) = -\frac{m_k}{3}h_k - \frac{m_{k+1}}{6}h_k + d_k, \qquad \text{where} \quad d_k = \frac{y_{k+1} - y_k}{h_k}.$$

- Similarly, we can replace k by k-1, and get $S'_{k-1}(x)$ and evaluate it at x_k

$$S'_{k-1}(x_k) = \frac{m_k}{3}h_{k-1} + \frac{m_{k-1}}{6}h_{k-1} + d_{k-1}.$$

- Using property IV, we can get

(12) $$h_{k-1}m_{k-1} + 2(h_{k-1} + h_k)m_k + h_k m_{k+1} = u_k,$$

where $u_k = 6(d_k - d_{k-1})$ for $k = 1, 2, \ldots, N - 1$.

Construction of Cubic Splines

$$(12) \qquad h_{k-1}m_{k-1} + 2(h_{k-1} + h_k)m_k + h_k m_{k+1} = u_k,$$

where $u_k = 6(d_k - d_{k-1})$ for $k = 1, 2, \ldots, N-1$.

- System (12) is an underdetermined system of $N - 1$ linear equations involving $N + 1$ unknowns. Hence two additional equations must be supplied.

- Regardless of the particular strategy for endpoint constrain, we can rewrite equations 1 and $N - 1$ in (12) and obtain a tridiagonal linear system of the form $\boldsymbol{HM} = \boldsymbol{V}$,

$$(15) \qquad \begin{bmatrix} b_1 & c_1 & & & & \\ a_1 & b_2 & c_2 & & & \\ & & \ddots & & & \\ & & & a_{N-3} & b_{N-2} & c_{N-2} \\ & & & & a_{N-2} & b_{N-1} \end{bmatrix} \begin{bmatrix} m_1 \\ m_2 \\ \vdots \\ m_{N-2} \\ m_{N-1} \end{bmatrix} = \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_{N-2} \\ v_{N-1} \end{bmatrix}.$$

Construction of Cubic Splines

- The linear system in (15) is strictly diagonally dominant and has a unique solution. After the coefficients $\{m_k\}$ are determined, the spline coefficients can be computed

$$s_{k,0} = y_k, \qquad s_{k,1} = d_k - \frac{h_k(2m_k + m_{k+1})}{6},$$

$$s_{k,2} = \frac{m_k}{2}, \qquad s_{k,3} = \frac{m_{k+1} - m_k}{6h_k}.$$

- Each cubic polynomial $S_k(x)$ can be written in nested multiplication form for efficient computation:

$S_k(x) = \Big(\big(s_{k,3}w + s_{k,2}\big)w + s_{k,1}\Big)w + y_k$, where $w = x - x_k$ and $S_k(x)$ is used on the interval $x_k \le x \le x_{k+1}$.

Endpoint Constraints

Table 5.8 Endpoint Constraints for a Cubic Spline

Description of the strategy		Equations involving m_0 and m_N
(i)	*Clamped cubic spline:* specify $S'(x_0)$, $S'(x_n)$ (the "best choice" if the derivatives are known)	$m_0 = \frac{3}{h_0}(d_0 - S'(x_0)) - \frac{m_1}{2}$ $m_N = \frac{3}{h_{N-1}}(S'(x_N) - d_{N-1}) - \frac{m_{N-1}}{2}$
(ii)	*Natural cubic spline* (a "relaxed curve")	$m_0 = 0, m_N = 0$
(iii)	Extrapolate $S''(x)$ to the endpoints	$m_0 = m_1 - \frac{h_0(m_2 - m_1)}{h_1}$, $m_N = m_{N-1} + \frac{h_{N-1}(m_{N-1} - m_{N-2})}{h_{N-2}}$
(iv)	$S''(x)$ is constant near the endpoints	$m_0 = m_1, m_N = m_{N-1}$
(v)	Specify $S''(X)$ at each endpoint	$m_0 = S''(x_0), m_N = S''(x_N)$

Endpoint Constraints

- **Lemma 5.1** (**Clamped Spline**). There exists a unique cubic spline with the first derivative boundary conditions $S'(x_0)$ and $S'(x_N)$ specified.

$$\left(\frac{3}{2}h_0 + 2h_1\right)m_1 + h_1 m_2 = u_1 - 3(d_0 - S'(x_0))$$

$$h_{k-1}m_{k-1} + 2(h_{k-1} + h_k)m_k + h_k m_{k+1} = u_k \quad \text{for } k = 2, 3, \dots, N-2$$

$$h_{N-2}m_{N-2} + \left(2h_{N-2} + \frac{3}{2}h_{N-1}\right)m_{N-1} = u_{N-1} - 3(S'(x_N) - d_{N-1}).$$

Remark. The clamped spline involves slope at the ends. This spline can be visualized as the curve obtained when a flexible elastic rod is forced to pass through the data points, and the rod is clamped at each end with a fixed slope. This spline would be useful to a draftsman for drawing a smooth curve through several points.

Endpoint Constraints

- **Lemma 5.2** (**Natural Spline**). There exists a unique cubic spline with the free boundary conditions $S''(x_0) = 0$ and $S''(x_N) = 0$.

$$2(h_0 + h_1)m_1 + h_1 m_2 = u_1$$

$$h_{k-1}m_{k-1} + 2(h_{k-1} + h_k)m_k + h_k m_{k+1} = u_k \quad \text{for } k = 2, 3, \ldots, N-2$$

$$h_{N-2}m_{N-2} + 2(h_{N-2} + h_{N-1})m_{N-1} = u_{N-1}.$$

Remark. The natural spline is the curve obtained by forcing a flexible elastic rod through the data points but letting the slope at the ends be free to equilibrate to the position that minimizes the oscillatory behavior of the curve. It is useful for fitting a curve to experimental data that are significant to several significant digits.

Endpoint Constraints

- **Lemma 5.3** (**Extrapolated Spline**). There exists a unique cubic spline that uses extrapolation from the interior nodes at x_1 and x_2 to determine $S''(x_0)$ and extrapolation from the nodes at x_{N-1} and x_{N-2} to determine $S''(x_N)$.

$$\left(3h_0 + 2h_1 + \frac{h_0^2}{h_1}\right)m_1 + \left(h_1 - \frac{h_0^2}{h_1}\right)m_2 = u_1$$

$$h_{k-1}m_{k-1} + 2(h_{k-1} + h_k)m_k + h_k m_{k+1} = u_k \quad \text{for } k = 2, 3, \ldots, N-2$$

$$\left(h_{N-2} - \frac{h_{N-1}^2}{h_{N-2}}\right)m_{N-2} + \left(2h_{N-2} + 3h_{N-1} + \frac{h_{N-1}^2}{h_{N-2}}\right)m_{N-1} = u_{N-1}.$$

Remark. The extrapolated spline is equivalent to assuming that the end cubic is an extension of the adjacent cubic; that is, the spline forms a single cubic curve over the interval $[x_0, x_2]$ and another single cubic over the interval $[x_{N-2}, x_N]$.

Endpoint Constraints

- **Lemma 5.4** (**Parabolically Terminated Spline**). There exists a unique cubic spline that uses $S''(x) \equiv 0$ on the interval $[x_0, x_1]$ and $S''(x) \equiv 0$ on the interval $[x_{N-1}, x_N]$.

$$(3h_0 + 2h_1)m_1 + h_1 m_2 = u_1$$

$$h_{k-1}m_{k-1} + 2(h_{k-1} + h_k)m_k + h_k m_{k+1} = u_k \quad \text{for } k = 2, 3, \ldots, N-2$$

$$h_{N-2}m_{N-2} + (2h_{N-2} + 3h_{N-1})m_{N-1} = u_{N-1}.$$

Remark. The assumption that $S''(x) \equiv 0$ on the interval $[x_0, x_1]$ forces the cubic to degenerate to a quadratic over $[x_0, x_1]$, and a similar situation occurs over $[x_{N-1}, x_N]$.

Endpoint Constraints

- **Lemma 5.5** (**Endpoint Curvature-Adjusted Spline**). There exists a unique cubic spline with the second derivative boundary conditions $S''(x_0)$ and $S''(x_N)$ specified.

$$2(h_0 + h_1)m_1 + h_1 m_2 = u_1 - h_0 S''(x_0)$$

$$h_{k-1}m_{k-1} + 2(h_{k-1} + h_k)m_k + h_k m_{k+1} = u_k \quad \text{for } k = 2, 3, \ldots, N-2$$

$$h_{N-2}m_{N-2} + 2(h_{N-2} + h_{N-1})m_{N-1} = u_{N-1} - h_{N-1}S''(x_N).$$

Remark. Imposing values for $S''(a)$ and $S''(b)$ permits the practitioner to adjust the curvature at each endpoint.

An example

Find the clamped cubic spline that passes through (0, 0), (1, 0.5), (2, 2.0), and (3, 1.5) with the first derivatives boundary conditions $S'(0) = 0.2$ and $S'(3) = -1$.

First, compute the quantities

$$h_0 = h_1 = h_2 = 1$$

$$d_0 = (y_1 - y_0)/h_0 = (0.5 - 0.0)/1 = 0.5$$

$$d_1 = (y_2 - y_1)/h_1 = (2.0 - 0.5)/1 = 1.5$$

$$d_2 = (y_3 - y_2)/h_2 = (1.5 - 2.0)/1 = -0.5$$

$$u_1 = 6(d_1 - d_0) = 6(1.5 - 0.5) = 6.0$$

$$u_2 = 6(d_2 - d_1) = 6(-0.5 - 1.5) = -12.0\ .$$

Then use Lemma 5.1 and obtain the equations

$$\left(\frac{3}{2} + 2\right) m_1 + m_2 = 6.0 - 3(0.5 - 0.2) = 5.1\ ,$$

$$m_1 + \left(2 + \frac{3}{2}\right) m_2 = -12.0 - 3\big(-1.0 - (-0.5)\big) = -10.5.$$

An example

compute the solution $m_1 = 2.52$ and $m_2 = -3.72$. Now apply the equation in (i) of Table 5.8 to determine the coefficients m_0 and m_3:

$$m_0 = 3(0.5 - 0.2) - \frac{2.52}{2} = -0.36\,,$$

$$m_3 = 3(-1.0 + 0.5) - \frac{-3.72}{2} = 0.36$$

$$S_0(x) = 0.48x^3 - 0.18x^2 + 0.2x \qquad \text{for } 0 \le x \le 1$$

$$S_1(x) = -1.04(x-1)^3 + 1.26(x-1)^2 + 1.28(x-1) + 0.5 \qquad \text{for } 1 \le x \le 2$$

$$S_2(x) = 0.68(x-2)^3 - 1.86(x-2)^2 + 0.68(x-2) + 2.0 \qquad \text{for } 2 \le x \le 3$$

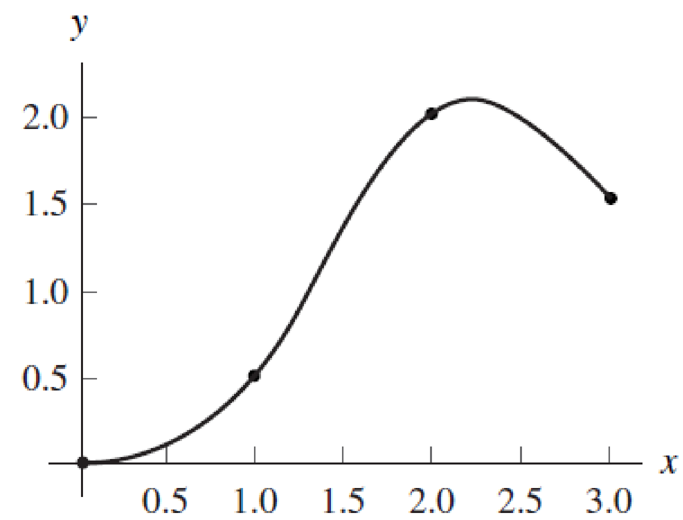


Figure 5.12 The clamped cubic spline with derivative boundary conditions: $S'(0) = 0.2$ and $S'(3) = -1$.

A practical feature of splines is the minimum of the oscillatory behavior that they possess.

Suitability of Cubic Splines

- A practical feature of splines is the minimum of the oscillatory behavior that they possess.
- Consequently, among all functions $f(x)$ that are twice continuously differentiable on [a, b] and interpolate a given set of data points $\{(x_k, y_k)\}_{k=0}^{N}$, the cubic spline has less wiggle.

Theorem 5.4 (Minimum Property of Cubic Splines). Assume that $f \in C^2[a, b]$ and $S(x)$ is the unique cubic spline interpolant for $f(x)$ that passes through the points $\{(x_k, f(x_k))\}_{k=0}^{N}$ and satisfies the clamped end conditions $S'(a) = f'(a)$ and $S'(b) = f'(b)$. Then

$$\int_a^b (S''(x))^2\,dx \le \int_a^b (f''(x))^2\,dx. \tag{23}$$

MATLAB Code

Program 5.3 (Clamped Cubic Spline). To construct and evaluate a clamped cubic spline interpolant $S(x)$ for the $N+1$ data points $\{(x_k, y_k)\}_{k=0}^{N}$.

```
function S=csfit(X,Y,dx0,dxn)

%Input  - X is the 1xn abscissa vector
%       - Y is the 1xn ordinate vector
%       - dx0 = S'(x0) first derivative boundary condition
%       - dxn = S'(xn) first derivative boundary condition
%Output - S: rows of S are the coefficients, in descending
%         order, for the cubic interpolants

N=length(X)-1;
H=diff(X);
D=diff(Y)./H;
A=H(2:N-1);
B=2*(H(1:N-1)+H(2:N));
C=H(2:N);
U=6*diff(D);

%Clamped spline endpoint constraints
B(1)=B(1)-H(1)/2;
U(1)=U(1)-3*(D(1)-dx0);
B(N-1)=B(N-1)-H(N)/2;
U(N-1)=U(N-1)-3*(dxn-D(N));
```

MATLAB Code

```
for k=2:N-1
   temp=A(k-1)/B(k-1);
   B(k)=B(k)-temp*C(k-1);
   U(k)=U(k)-temp*U(k-1);
end

M(N)=U(N-1)/B(N-1);

for k=N-2:-1:1
   M(k+1)=(U(k)-C(k)*M(k+2))/B(k);
end

M(1)=3*(D(1)-dx0)/H(1)-M(2)/2;
M(N+1)=3*(dxn-D(N))/H(N)-M(N)/2;

for k=0:N-1
   S(k+1,1)=(M(k+2)-M(k+1))/(6*H(k+1));
   S(k+1,2)=M(k+1)/2;
   S(k+1,3)=D(k+1)-H(k+1)*(2*M(k+1)+M(k+2))/6;
   S(k+1,4)=Y(k+1);
end
```

Fourier Series and Trigonometric Polynomials

Periodic functions

- For periodic functions

$$g(x + P) = g(x) \quad \text{for all } x.$$

- It will suffice to consider functions that have period 2π

- If $g(x)$ has period P, then $f(x) = g(Px/2\pi)$ will be periodic with period 2π.

$$f(x + 2\pi) = g\left(\frac{Px}{2\pi} + P\right) = g\left(\frac{Px}{2\pi}\right) = f(x).$$

- Hence we assume that $f(x)$ is a function that is periodic with period 2π

$$f(x + 2\pi) = f(x) \quad \text{for all } x.$$

Periodic functions

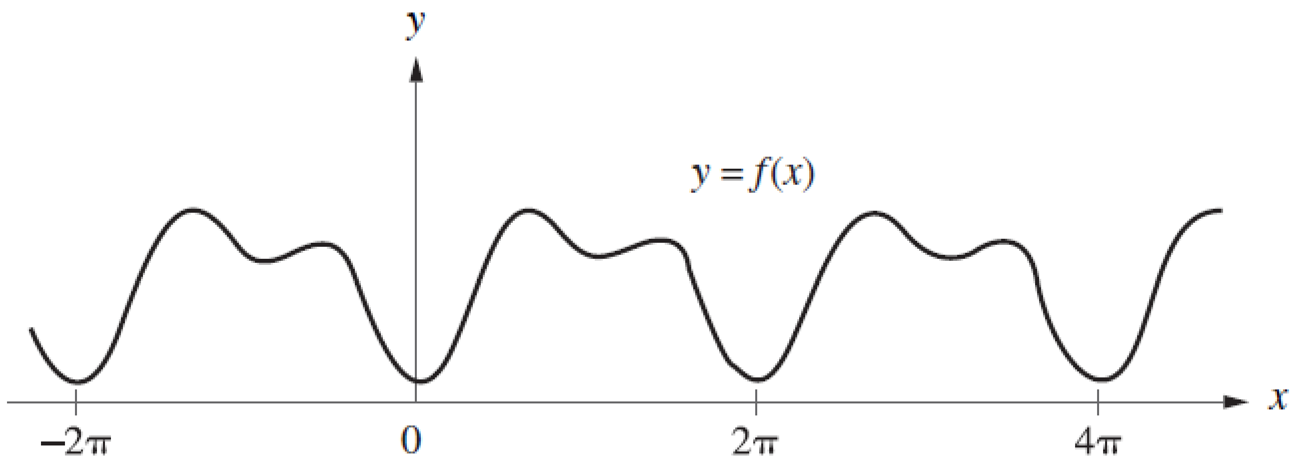


Figure 5.17 A continuous function $f(x)$ with period 2π.

Examples of functions with period 2π are $\sin(jx)$ and $\cos(jx)$, where j is an integer. This raises the following question: Can a periodic function be represented by the sum of terms involving $a_j\cos(jx)$ and $b_j\sin(jx)$?

Fourier Series

Definition 5.2. The function $f(x)$ is said to be ***piecewise continues*** on $[a, b]$ if there exist values $t_0, t_1, \ldots, t_k$ with $a = t_0 < t_1 < \cdots < t_k = b$ such that $f(x)$ is continuous on each open interval $t_{i-1} < x < t_i$ for $i = 1, 2, \ldots, K$, and $f(x)$ has left- and right-hand limits at each of the points t_i.

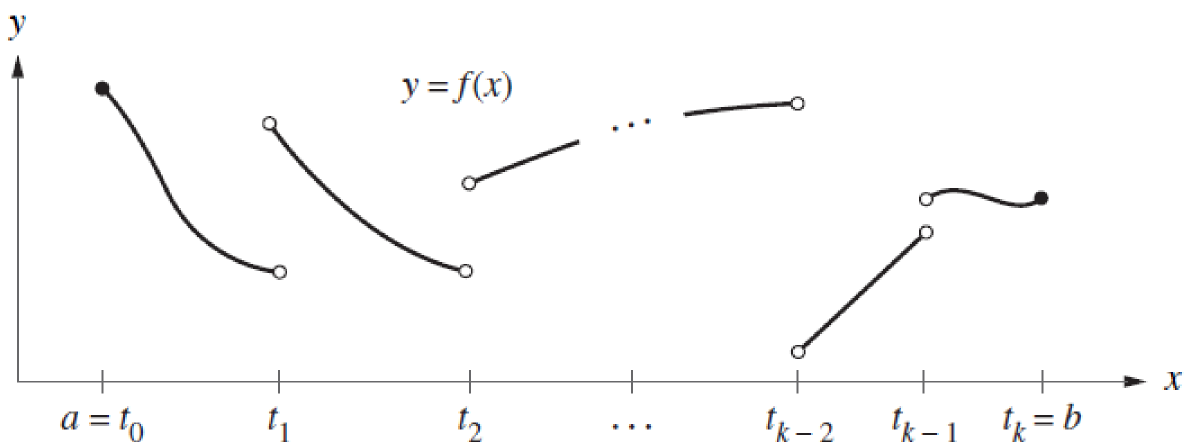


Figure 5.18 A piecewise continuous function over $[a, b]$.

Fourier Series

Definition 5.3. Assume that $f(x)$ is periodic with period 2π and that $f(x)$ is piece-wise continuous on $[-\pi, \pi]$. The ***Fourier series*** $S(x)$ for $f(x)$ is

(4) $$S(x) = \frac{a_0}{2} + \sum_{j=1}^{\infty}(a_j \cos(jx) + b_j \sin(jx)) \ ,$$

where the coefficients a_j and b_j are computed with Euler's formulas:

(5) $$a_j = \frac{1}{\pi}\int_{-\pi}^{\pi} f(x)\cos(jx)\,dx \qquad \text{for } j = 0, 1, \ldots$$

and

(6) $$b_j = \frac{1}{\pi}\int_{-\pi}^{\pi} f(x)\sin(jx)\,dx \qquad \text{for } j = 1, 2, \ldots$$

The factor 1/2 in the constant term $a_0/2$ in the Fourier series (4) has been introduced for convenience so that a_0 could be obtained from the general formula (5) by setting $j = 0$.

Fourier Expansion

Theorem 5.5 (Fourier Expansion). Assume that $S(x)$ is the Fourier series for $f(x)$ over $[-\pi, \pi]$. If $f'(x)$ is piecewise continuous on $[-\pi, \pi]$ and has both a left- and right-hand derivative at each point in this interval, then $S(x)$ is convergent for all $x \in [-\pi, \pi]$. The relation

$$S(x) = f(x)$$

holds at all points $x \in [-\pi, \pi]$, where $f(x)$ is continuous. If $x = a$ is a point of discontinuous of f, then

$$S(a) = \frac{f(a^-) + f(a^+)}{2} .$$

Where $f(a^-)$ and $f(a^+)$ denote the left- and right-hand limits, respectively. With this understanding, we obtain the Fourier expansion:

$$f(x) = \frac{a_0}{2} + \sum_{j=1}^{\infty} \left(a_j \cos(jx) + b_j \sin(jx)\right). \tag{7}$$

Fourier Expansion: Example

Show that the function $f(x) = x/2$ for $-\pi < x < \pi$, extended periodically by the equation $f(x + 2\pi) = f(x)$, has the Fourier series representation

$$f(x) = \sum_{j=1}^{\infty} \frac{(-1)^{j+1}}{j} \sin(jx) = \sin(x) - \frac{\sin(2x)}{2} + \frac{\sin(3x)}{3} - \cdots .$$

Using Euler's formulas and integration by parts, we get

$$a_j = \frac{1}{\pi} \int_{-\pi}^{\pi} \frac{x}{2} \cos(jx)\, dx = \frac{x \sin(jx)}{2\pi j} + \frac{\cos(jx)}{2\pi j^2} \Bigg|_{-\pi}^{\pi} = 0$$

for $j = 1, 2, 3, \ldots,$ and

$$b_j = \frac{1}{\pi} \int_{-\pi}^{\pi} \frac{x}{2} \sin(jx)\, dx = \frac{-x \cos(jx)}{2\pi j} + \frac{\sin(jx)}{2\pi j^2} \Bigg|_{-\pi}^{\pi} = \frac{(-1)^{j+1}}{j}$$

for $j = 1, 2, 3, \ldots$ .The coefficient a_0 is obtained by a separate calculation:

$$a_0 = \frac{1}{\pi} \int_{-\pi}^{\pi} \frac{x}{2}\, dx = \frac{x^2}{4\pi} \Bigg|_{-\pi}^{\pi} = 0 .$$

Fourier Expansion: Example

$$S_2(x) = \sin(x) - \frac{\sin(2x)}{2} \,,$$

$$S_3(x) = \sin(x) - \frac{\sin(2x)}{2} + \frac{\sin(3x)}{3} \,,$$

$$S_4(x) = \sin(x) - \frac{\sin(2x)}{2} + \frac{\sin(3x)}{3} - \frac{\sin(4x)}{4}$$

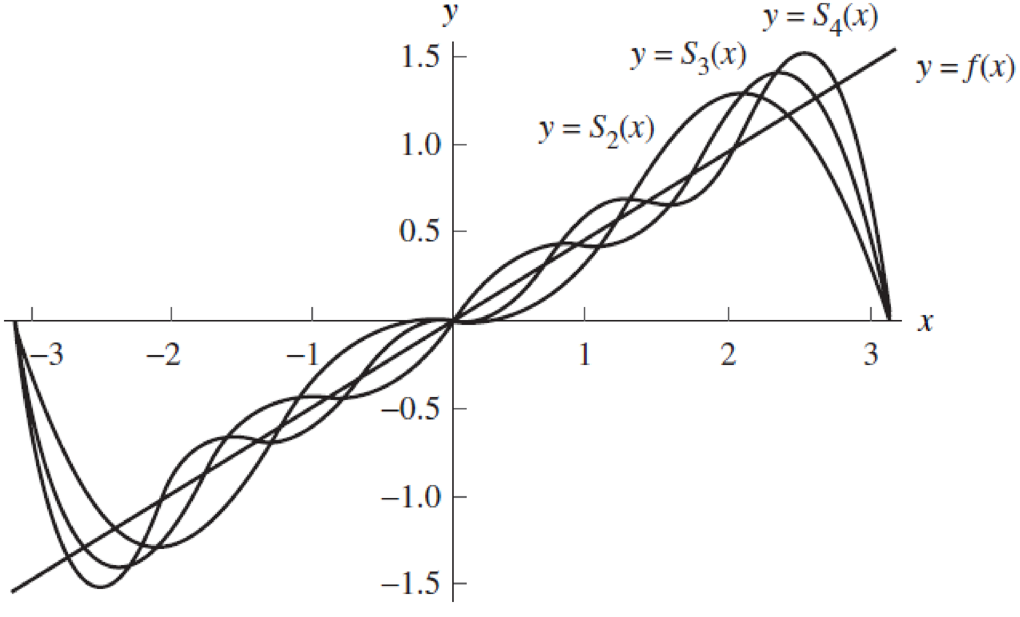


Figure 5.19 The function $f(x) = x/2$ over $[-\pi, \pi]$ and its trigonometric approximations $S_2(x)$, $S_3(x)$, and $S_4(x)$.

Cosine Series and Sine Series

Theorem 5.6 (Cosine Series). Suppose that $f(x)$ is an even function; that is, suppose that $f(-x) = f(x)$ holds for all x. If $f(x)$ has a period 2π and if $f(x)$ and $f'(x)$ are piecewise continuous, then the Fourier series for $f(x)$ involves only cosine terms:

$$f(x) = \frac{a_0}{2} + \sum_{j=1}^{\infty} a_j \cos(jx) \ ,$$

where

$$a_j = \frac{2}{\pi} \int_0^{\pi} f(x) \cos(jx)\, dx \qquad \text{for } j = 0, 1, \ldots .$$

Theorem 5.7 (Sine Series). Suppose that $f(x)$ is an odd function; that is, $f(-x) = -f(x)$ holds for all x. If $f(x)$ has a period 2π and if $f(x)$ and $f'(x)$ are piecewise continuous, then the Fourier series for $f(x)$ involves only the sine terms:

$$f(x) = \sum_{j=1}^{\infty} b_j \sin(jx)\, ,$$

where

$$b_j = \frac{2}{\pi} \int_0^{\pi} f(x) \sin(jx)\, dx \quad \text{for } j = 1, 2, \ldots .$$

Trigonometric Polynomial Approximation

Definition 5.4. A series of the form

(23) $$T_M(x) = \frac{a_0}{2} + \sum_{j=1}^{M} (a_j \cos(jx) + b_j \sin(jx))$$

is called a ***trigonometric polynomial*** of order M .

Theorem 5.8 (Discrete Fourier Series). Suppose that $\{(x_j, y_j)\}_{j=0}^{N}$ are $N+1$ points, where $y_j = f(x_j)$, and the abscissas are equally spaced:

$$x_j = -\pi + \frac{2j\pi}{N} \quad \text{for } j = 0, 1, \ldots, N\,.$$

If $f(x)$ is periodic with period 2π and $2M < N$, then there exist a trigonometric polynomial $T_M(x)$ of the form (23) that minimize the quantity

$$\sum_{k=1}^{N} (f(x_k) - T_M(x_k))^2\,.$$

Trigonometric Polynomial Approximation

The coefficient a_j and b_j of this polynomial are computed with the formulas

(26) $$a_j = \frac{2}{N}\sum_{k=1}^{N} f(x_k)\cos(jx_k) \quad \text{for} \quad j = 0, 1, \ldots, M\ ,$$

And

(27) $$b_j = \frac{2}{N}\sum_{k=1}^{N} f(x_k)\sin(jx_k) \quad \text{for} \quad j = 1, 2, \ldots, M\ ,$$

- Although formulas (26) and (27) are defined with the least-squares procedure, they can also be viewed as numerical approximations to the integrals in Euler’s formulas.
- Euler’s formulas give the coefficients for the Fourier series of a continuous function, whereas formulas (26) and (27) give the trigonometric polynomial coefficients for curve fitting to data points.
- The next example uses data points generated by the function $f(x) = x/2$ at discrete points.

An example

Use the 12 equally spaced points $x_k = -\pi + k\pi/6$, for $k = 1, 2, \ldots, 12$, and find the trigonometric polynomial approximation for $M = 5$ to the 12 data points $\{(x_k, f(x_k))\}_{k=1}^{12}$, where $f(x) = x/2$. Also compare the results when 60 and 360 points are used and with the first five terms of the Fourier series expansion for $f(x)$ that is given in Example 5.13.

- Since the periodic extension is assumed, at a point of discontinuity, the function value $f(\pi)$ must be computed using the formula

$$f(\pi) = \frac{f(\pi^-) + f(\pi^+)}{2} = \frac{\pi/2 - \pi/2}{2} = 0\,.$$

- The function $f(x)$ is an odd function; hence the coefficients for the cosine terms are all zero (i.e., $a_j = 0$ for all j).

$$T_5(x) = 0.9770486\sin(x) - 0.4534498\sin(2x) + 0.26179938\sin(3x)$$
$$-0.1511499\sin(4x) + 0.0701489\sin(5x)\,.$$

An example

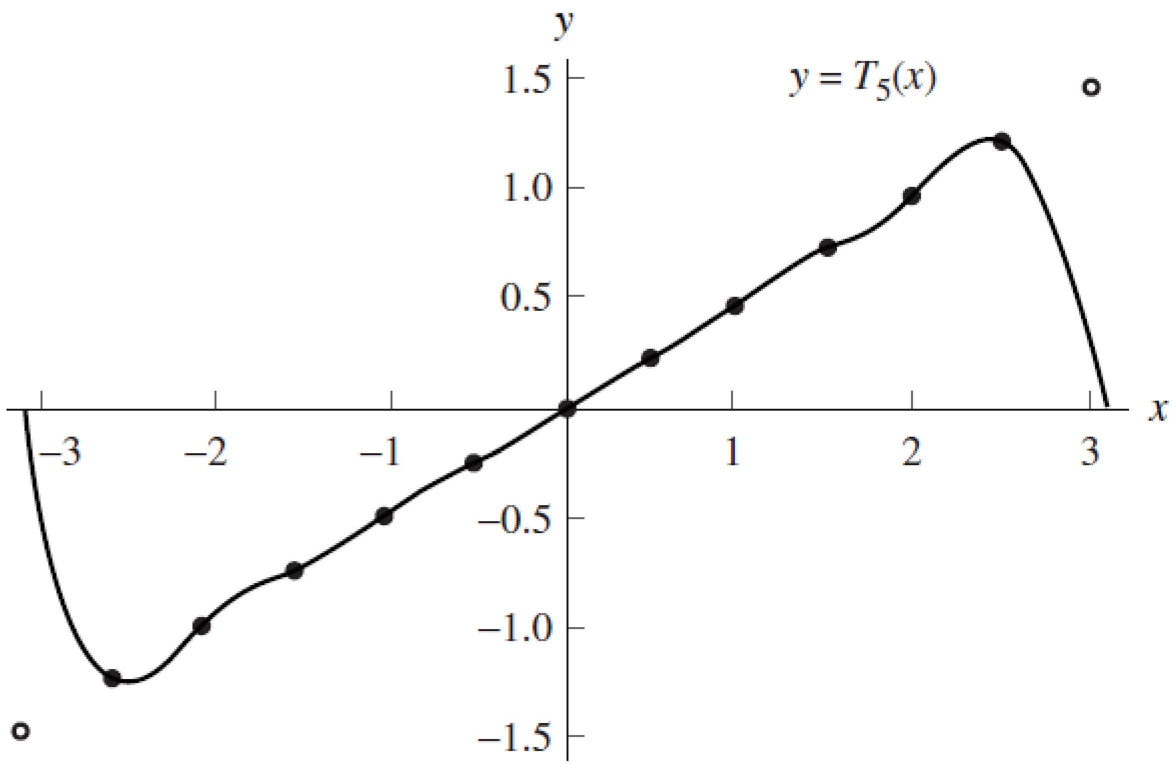


Figure 5.20 The trigonometric polynomial $T_5(x)$ of degree $M = 5$, based on 12 data points that lie on the line $y = x/2$.

An example

- The coefficients of the fifth-degree trigonometric polynomial change slightly when the number of interpolation points increases to 60 and 360.
- As the number of points increases, they get closer to the coefficients of the Fourier series expansion of $f(x)$.

Table 5.9 Comparison of Trigonometric Polynomial Coefficients for Approximations to $f(x) = x/2$ over $[-\pi, \pi]$

	Trigonometric polynomial coefficients			Fourier series coefficients
	12 points	60 points	360 points	
b_1	0.97704862	0.99908598	0.99997462	1.0
b_2	−0.45344984	−0.49817096	−0.49994923	−0.5
b_3	0.26179939	0.33058726	0.33325718	0.33333333
b_4	−0.15114995	−0.24633386	−0.24989845	−0.25
b_5	0.07014893	0.19540972	0.19987306	0.2

MATLAB Code

Program 5.4 (Trigonometric Polynomials). To construct the trigonometric polynomial of order M of the form

$$P(x) = \frac{a_0}{2} + \sum_{j=1}^{M} (a_j \cos(jx) + b_j \sin(jx))$$

based on the N equally spaced values $x_k = -\pi + 2\pi k/N$, for $k = 1, 2, \ldots, N$. The construction is possible provided that $2M + 1 \leq N$.

```
function [A,B]=tpcoeff(X,Y,M)

%Input  - X is a vector of equally spaced abscissas in [-pi,pi]
%       - Y is a vector of ordinates
%       - M is the degree of the trigonometric polynomial
%Output - A is a vector containing the coefficients of cos(jx)
%       - B is a vector containing the coefficients of sin(jx)

N=length(X)-1;
max1=fix((N-1)/2);

if M>max1
   M=max1;
end

A=zeros(1,M+1);
B=zeros(1,M+1);
Yends=(Y(1)+Y(N+1))/2;
Y(1)=Yends;
Y(N+1)=Yends;
A(1)=sum(Y);

for j=1:M
   A(j+1)=cos(j*X)*Y';
   B(j+1)=sin(j*X)*Y';
end

A=2*A/N;
B=2*B/N;
A(1)=A(1)/2;
```

Bézier Curves

Bézier Curves

- Pierre Bézier at Renault and Paul de Casteljau at Citroen independently developed the Bézier curve for CAD/CAM operations, in the 1970s.
- Bézier curves are the basis of the entire Adobe PostScript drawing model that is used in the software products Adobe Illustrator, Macromedia Freehand, and Fontographer.
- Bézier curves continue to be the primary method of representing curves and surfaces in computer graphics (CAD/CAM, computer-aided geometric design).
- The development of the properties of Bézier curves will be facilitated by defining them explicitly in terms of Bernstein polynomials

Bernstein polynomials

Definition 5.5. ***Bernstein polynomial*** of degree N are defined by

$$B_{i,N}(t) = \binom{N}{i} t^i (1-t)^{N-i} \ ,$$

for $i = 0, 1, 2, \ldots, N,$ where $\binom{N}{i} = \dfrac{N!}{i!\,(N-i)!}$.

In general, there are $N+1$ Bernstein polynomial of degree N. For example, the Bernstein polynomial of degrees 1, 2, and 3 are

(1) $$B_{0,1}(t) = 1 - t, B_{1,1}(t) = t;$$

(2) $$B_{0,2}(t) = (1-t)^2, B_{1,2}(t) = 2t(1-t), B_{2,2}(t) = t^2; \quad \text{and}$$

(3) $$B_{0,3}(t) = (1-t)^3, B_{1,3}(t) = 3t(1-t)^2, B_{2,3}(t) = 3t^2(1-t), B_{3,3}(t) = t^3;$$

respectively

Bernstein polynomials

(3) $B_{0,3}(t) = (1-t)^3, B_{1,3}(t) = 3t(1-t)^2, B_{2,3}(t) = 3t^2(1-t), B_{3,3}(t) = t^3;$

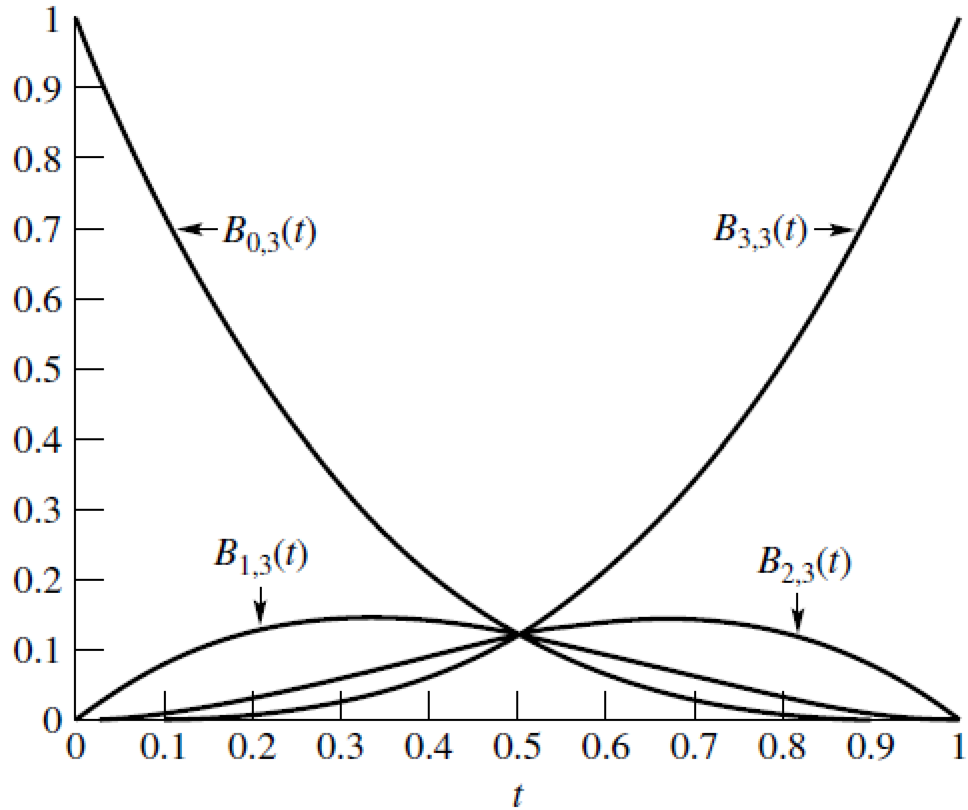


Figure 5.21 Bernstein polynomials of degree three.

Properties of Bernstein Polynomials

Property 1. Recurrence Relation

Bernstein polynomial can be generated in the following way. Set $B_{0,0}(t) = 1$ and $B_{1,N}(t) = 0$ for $i < 0$ or $i > N$, and use the recurrence relation

(4) $$B_{i,N}(t) = (1-t)B_{i,N-1}(t) + tB_{i-1,N-1}(t) \quad \text{for } i = 1, 2, 3, \ldots, N-1.$$

Property 2. Nonnegative on $[0, 1]$

The Bernstein polynomials are nonnegative over the interval $[0, 1]$ (see Figure 5.21).

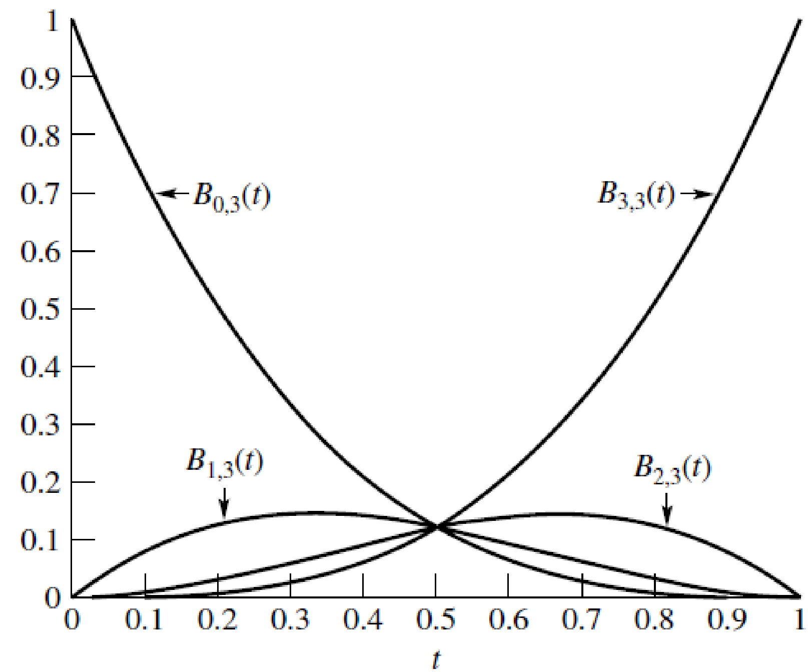


Figure 5.21 Bernstein polynomials of degree three.

Properties of Bernstein Polynomials

Property 3. The Bernstein polynomials form a partition of unity

(5) $$\sum_{i=0}^{N} B_{i,N}(t) = 1$$

Substituting $x = t$ and $y = 1 - t$ into the binomial theorem

$$(x + y)^N = \sum_{i=0}^{N} \binom{N}{i} x^i y^{N-i}$$

yields

$$\sum_{i=0}^{N} \binom{N}{i} x^i y^{N-i} = (t + (1 - t))^N = 1^N = 1\,.$$

Properties of Bernstein Polynomials

Property 4. Derivatives

(6) $$\frac{d}{dt}B_{i,N}(t) = N(B_{i-1,N-1}(t) - B_{i,N-1}(t))$$

Formula (6) is established by taking the derivative of the Bernstein polynomial in Definition 5.5.

$$\begin{aligned}
&\frac{d}{dt}B_{i,N}(t) \\
&= \frac{d}{dt}\binom{N}{i} t^i(1-t)^{N-i} \\
&= \frac{iN!}{i!\,(N-i)!}t^{i-1}(1-t)^{N-i} - \frac{(N-i)N!}{i!\,(N-i)!}t^i(1-t)^{N-i-1} \\
&= \frac{N(N-1)!}{(i-1)!\,(N-i)!}t^{i-1}(1-t)^{N-i} - \frac{N(N-1)!}{i!\,(N-i-1)!}t^i(1-t)^{N-i-1} \\
&= N\left(\frac{(N-1)!}{(i-1)!\,(N-i)!}t^{i-1}(1-t)^{N-i} - \frac{(N-1)!}{i!\,(N-i-1)!}t^i(1-t)^{N-i-1}\right) \\
&= N\left(B_{i-1,N-1}(t) - B_{i,N-1}(t)\right)
\end{aligned}$$

Properties of Bernstein Polynomials

Property 5. Basis

The Bernstein polynomials of order N ($B_{i,N}(t)$ for $i = 0, 1, \ldots, N$) form a basis of the space of all polynomials of degree less than or equal to N.

Property 5 states that any polynomial of degree less than or equal to N can be written uniquely as a linear combination of the Bernstein polynomial of order N.

Bézier Curves

Definition 5.6. Given a set of control points $\{\mathbf{P}_i\}_{i=0}^{N}$, where $\mathbf{P}_i = (x_i, y_i)$, a ***Bézier curve of degree N*** is

(7) $$\mathbf{P}(t) = \sum_{i=0}^{N} \mathbf{P}_i B_{i,N}(t) \ ,$$

where $B_{i,N}(t)$, for $i = 0, 1, \ldots, N$, are the Bernstein polynomial of degree N, and $t \in [0, 1]$.

In formula (7) the control points are ordered pairs representing $x-$ and $y-$coordinates in the plane. Without ambiguity the control points can be treated as vectors and the corresponding Bernstein polynomials as scalars. Thus formula (7) can be represented parametrically as $\mathbf{P}(t) = (x(t), y(t))$, where

(8) $$x(t) = \sum_{i=0}^{N} x_i B_{i,N}(t) \quad \text{and} \quad y(t) = \sum_{i=0}^{N} y_i B_{i,N}(t)$$

and $0 \leq t \leq 1$. The function $\mathbf{P}(t)$ is said to be a vector-valued function, or equivalently, the range of the function is a set of points in the *xy*-plane.

Bézier Curves: An Example

Find the *Bézier curve* which has the control points (2, 2), (1, 1.5), (3.5, 0), and (4, 1).

Substituting the *x*- and *y*-coordinates of the control points and *N* = 3 into formula (8) yields

(9) $$x(t) = 2B_{0,3}(t) + 1B_{1,3}(t) + 3.5B_{2,3}(t) + 4B_{3,3}(t)$$

(10) $$y(t) = 2B_{0,3}(t) + 1.5B_{1,3}(t) + 0B_{2,3}(t) + 1B_{3,3}(t)$$

Substituting the Bernstein polynomials of degree three, found in formula (3), into formulas (9) and (10) yields

(11) $$x(t) = 2(1-t)^3 + 3t(1-t)^2 + 10.5t^2(1-t) + 4t^3$$

(12) $$y(t) = 2(1-t)^3 + 4.5t(1-t)^2 + t^3.$$

Simplify formulas (11) and (12) yields

$$\mathbf{P}(t) = (2 - 3t + 10.5t^2 - 5.5t^3, 2 - 1.5t - 3t^2 + 3.5t^3),$$

where $0 \le t \le 1$.

Bézier Curves: An Example

Find the *Bézier curve* which has the control points (2, 2), (1, 1.5), (3.5, 0), and (4, 1).

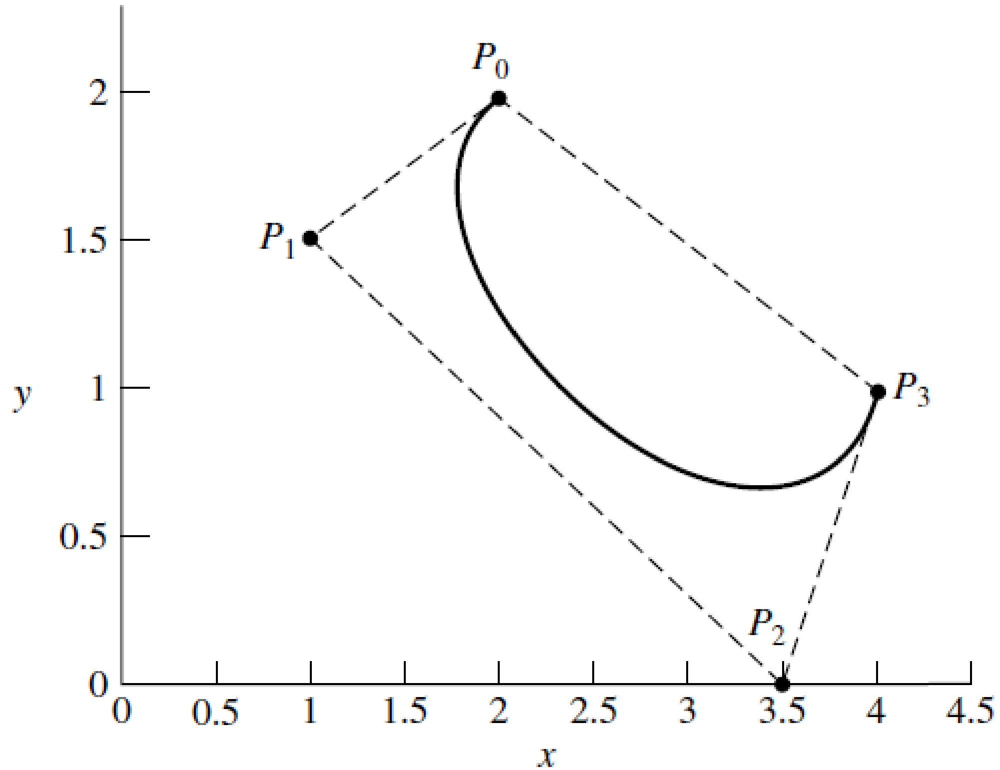


Figure 5.22 Bézier curve of degree three and convex hull of control points.

Properties of Bézier Curves

Property 1. The points $\mathbf{P_0}$ and $\mathbf{P_1}$ are on the curve $\boldsymbol{P(t)}$
Substituting $t = 0$ into Definition 5.5 yields

$$B_{i,N}(0) = \begin{cases} 1 & \text{for } i = 0 \\ 0 & \text{for } i \neq 0. \end{cases}$$

Similarly, $B_{i,N}(1) = 1$ for $i = N$ and is zero for $i = 0, 1, \dots, N-1$. Substituting these results into Definition 5.6 yields

$$\mathbf{P}(0) = \sum_{i=0}^{N} \mathbf{P}_i B_{i,N}(0) = \mathbf{P}_0 \quad \text{and} \quad \mathbf{P}(1) = \sum_{i=0}^{N} \mathbf{P}_i B_{i,N}(1) = \mathbf{P}_N\ .$$

Thus the first and last points in the sequence of control points, $\{\mathbf{P}_i\}_{i=0}^{N}$, are the end-point of the Bézier Curve. *Note.* The remaining control points are not necessarily on the curve.

Properties of Bézier Curves

Property 2. P(*t*) is continuous and has derivatives of all orders on the interval [0, 1]

The derivative of **P**(t), with respect to t, is

$$\begin{aligned}\mathbf{P}'(t) &= \frac{d}{dt}\sum_{i=0}^{N}\mathbf{P}_i B_{i,N}(t) \\ &= \sum_{i=0}^{N}\mathbf{P}_i \frac{d}{dt} B_{i,N}(t) \\ &= \sum_{i=0}^{N}\mathbf{P}_i\, N(B_{i-1,N-1}(t) - B_{i,N-1}(t))\end{aligned}$$

Properties of Bézier Curves

Property 3. $\mathbf{P}'(\mathbf{0}) = \boldsymbol{N}(\mathbf{P_1} - \boldsymbol{P_0})$ **and** $\mathbf{P}'(\mathbf{1}) = \boldsymbol{N}(\mathbf{P}_{\boldsymbol{N}} - \boldsymbol{P}_{\boldsymbol{N-1}})$

According to property 4 of Bernstein polynomial, setting $t = 0$ and substituting $B_{i,N}(0) = 1$ for $i = 0$ and $B_{i,N}(0) = 0$ for $i \geq 1$ (Definition 5.5) into the right-hand side of the expression for $P'(t)$ and simplifying yields.

$$\mathbf{P}'(0) = \sum_{i=0}^{N} \mathbf{P}_i \, N\left(B_{i-1,N-1}(0) - B_{i,N-1}(0)\right) = N(\mathbf{P}_1 - \mathbf{P}_0).$$

Similarly, $\mathbf{P}'(1) = N(\mathbf{P}_N - \mathbf{P}_{N-1})$. In other words, the tangent lines to a Bézier Curve at the endpoints are parallel to the lines through the endpoints and the adjacent control points. The property is illustrated in Figure 5.23.

Properties of Bézier Curves

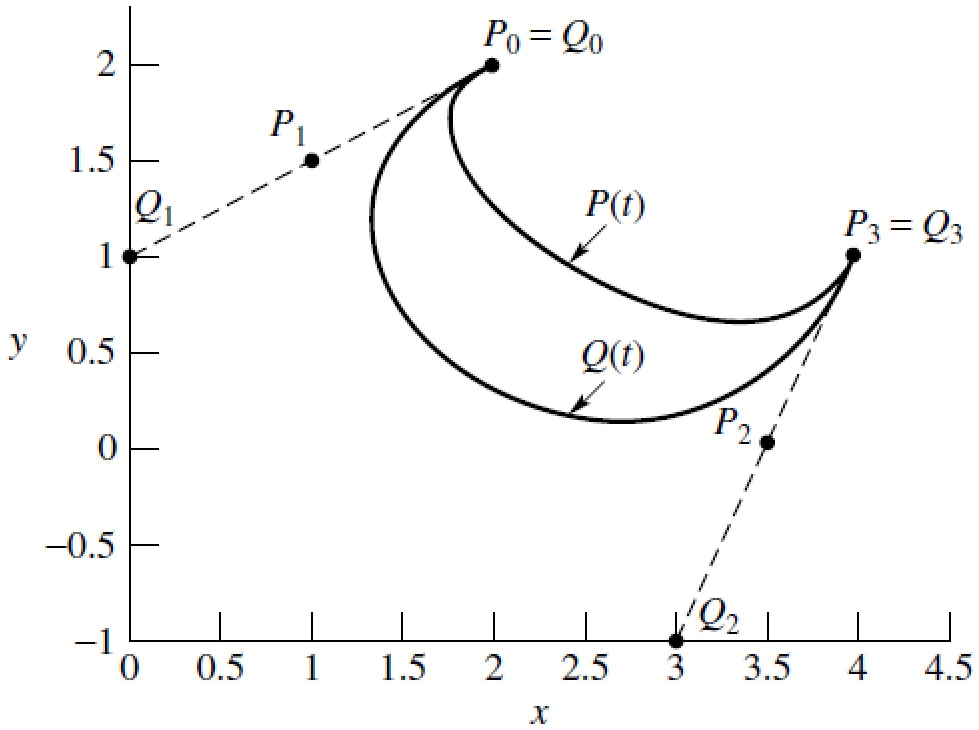


Figure 5.23 $P(t)$, $Q(t)$ and control points.

Properties of Bézier Curves

Property 4. The Bézier curve lies in the convex hull of its set of control points

A subset C of the xy-plane is said to be a convex set, provided that all the points on the line segment joining any two points in C are also elements of the set C.

For example, a line segment or a circle and its interior are convex sets, while a circle without its interior is not a convex set. The convex set concept extends naturally to higher-dimension spaces.

Definition 5.7. The convex hull of a set C is the intersection of all convex sets containing C.

Properties of Bézier Curves

The properties indicate that the graph of a Bézier curve of degree N is a continuous curve, bounded by the convex hull of the set of control points, $\{\mathbf{P}i\}_{i=0}^{N}$, and that the curve begins and ends at points $\mathbf{P}_0$ and $\mathbf{P}_N$, respectively.

The graph is sequentially *pulled* toward each of the remaining control points $\mathbf{P}_1$, $\mathbf{P}_2$, . . . , $\mathbf{P}_{N-1}$.

For example, if the control points $\mathbf{P}_1$ and $\mathbf{P}_{N-1}$ are replaced by the control points $\mathbf{Q}_1$ and $\mathbf{Q}_{N-1}$, which are farther away (but in the same direction) from the respective endpoints, then the resulting Bézier curve will more closely approximate the tangent line near the endpoints.

Properties of Bézier Curves

The effectiveness of Bézier curves lies in the ease with which the shape of the curve can be modified (mouse, keyboard, or other graphical interface) by making small adjustments to the control points.

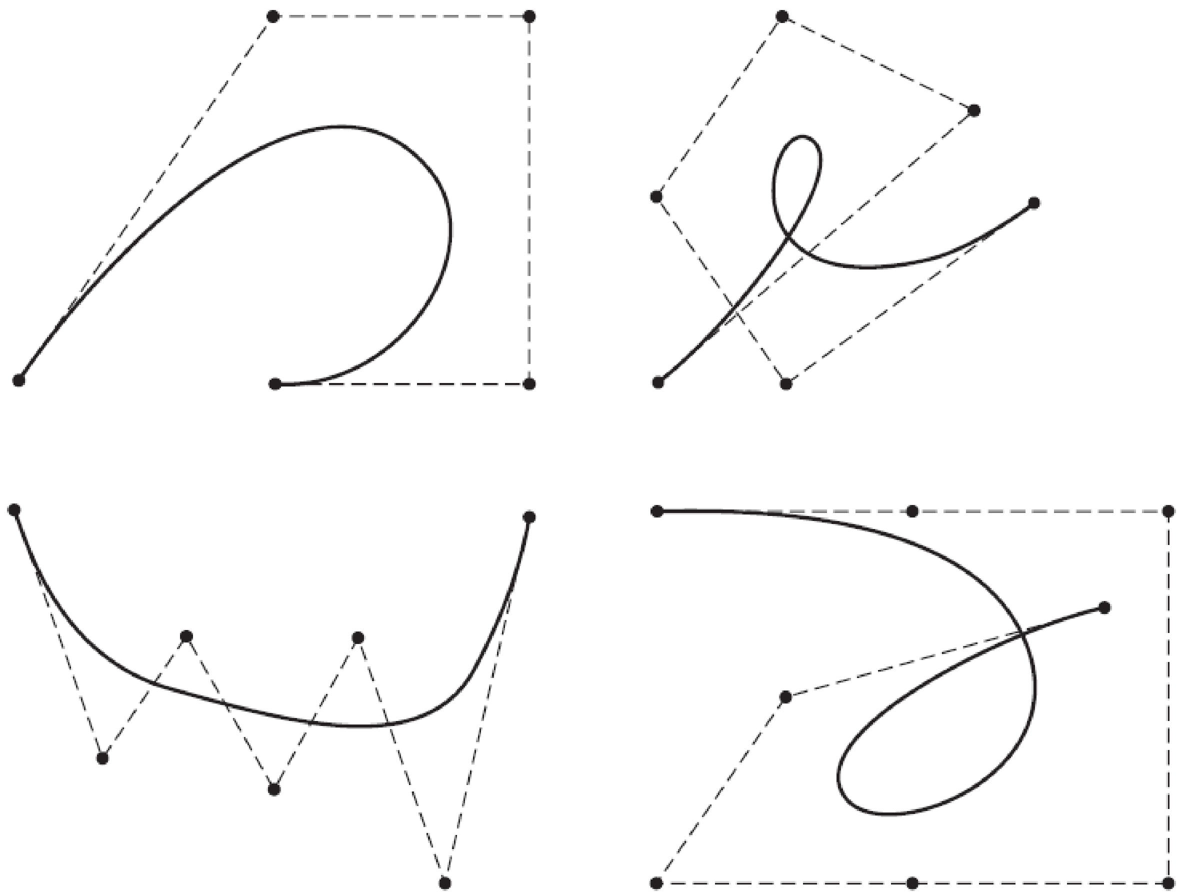


Figure 5.24 Bézier curves and polygonal paths.