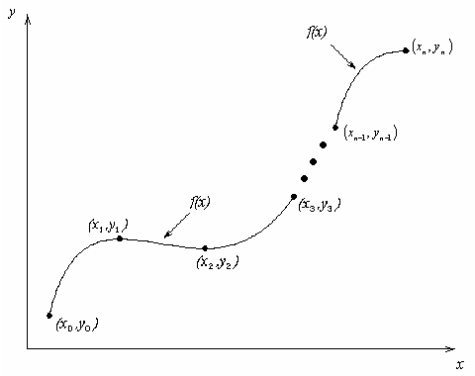
Interpolation

Given a set of discrete values (x_0,y_0), (x_1,y_1), …… (x_n,y_n), find a function that matches these values exactly. The resulting function can then be used to estimate the value of 'y' at a value of 'x' that is not given.



Interpolants

Polynomials are the most common choice of interpolants because they are easy to:

- Evaluate
- Differentiate, and
- Integrate.

Direct Method

Given 'n+1' data points (x_0,y_0), (x_1,y_1),..... (x_n,y_n), pass a polynomial of order 'n' through the data as given below:

$$y = a_0 + a_1 x + \ldots\ldots\ldots\ldots\ldots + a_n x^n .$$

where a_0, a_1,..... a_n are real constants.

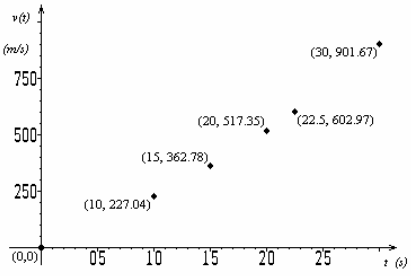
- Set up 'n+1' equations to find 'n+1' constants.
- To find the value 'y' at a given value of 'x', simply substitute the value of 'x' in the above polynomial.

Linear Interpolation

The upward velocity of a rocket is given as a function of time. Find the velocity at t=16 seconds using the direct method for linear interpolation.

t	v(t)
s	m/s
0	0
10	227.04
15	362.78
20	517.35
22.5	602.97
30	901.67

Velocity as a function of time



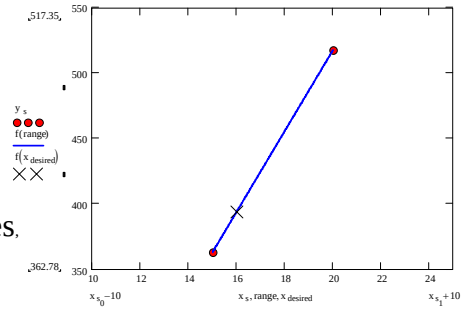
Velocity vs. time data for the rocket example

Linear Interpolation

$$v(t) = a_0 + a_1 t$$

$$v(15) = a_0 + a_1(15) = 362.78$$

$$v(20) = a_0 + a_1(20) = 517.35$$



Solving the above two equations gives.

$a_0 = -100.91$ $a_1 = 30.913$

Hence

$$v(t) = -100.91 + 30.913t, \quad 15 \le t \le 20.$$

$$v(16) = -100.91 + 30.913(16) = 393.7 m/s$$

t	v(t)
s	m/s
0	0
10	227.04
15	362.78
20	517.35
22.5	602.97
30	901.67

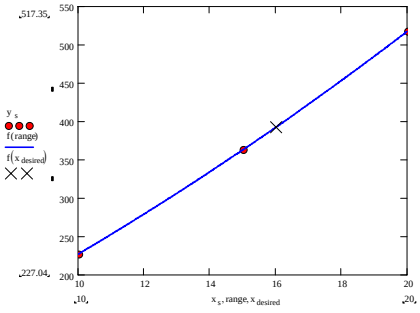
Find the velocity at t=16 seconds using the direct method for quadratic interpolation.

$$v(t) = a_0 + a_1 t + a_2 t^2$$

$$v(10) = a_0 + a_1(10) + a_2(10)^2 = 227.04$$

$$v(15) = a_0 + a_1(15) + a_2(15)^2 = 362.78$$

$$v(20) = a_0 + a_1(20) + a_2(20)^2 = 517.35$$



$a_0 = 12.001$ $a_1 = 17.740$ $a_2 = 0.37637$

$$v(16) = 12.001 + 17.740(16) + 0.37637(16)^2$$

$$= 392.19 m/s$$

Cubic Interpolation

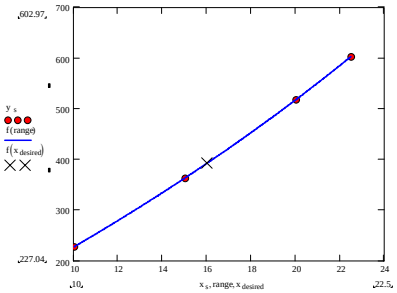
t	v(t)
s	m/s
0	0
10	227.04
15	362.78
20	517.35
22.5	602.97
30	901.67

Find the velocity at t=16 seconds using the direct method for cubic interpolation.

$$v(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3$$

$$v(t) = -4.3810 + 21.289t + 0.13064t^2 + 0.0054606t^3, \quad 10 \le t \le 22.5$$

$$v(16) = -4.3810 + 21.289(16) + 0.13064(16)^2 + 0.0054606(16)^3 = 392.06\, m/s$$



$$v(t) = -4.3810 + 21.289t + 0.13064t^2 + 0.0054606t^3, \quad 10 \le t \le 22.5$$

Find the *distance* covered by the rocket from t=11s to t=16s ?

$$s(16) - s(11) = \int_{11}^{16} v(t)dt$$

$$\approx \int_{11}^{16} \left(-4.3810 + 21.289t + 0.13065t^2 + 0.0054606t^3\right)dt = 1605\, m$$

Find the *acceleration* of the rocket at t=16s

$$a(t) = \frac{d}{dt}v(t) = \frac{d}{dt}\left(-4.3810 + 21.289t + 0.13064t^2 + 0.0054606t^3\right)$$

$$= 21.289 + 0.26130t + 0.016382t^2, \quad 10 \le t \le 22.5$$

$$a(16) = 21.289 + 0.26130(16) + 0.016382(16)^2 = 29.664\, m/s^2$$

Lagrangian Interpolation

Consider a line segment that goes through (x_0, y_0) and (x_1, y_1)

$y = m(x - x_0) + y_0$ where m is the slope $m = (y_1 - y_0)/(x_1 - x_0)$

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

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

$$L_{1,0}(x) = \frac{x - x_1}{x_0 - x_1}$$

$$L_{1,1}(x) = \frac{x - x_0}{x_1 - x_0}.$$

$$P_1(x) = \sum_{k=0}^{1} y_k L_{1,k}(x).$$

$$L_{1,0}(x_0) = 1,\ L_{1,0}(x_1) = 0,\ L_{1,1}(x_0) = 0, \text{ and } L_{1,1}(x_1) = 1$$

Example 4.6. Consider the graph $y = f(x) = \cos(x)$ over $[0.0, 1.2]$.

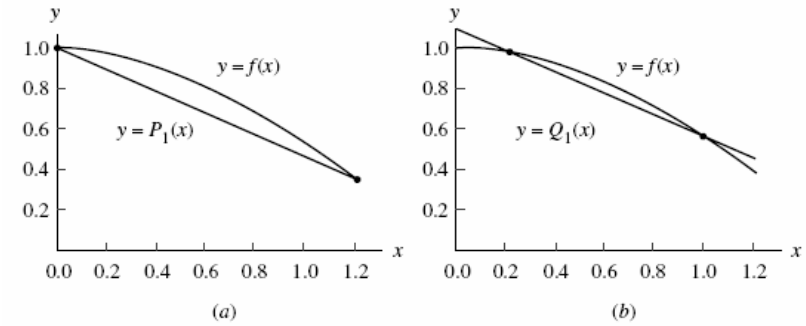


Figure 4.11 (a) The linear approximation $y = P_1(x)$ where the nodes $x_0 = 0.0$ and $x_1 = 1.2$ are the endpoints of the interval $[a, b]$. (b) The linear approximation $y = Q_1(x)$ where the nodes $x_0 = 0.2$ and $x_1 = 1.0$ lie inside the interval $[a, b]$.

$$P_1(x) = 1.000000\frac{x - 1.2}{0.0 - 1.2} + 0.362358\frac{x - 0.0}{1.2 - 0.0}$$

$$Q_1(x) = 0.980067\frac{x - 1.0}{0.2 - 1.0} + 0.540302\frac{x - 0.2}{1.0 - 0.2}$$

General form of Lagrange polynomials

$$P_N(x) = \sum_{k=0}^{N} y_k L_{N,k}(x),$$

$$L_{N,k}(x) = \frac{(x - x_0)\cdots(x - x_{k-1})(x - x_{k+1})\cdots(x - x_N)}{(x_k - x_0)\cdots(x_k - x_{k-1})(x_k - x_{k+1})\cdots(x_k - x_N)}.$$

$$L_{N,k}(x) = \frac{\prod_{\substack{j=0 \\ j\neq k}}^{N}(x - x_j)}{\prod_{\substack{j=0 \\ j\neq k}}^{N}(x_k - x_j)}.$$

$L_{N,k}(x_j) = 1$ when $j = k$ and $L_{N,k}(x_j) = 0$ when $j \neq k$.

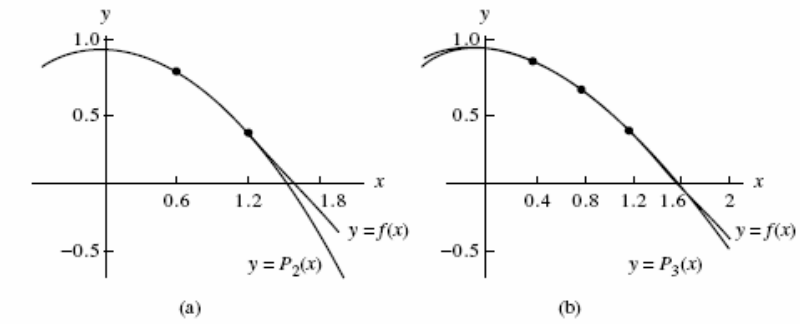


Figure 4.12 (a) The quadratic approximation polynomial $y = P_2(x)$ based on the nodes $x_0 = 0.0$, $x_1 = 0.6$, and $x_2 = 1.2$. (b) The cubic approximation polynomial $y = P_3(x)$ based on the nodes $x_0 = 0.0$, $x_1 = 0.4$, $x_2 = 0.8$, and $x_3 = 1.2$.

$$P_2(x) = y_0\frac{(x-x_1)(x-x_2)}{(x_0-x_1)(x_0-x_2)} + y_1\frac{(x-x_0)(x-x_2)}{(x_1-x_0)(x_1-x_2)} + y_2\frac{(x-x_0)(x-x_1)}{(x_2-x_0)(x_2-x_1)}.$$

$$\begin{aligned} P_2(x) &= 1.0\frac{(x-0.6)(x-1.2)}{(0.0-0.6)(0.0-1.2)} + 0.825336\frac{(x-0.0)(x-1.2)}{(0.6-0.0)(0.6-1.2)} \\ &\quad + 0.362358\frac{(x-0.0)(x-0.6)}{(1.2-0.0)(1.2-0.6)} \\ &= 1.388889(x-0.6)(x-1.2) - 2.292599(x-0.0)(x-1.2) \\ &\quad + 0.503275(x-0.0)(x-0.6). \end{aligned}$$

Error Terms and Error Bounds

Theorem 4.3 (Lagrange Polynomial Approximation). Assume that $f \in C^{N+1}[a, b]$ and that $x_0, x_1, \ldots, x_N \in [a, b]$ are $N + 1$ nodes. If $x \in [a, b]$, then

$$f(x) = P_N(x) + E_N(x), \tag{14}$$

where $P_N(x)$ is a polynomial that can be used to approximate $f(x)$:

$$f(x) \approx P_N(x) = \sum_{k=0}^{N} f(x_k)L_{N,k}(x). \tag{15}$$

The error term $E_N(x)$ has the form

$$E_N(x) = \frac{(x-x_0)(x-x_1)\cdots(x-x_N)f^{(N+1)}(c)}{(N+1)!} \tag{16}$$

for some value $c = c(x)$ that lies in the interval $[a, b]$.

Compare with Taylor approximation

$$P_N(x) = \sum_{k=0}^{N} \frac{f^{(k)}(x_0)}{k!}(x-x_0)^k \qquad E_N(x) = \frac{f^{(N+1)}(c)}{(N+1)!}(x-x_0)^{N+1}$$

Error Terms and Error Bounds $f(x) = P_N(x) + E_N(x),$

For N=1

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

$$E_1(x) = \frac{(x - x_0)(x - x_1)f^{(2)}(c)}{2!},$$

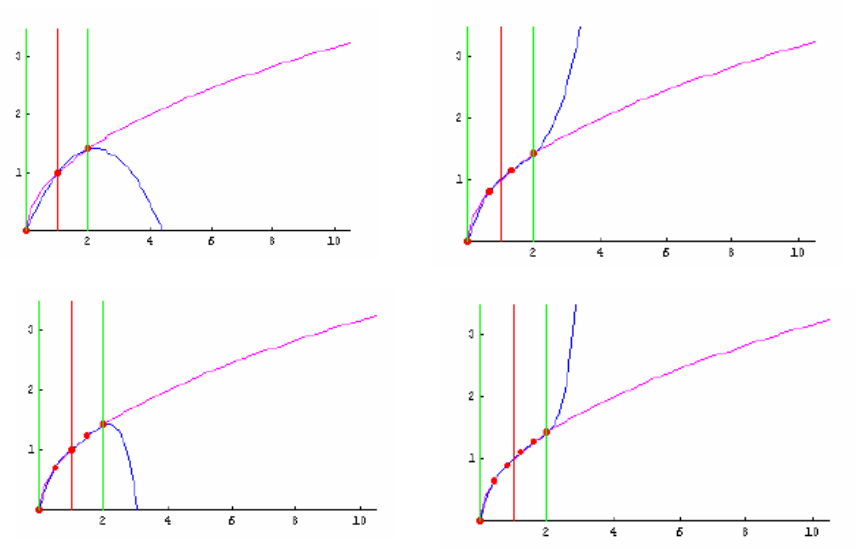
Error bounds for equally spaced nodes $x_k = x_0 + hk$, for $k = 0, 1, \ldots, N$,

$$|E_1(x)| \le \frac{h^2 M_2}{8} \quad \text{valid for } x \in [x_0, x_1], \qquad |E_1(x)| = O(h^2)$$

$$|E_2(x)| \le \frac{h^3 M_3}{9\sqrt{3}} \quad \text{valid for } x \in [x_0, x_2],$$

$$|E_3(x)| \le \frac{h^4 M_4}{24} \quad \text{valid for } x \in [x_0, x_3].$$

Lagrange polynomial approximation for $f[x] = \sqrt{x}$ on interval [0,2]



Interpolation 0≤x≤2
Extrapolation x<0,x>2

Newton's Divided Difference Polynomial Method

It is sometimes useful to find several approximating polynomials $P_1(x)$, $P_2(x)$, . . . ,$P_N(x)$ and then choose the one that suits our needs.

If the Lagrange polynomials are used, there is no constructive relationship between $P_{N-1}(x)$ and $P_N(x)$. Each polynomial has to be constructed individually, and the work required to compute the higher-degree polynomials involves many computations.

Newton polynomials have the **recursive** pattern:

$$
\begin{aligned}
P_1(x) &= a_0 + a_1(x - x_0), \\
P_2(x) &= a_0 + a_1(x - x_0) + a_2(x - x_0)(x - x_1), \\
P_3(x) &= a_0 + a_1(x - x_0) + a_2(x - x_0)(x - x_1) \\
&\quad + a_3(x - x_0)(x - x_1)(x - x_2), \\
&\vdots \\
P_N(x) &= a_0 + a_1(x - x_0) + a_2(x - x_0)(x - x_1) \\
&\quad + a_3(x - x_0)(x - x_1)(x - x_2) \\
&\quad + a_4(x - x_0)(x - x_1)(x - x_2)(x - x_3) + \cdots \\
&\quad + a_N(x - x_0) \cdots (x - x_{N-1}).
\end{aligned}
$$

$$P_N(x) = P_{N-1}(x) + a_N(x - x_0)(x - x_1)(x - x_2) \cdots (x - x_{N-1})$$

Newton's Divided Difference Method

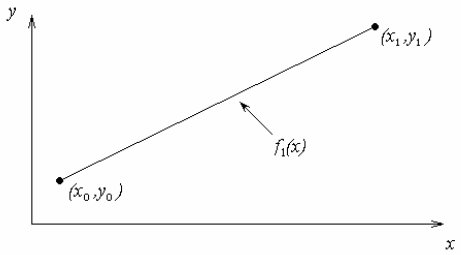
Linear interpolation: Given $(x_0, y_0), (x_1, y_1),$ pass a linear interpolant through the data

$$f_1(x) = b_0 + b_1(x - x_0)$$

where

$$b_0 = f(x_0)$$

$$b_1 = \frac{f(x_1) - f(x_0)}{x_1 - x_0}$$



Quadratic Interpolation

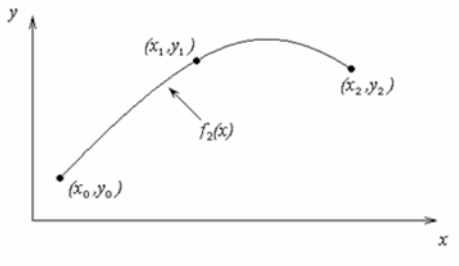
Given (x_0, y_0), (x_1, y_1), and (x_2, y_2), fit a quadratic interpolant through the data.

$$f_2(x) = b_0 + b_1(x - x_0) + b_2(x - x_0)(x - x_1)$$

$$b_0 = f(x_0)$$

$$b_1 = \frac{f(x_1) - f(x_0)}{x_1 - x_0}$$

$$b_2 = \frac{\dfrac{f(x_2) - f(x_1)}{x_2 - x_1} - \dfrac{f(x_1) - f(x_0)}{x_1 - x_0}}{x_2 - x_0}$$



General Form

$$f_2(x) = b_0 + b_1(x - x_0) + b_2(x - x_0)(x - x_1)$$

where

$$b_0 = f[x_0] = f(x_0)$$

$$b_1 = f[x_1, x_0] = \frac{f(x_1) - f(x_0)}{x_1 - x_0}$$

$$b_2 = f[x_2, x_1, x_0] = \frac{f[x_2, x_1] - f[x_1, x_0]}{x_2 - x_0} = \frac{\dfrac{f(x_2) - f(x_1)}{x_2 - x_1} - \dfrac{f(x_1) - f(x_0)}{x_1 - x_0}}{x_2 - x_0}$$

Rewriting

$$f_2(x) = f[x_0] + f[x_1, x_0](x - x_0) + f[x_2, x_1, x_0](x - x_0)(x - x_1)$$

General Form

Given $(n+1)$ data points, $(x_0, y_0), (x_1, y_1), ..., (x_{n-1}, y_{n-1}), (x_n, y_n)$ as

$$f_n(x) = b_0 + b_1(x - x_0) + ... + b_n(x - x_0)(x - x_1)...(x - x_{n-1})$$

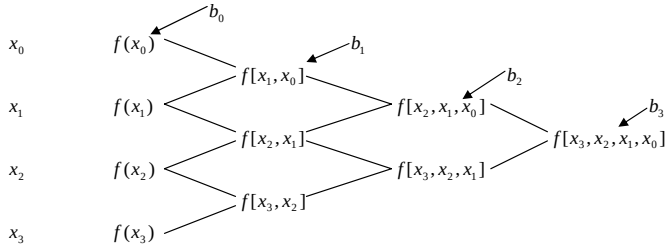
where

$$b_0 = f[x_0]$$
$$b_1 = f[x_1, x_0]$$
$$b_2 = f[x_2, x_1, x_0]$$
$$\vdots$$
$$b_{n-1} = f[x_{n-1}, x_{n-2}, ..., x_0]$$
$$b_n = f[x_n, x_{n-1}, ..., x_0]$$

General form

The third order polynomial, given (x_0, y_0), (x_1, y_1), (x_2, y_2), and (x_3, y_3), is

$$f_3(x) = f[x_0] + f[x_1, x_0](x - x_0) + f[x_2, x_1, x_0](x - x_0)(x - x_1) + f[x_3, x_2, x_1, x_0](x - x_0)(x - x_1)(x - x_2)$$



The upward velocity of a rocket is given as a function of time. Find the velocity at t=16 seconds using the Newton Divided Difference method for cubic interpolation.

$$v(t) = b_0 + b_1(t-t_0) + b_2(t-t_0)(t-t_1) + b_3(t-t_0)(t-t_1)(t-t_2)$$

t	v(t)
s	m/s
0	0
10	227.04
15	362.78
20	517.35
22.5	602.97
30	901.67

Table 1: Velocity as a function of time

t	v(t)
s	m/s
0	0
10	227.04
15	362.78
20	517.35
22.5	602.97
30	901.67

t_k	$f[t_k]$	$f[\,,\,]$	$f[\,,\,,\,]$	$f[\,,\,,\,,\,]$
		b_0		
$t_0 = 10$	227.04		b_1	
		27.148		b_2
$t_1 = 15,$	362.78		0.37660	
		30.914		b_3
$t_2 = 20,$	517.35		0.44453	$5.4347x10^{-3}$
		34.248		
$t_3 = 22.5,$	602.97			

$b_0 = 227.04$; $b_1 = 27.148$; $b_2 = 0.37660$; $b_3 = 5.4347*10^{-3}$

$b_0 = 227.04$; $b_1 = 27.148$; $b_2 = 0.37660$; $b_3 = 5.4347*10^{-3}$

$$v(t) = b_0 + b_1(t - t_0) + b_2(t - t_0)(t - t_1) + b_3(t - t_0)(t - t_1)(t - t_2)$$
$$= 227.04 + 27.148(t - 10) + 0.37660(t - 10)(t - 15)$$
$$+ 5.4347 * 10^{-3}(t - 10)(t - 15)(t - 20)$$

Spline Interpolation Method

Pittfalls of polynomial approximation

$$f(x) = \frac{1}{1+25x^2}$$

Table : Six equidistantly spaced points in [-1, 1]

x	$y = \frac{1}{1+25x^2}$
-1.0	0.038461
-0.6	0.1
-0.2	0.5
0.2	0.5
0.6	0.1
1.0	0.038461

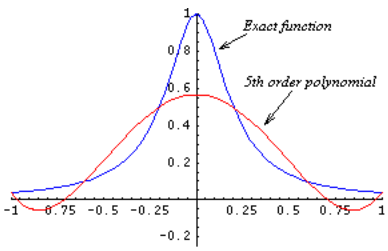


Figure : 5th order polynomial vs. exact function

Pittfalls of polynomial approximation

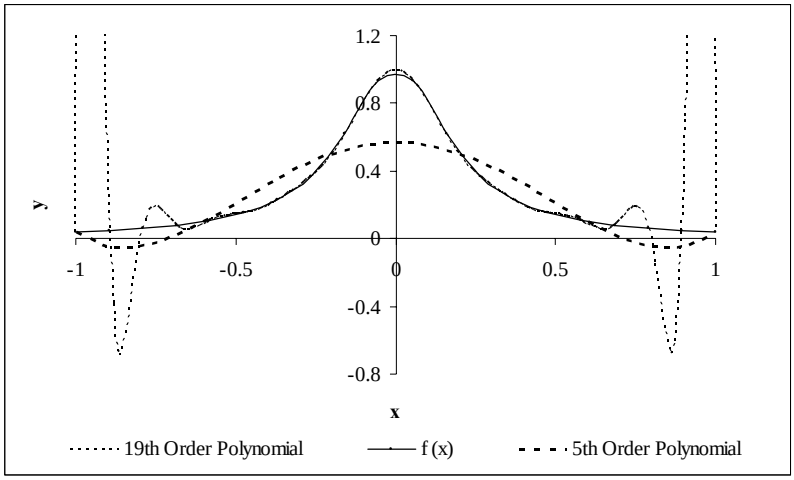


Figure : Higher order polynomial interpolation is a bad idea

Linear spline interpolation

Applying linear Lagrange interpolation polynomial ***piecewise,*** that is between two successive points

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

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

We get *N* linear polynomials for *N+1* points

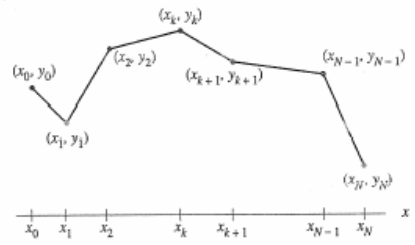
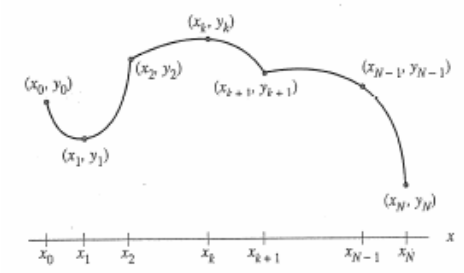


Figure 5.11 Piecewise linear interpolation (a linear spline).

Note: the interpolation function is continuos over the domain, while the slope is discontinuous.

Quadratic spline interpolation

We can also use quadratic Lagrange interpolation polynomial ***piecewise*** using three consecutive points



Note: the interpolation function and its slope are continuos over the domain, while the curvature is discontinuous.

Most popular splines are the Cubic Splines

Given *N+1* points construct *N* cubic polynomials S_k for each interval $[x_k, x_{k+1}]$ so that the resulting cubic spline *S(x),* its first derivative *S'(x),* and its second derivative *S''(x)* are continuous over $[x_0, x_N]$.

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 $k = 0, 1, \ldots, N-2$.

Property I states that $S(x)$ consists of piecewise cubics. Property II states that the piecewise cubics interpolate the given set of data points. Properties III and IV require that the piecewise cubics represent a smooth continuous function. Property V states that the second derivative of the resulting function is also continuous.

Construction of Cubic Splines

IV. $S_k''(x_{k+1}) = S_{k+1}''(x_{k+1})$ for $k = 0, 1, \ldots, n-2$. The second derivative is continuous.

Start with $s_k''(x)$ which satisfies IV.

$$s_k''(x) = s''(x_k)\frac{x - x_{k+1}}{x_k - x_{k+1}} + s''(x_{k+1})\frac{x - x_k}{x_{k+1} - x_k}$$

Let $s''(x_k) = m_k$, $s''(x_{k+1}) = m_{k+1}$, $x_{k+1} - x_k = h_k$ and integrate s_k'' twice

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

where p_k, q_k are integration constants.

Construction of Cubic Splines (cont.)

I. $S(x_k) = y_k$ for $k = 0, 1, \ldots, n$. The spline passes through each data point.

II. $S_k(x_{k+1}) = S_{k+1}(x_{k+1})$ for $k = 0, 1, \ldots, n-2$. The spline is continuous over [a,b].

Impose I & II

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

Consider $s_k'(x) = -\frac{m_k}{2h_k}(x_{k+1} - x)^2 + \frac{m_{k+1}}{2h_k}(x - x_k)^2 - p_k + q_k$

Construction of Cubic Splines (cont.)

III. $S_k'(x_{k+1}) = S_{k+1}'(x_{k+1})$ for $k = 0, 1, \ldots, n-2$. The first derivative is continuous.

Impose III and substitute p_k, q_k

$$h_{k-1} m_{k-1} + 2(h_{k-1} + h_k) m_k + h_k m_{k+1} = 6\left(\frac{y_{k+1} - y_k}{h_k} - \frac{y_k - y_{k-1}}{h_{k-1}}\right)$$

$k = 1, \ldots, n-1$

Solve the above for m_k, then substitute in

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

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

Construction of Cubic Splines (cont.)

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 $k = 0, 1, \ldots, N-2$.

Note: $4n-2$ equations are available to solve $4n$ unknowns

2 more equations are needed

- $s''(a) = 0$, $s''(b) = 0$ natural cubic spline
- $s'(a) = d_0$, $s'(b) = d_n$ clamped cubic spline

Example 5.8. Find the natural cubic spline that passes through (0, 0.0), (1, 0.5), (2, 2.0), and (3, 1.5) with the free boundary conditions $S''(x) = 0$ and $S''(3) = 0$.

$$h_{k-1} m_{k-1} + 2(h_{k-1} + h_k) m_k + h_k m_{k+1} = 6\left(\frac{y_{k+1} - y_k}{h_k} - \frac{y_k - y_{k-1}}{h_{k-1}}\right)$$

$$2(1+1)m_1 + m_2 = 6.0,$$
$$m_1 + 2(1+1)m_2 = -12.0.$$

$$m_1 = 2.4 \text{ and } m_2 = -3.6.$$

$$m_0 = S''(0) = 0 \quad m_3 = S''(3) = 0$$

$$S_0(x) = 0.4x^3 + 0.1x \qquad \text{for } 0 \le x \le 1,$$
$$S_1(x) = -(x-1)^3 + 1.2(x-1)^2 + 1.3(x-1) + 0.5 \qquad \text{for } 1 \le x \le 2,$$
$$S_2(x) = 0.6(x-2)^3 - 1.8(x-2)^2 + 0.7(x-2) + 2.0 \qquad \text{for } 2 \le x \le 3.$$

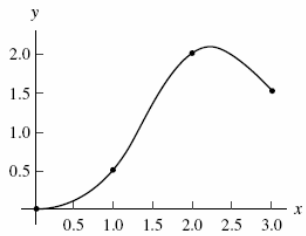


Figure 5.13 The natural cubic spline with $S''(0) = 0$ and $S''(3) = 0$.

Linear Regression
(Linear Curve Fitting)

What is Regression?

Given n data points $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$ best fit $y = f(x)$ to the data. The best fit is generally based on minimizing the sum of the square of the residuals, S_r .

Residual at a point is

$$\varepsilon_i = y_i - f(x_i)$$

Sum of the square of the residuals

$$S_r = \sum_{i=1}^{n} (y_i - f(x_i))^2$$

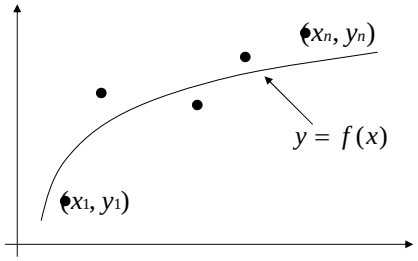


Figure. Basic model for regression

Linear Regression-Criterion#1

Given n data points $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$ best fit $y = a_0 + a_1 x$ to the data.

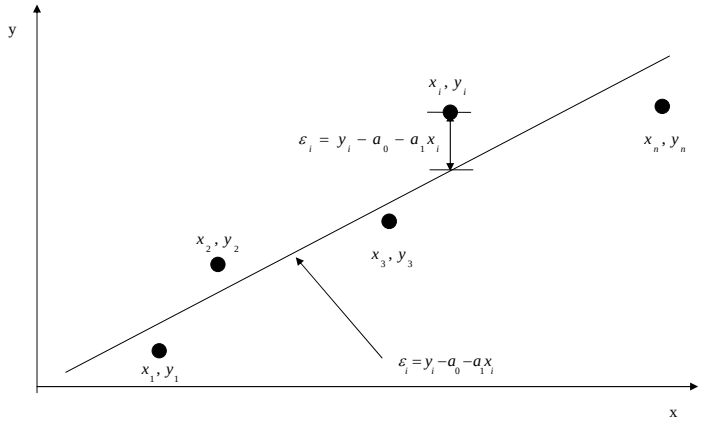


Figure. Linear regression of y vs. x data showing residuals at a typical point, x_i.

Does minimizing $\sum_{i=1}^{n} \varepsilon_i$ work as a criterion, where $\varepsilon_i = y_i - (a_0 + a_1 x_i)$

Example for Criterion#1

Example: Given the data points (2,4), (3,6), (2,6) and (3,8), best fit the data to a straight line using Criterion#1

Table. Data Points

x	y
2.0	4.0
3.0	6.0
2.0	6.0
3.0	8.0

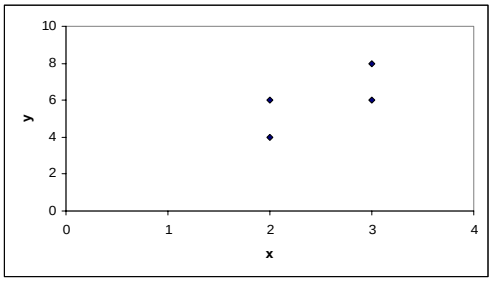


Figure. Data points for y vs. x data.

Linear Regression-Criterion#1

Using *y=4x-4* as the regression curve

Table. Residuals at each point for regression model y = 4x – 4.

x	y	$y_{predicted}$	$\varepsilon = y - y_{predicted}$
2.0	4.0	4.0	0.0
3.0	6.0	8.0	-2.0
2.0	6.0	4.0	2.0
3.0	8.0	8.0	0.0
			$\sum_{i=1}^{4} \varepsilon_i = 0$

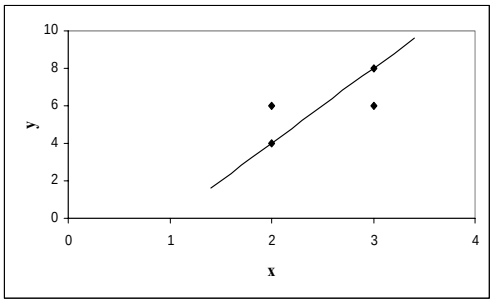


Figure. Regression curve for y=4x-4, y vs. x data

Linear Regression-Criterion#1

Using $y=6$ as a regression curve

Table. Residuals at each point for y=6

x	y	$y_{predicted}$	$\varepsilon = y - y_{predicted}$
2.0	4.0	6.0	-2.0
3.0	6.0	6.0	0.0
2.0	6.0	6.0	0.0
3.0	8.0	6.0	2.0
			$\sum_{i=1}^{4} \varepsilon_i = 0$

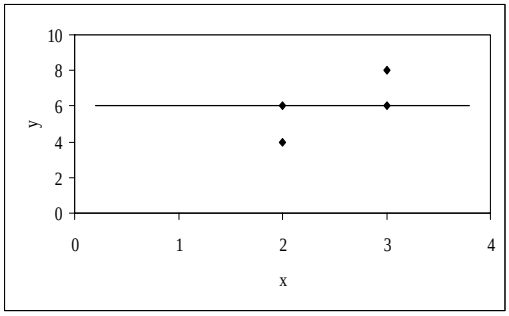


Figure. Regression curve for y=6, y vs. x data

Linear Regression – Criterion #1

$\sum_{i=1}^{4} \varepsilon_i = 0$ for both regression models of y=4x-4 and y=6.

The sum of the residuals is as small as possible, that is zero, but the regression model is not unique.

Hence the above criterion of minimizing the sum of the residuals is a bad criterion.

Linear Regression-Criterion#2

Will minimizing $\sum_{i=1}^{n} \left| \varepsilon_i \right|$ work any better?

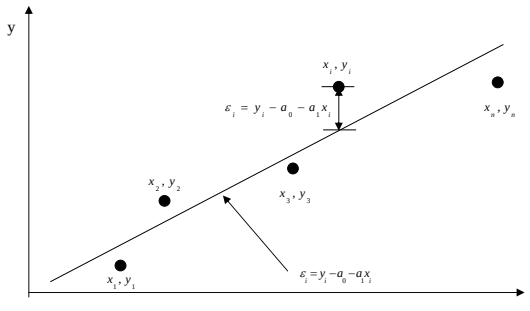


Figure. Linear regression of y vs. x data showing residuals at a typical point, x_i.

Linear Regression-Criterion 2

Using *y=4x-4* as the regression curve

Table. The absolute residuals employing the y=4x-4 regression model

x	y	$y_{predicted}$	$\lvert\varepsilon\rvert = \lvert y - y_{predicted}\rvert$
2.0	4.0	4.0	0.0
3.0	6.0	8.0	2.0
2.0	6.0	4.0	2.0
3.0	8.0	8.0	0.0
			$\sum_{i=1}^{4} \left\| \varepsilon_i \right\| = 4$

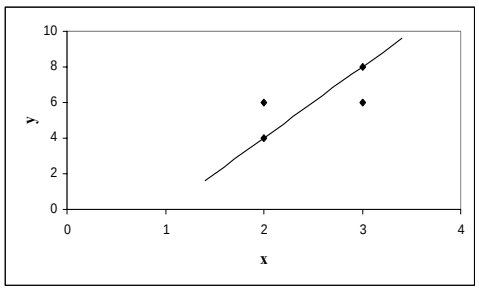


Figure. Regression curve for y=4x-4, y vs. x data

Linear Regression-Criterion#2

Using $y=6$ as a regression curve

Table. Absolute residuals employing the y=6 model

x	y	$y_{predicted}$	$\|\varepsilon\| = \|y - y_{predicted}\|$
2.0	4.0	6.0	2.0
3.0	6.0	6.0	0.0
2.0	6.0	6.0	0.0
3.0	8.0	6.0	2.0
			$\sum_{i=1}^{4} \left\| \varepsilon_i \right\| = 4$

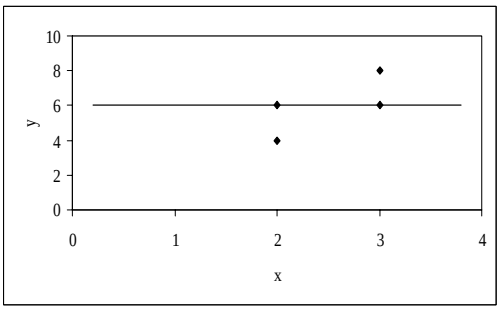


Figure. Regression curve for y=6, y vs. x data

Linear Regression-Criterion#2

$\sum_{i=1}^{4} \left\| \varepsilon_i \right\| = 4$ for both regression models of y=4x-4 and y=6.

The sum of the errors has been made as small as possible, that is 4, but the regression model is not unique.

Hence the above criterion of minimizing the sum of the absolute value of the residuals is also a bad criterion.

Least Squares Criterion

The least squares criterion minimizes the sum of the square of the residuals in the model, and also produces a unique line.

$$S_r = \sum_{i=1}^{n} \varepsilon_i^2 = \sum_{i=1}^{n} \left(y_i - a_0 - a_1 x_i\right)^2$$

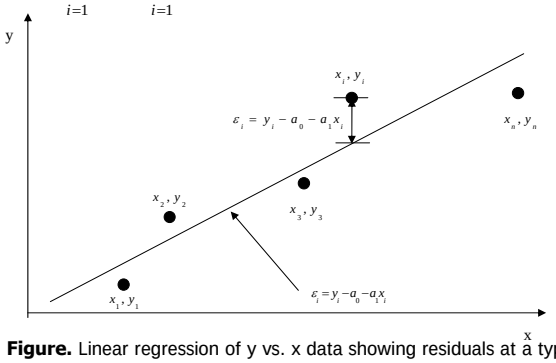


Figure. Linear regression of y vs. x data showing residuals at a typical point, x_i.

Finding Constants of Linear Model

Minimize the sum of the square of the residuals: $S_r = \sum_{i=1}^{n} \varepsilon_i^2 = \sum_{i=1}^{n} \left(y_i - a_0 - a_1 x_i\right)^2$

To find a_0 and a_1 we minimize S_r with respect to a_0 and a_1.

$$\frac{\partial S_r}{\partial a_0} = -2\sum_{i=1}^{n} \left(y_i - a_0 - a_1 x_i\right)(-1) = 0$$

$$\frac{\partial S_r}{\partial a_1} = -2\sum_{i=1}^{n} \left(y_i - a_0 - a_1 x_i\right)(-x_i) = 0$$

giving

$$\sum_{i=1}^{n} a_0 + \sum_{i=1}^{n} a_1 x_i = \sum_{i=1}^{n} y_i$$

$$\sum_{i=1}^{n} a_0 x_i + \sum_{i=1}^{n} a_1 x_i^2 = \sum_{i=1}^{n} y_i x_i$$

Solve for a_0 and a_1

Example 1

The torque, T needed to turn the torsion spring of a mousetrap through an angle, is given below. Find the constants for the model given by $T = k_1 + k_2\theta$

Table: Torque vs Angle for a torsional spring

Angle, θ	**Torque, T**
Radians	*N-m*
0.698132	0.188224
0.959931	0.209138
1.134464	0.230052
1.570796	0.250965
1.919862	0.313707

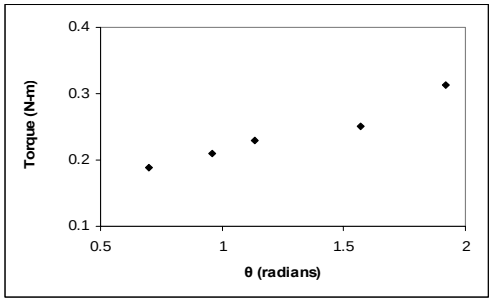


Figure. Data points for Angle vs. Torque data

Example 1 cont.

The following table shows the summations needed for the calculations of the constants in the regression model.

Table. Tabulation of data for calculation of important summations

θ	T	θ^2	$T\theta$
Radians	*N-m*	*Radians²*	*N-m-Radians*
0.698132	0.188224	0.487388	0.131405
0.959931	0.209138	0.921468	0.200758
1.134464	0.230052	1.2870	0.260986
1.570796	0.250965	2.4674	0.394215
1.919862	0.313707	3.6859	0.602274
6.2831	1.1921	8.8491	1.5896

Using equations described for a_0 and a_1 with $n = 5$

$$\sum_{i=1}^{n} a_0 + \sum_{i=1}^{n} a_1 x_i = \sum_{i=1}^{n} y_i$$

$$\sum_{i=1}^{n} a_0 x_i + \sum_{i=1}^{n} a_1 x_i^{\,2} = \sum_{i=1}^{n} y_i x_i$$

$a_0 = 1.1767 \times 10^{-1}$ *N-m*

$a_1 = 9.6091 \times 10^{-2}$ *N-m/rad*

Example 1 Results

Using linear regression, a trend line is found from the data

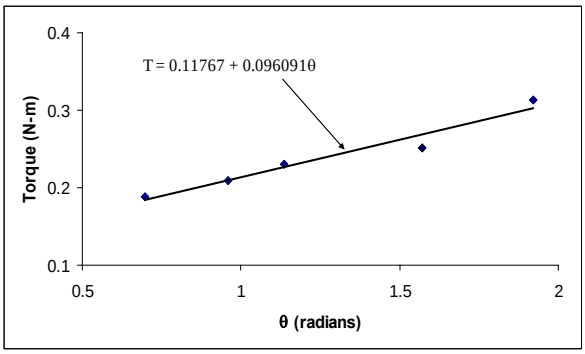


Figure. Linear regression of Torque versus Angle data

Can you find the energy in the spring if it is twisted from 0 to 180 degrees?

Polynomial Fitting

$$y = f(x) = Ax^2 + Bx + C$$

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

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

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

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

$$\left(\sum_{k=1}^{N} x_k^4\right) A + \left(\sum_{k=1}^{N} x_k^3\right) B + \left(\sum_{k=1}^{N} x_k^2\right) C = \sum_{k=1}^{N} y_k x_k^2,$$

$$\left(\sum_{k=1}^{N} x_k^3\right) A + \left(\sum_{k=1}^{N} x_k^2\right) B + \left(\sum_{k=1}^{N} x_k\right) C = \sum_{k=1}^{N} y_k x_k,$$

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

Generalized Polynomial Model

Given $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$ best fit $y = a_0 + a_1 x + \ldots + a_m x^m$ $(m \le n-2)$ to a given data set.

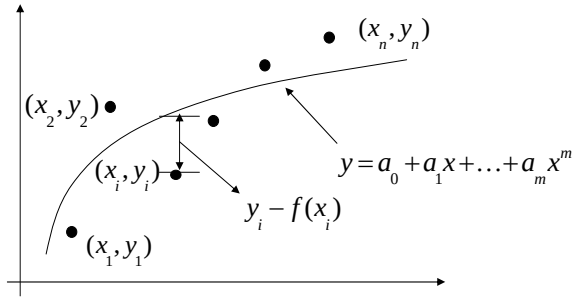


Figure. Polynomial model

Generalized Polynomial Model cont.

The residual at each data point is given by

$$E_i = y_i - a_0 - a_1 x_i - \ldots - a_m x_i^m$$

The sum of the square of the residuals then is

$$S_r = \sum_{i=1}^{n} E_i^2$$

$$= \sum_{i=1}^{n} \left(y_i - a_0 - a_1 x_i - \ldots - a_m x_i^m \right)^2$$

Generalized Polynomial Model cont.

To find the constants of the polynomial model, we set the derivatives with respect to a_i where $i = 1, \ldots m,$ equal to zero.

$$\frac{\partial S_r}{\partial a_0} = \sum_{i=1}^{n} 2.\left(y_i - a_0 - a_1 x_i - \ldots - a_m x_i^m\right)(-1) = 0$$

$$\frac{\partial S_r}{\partial a_1} = \sum_{i=1}^{n} 2.\left(y_i - a_0 - a_1 x_i - \ldots - a_m x_i^m\right)(-x_i) = 0$$

$$\vdots \qquad \vdots \qquad \vdots \qquad \vdots$$

$$\frac{\partial S_r}{\partial a_m} = \sum_{i=1}^{n} 2.\left(y_i - a_0 - a_1 x_i - \ldots - a_m x_i^m\right)(-x_i^m) = 0$$

Generalized Polynomial Model cont.

These equations in matrix form are given by

$$\begin{bmatrix} n & \left(\sum_{i=1}^{n} x_i\right) & \ldots & \left(\sum_{i=1}^{n} x_i^m\right) \\ \left(\sum_{i=1}^{n} x_i\right) & \left(\sum_{i=1}^{n} x_i^2\right) & \ldots & \left(\sum_{i=1}^{n} x_i^{m+1}\right) \\ \ldots & \ldots & \ldots & \ldots \\ \left(\sum_{i=1}^{n} x_i^m\right) & \left(\sum_{i=1}^{n} x_i^{m+1}\right) & \ldots & \left(\sum_{i=1}^{n} x_i^{2m}\right) \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \ldots \\ a_m \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{n} y_i \\ \sum_{i=1}^{n} x_i y_i \\ \ldots \\ \sum_{i=1}^{n} x_i^m y_i \end{bmatrix}$$

The above equations are then solved for $a_0, a_1, \ldots, a_m$

Nonlinear Regression

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:

(11) $$y = Ce^{Ax}.$$

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

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

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

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

and

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

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

Nonlinear equations in C, A

Newton's method can be used for the solution

(*fminsearch* command in MATLAB)

Linearization of Data

For some models which require simultaneous solution of nonlinear equations, data linearization may be mathematically convenient.

For example, the data for an exponential model can be linearized.

(See Table 5.6 in the textbook for other models)

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

(1) $$y = Ce^{Ax}.$$

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

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

Then introduce the change of variables:

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

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

(4) $$Y = AX + B.$$

$$Y = AX + B.$$

(5)
$$\begin{aligned} \left(\sum_{k=1}^{N} X_k^2\right) A + \left(\sum_{k=1}^{N} X_k\right) B &= \sum_{k=1}^{N} X_k Y_k, \\ \left(\sum_{k=1}^{N} X_k\right) A + NB &= \sum_{k=1}^{N} Y_k. \end{aligned}$$

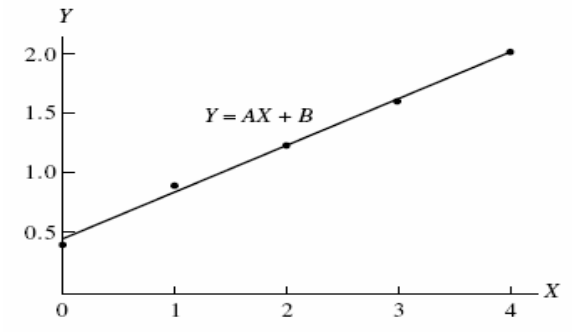
After A and B have been found, the parameter C in equation (1) is computed:

(6)
$$C = e^B.$$

Example 5.4. 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).

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 $= \sum X_k Y_k$



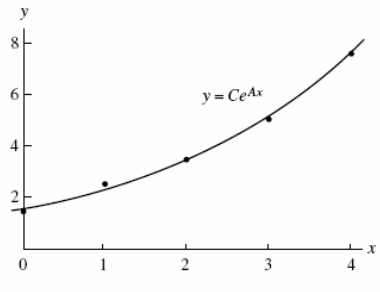


Figure 5.5 The exponential fit $y = 1.579910e^{0.3912023x}$ obtained by using the data linearization method.