Numerical Methods for Engineers

Steven C. Chapra
Raymond P. Canale

Fifth Edition

Interpolation

Chapter 18

Copyright © 2006 The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

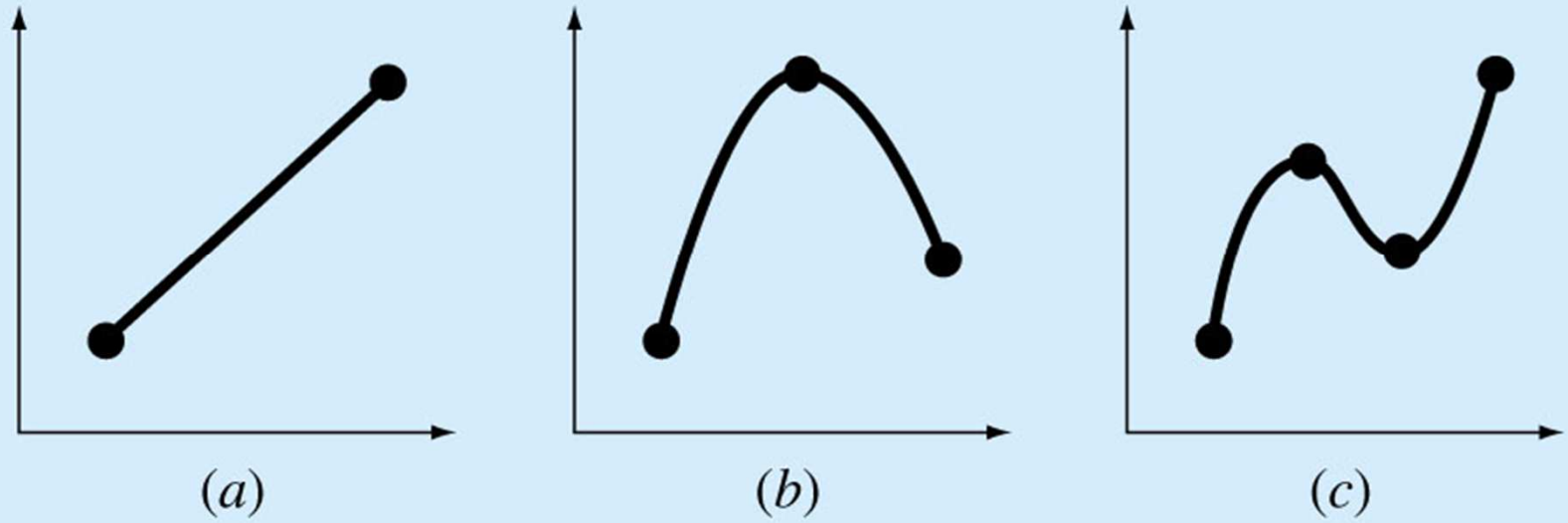
Interpolation **:** Estimation of a function value at an intermediate point that lie between precise data points.

- There is one and only one n^{th}-order polynomial that **perfectly** fits $n+1$ data points:

$$f(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_n x^n$$

- There are several methods to find the fitting polynomial:

the **Newton** polynomial and the **Lagrange** polynomial



Copyright © 2006 The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

Newton's Divided-Difference Interpolating Polynomials

Linear Interpolation

Connecting two data points with a straight line

$$\frac{f_1(x) - f(x_0)}{x - x_0} = \frac{f(x_1) - f(x_0)}{x_1 - x_0}$$

$$f_1(x) = f(x_0) + \frac{f(x_1) - f(x_0)}{x_1 - x_0}(x - x_0)$$

Linear-interpolation formula

Slope

$f_1(x)$ designates a **first-order** interpolating polynomial.

Copyright © 2006 The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

Quadratic Interpolation

- If **three (3)** data points are available, the estimate is improved by introducing some curvature into the line connecting the points.
  A ***second-order polynomial*** (***parabola***) can be used for this purpose
- A simple procedure can be used to determine the values of the coefficients

Represents a second order polynomial

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

$$x = x_0 \qquad b_0 = f(x_0)$$

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

Could you figure out how to derive this using the above equation?

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

Copyright © 2006 The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

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

$$b_0 = f(x_0) \qquad b_1 = \frac{f(x_1) - f(x_0)}{x_1 - x_0}$$

$$x = x_2 \qquad b_2 = \frac{f(x_2) - f(x_0) - b_1(x_2 - x_0)}{(x_2 - x_0)(x_2 - x_1)} \qquad b_2 = \frac{f(x_2) - f(x_0) - \dfrac{f(x_1) - f(x_0)}{x_1 - x_0}(x_2 - x_0)}{(x_2 - x_0)(x_2 - x_1)}$$

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

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

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

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

Copyright © 2006 The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

General Form of Newton's Interpolating Polynomials

$$f_n(x) = b_0 + b_1(x - x_0) + b_2(x - x_0)(x - x_1) + \cdots b_n(x - x_0)(x - x_1)\cdots(x - x_{n-1})$$

$$b_0 = f(x_0) \qquad b_1 = f[x_1, x_0] \qquad b_2 = f[x_2, x_1, x_0] \ \ldots \ \ b_n = f[x_n, x_{n-1}, \cdots, x_1, x_0]$$

$$f[x_i, x_j] = \frac{f(x_i) - f(x_j)}{x_i - x_j}$$

Bracketed function evaluations are finite divided differences

$$f[x_i, x_j, x_k] = \frac{f[x_i, x_j] - f[x_j, x_k]}{x_i - x_k}$$

$$\vdots$$

$$f[x_n, x_{n-1}, \ldots, x_1, x_0] = \frac{f[x_n, x_{n-1}, \ldots, x_1] - f[x_{n-1}, x_{n-2}, \ldots, x_0]}{x_n - x_0}$$

Copyright © 2006 The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

DIVIDED DIFFERENCE TABLE

x_i	$f(x_i)$	$f[x_i,x_j]$	$f[x_i,x_j,x_k]$	$f[x,x,x,x]$
x_0	$f(x_0)$			
x_1	$f(x_1)$	$f[x_1,x_0]$		
x_2	$f(x_2)$	$f[x_2,x_1]$	$f[x_2,x_1,x_0]$	
x_3	$f(x_3)$	$f[x_3,x_2]$	$f[x_3,x_2,x_1]$	$f[x_3,x_2,x_1,x_0]$
x_4	$f(x_4)$	$f[x_4,x_3]$	$f[x_4,x_3,x_2]$	$f[x_4,x_3,x_2,x_1]$

EXAMPLE

x_i	$f(x_i)$	$f[x_i,x_j]$	$f[x_i,x_j,x_k]$	$f[x,x,x,x]$	$f[x...x]$
x_0=0	2				
x_1=2	14	6			
x_2=3	74	60	18		
x_3=4	242	168	54	9	
x_4=5	602	360	96	14	1

$f_1(x) = 2 + 6*(x-0)$ (based on x_0 and x_1)

$f_2(x) = 2 + 6*(x-0)+18(x-0)(x-2)$ (based on x_0, x_1 and x_2)

$f_3(x) = 2 + 6*(x-0)+18(x-0)(x-2)+9(x-0)(x-2)(x-3)$ (based on x_0, x_1, x_2, and x_3)

$f_4(x) = 2 + 6x +18x(x-2) +9x(x-2)(x-3) +1x(x-2)(x-3)(x-4)$ (based on x_0, x_1, x_2, x_3, and x_4)

$= x^4 - x^2 + 2$

Copyright © 2006 The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

Given:

$x_0=1 \quad f(x_0)=\ln(1) = 0$

$x_1=e \quad f(x_1)=\ln(2.72) = 1$

$x_2=e^2 \quad f(x_2)=\ln(7.39) = 2$

Estimate ***ln(2) = ?***

using **interpolation**

Find $f(x)$ first

x_i	$f(x_i)$	$f[x_i,x_j]$	$f[x_i,x_j,x_k]$
$x_0=1$	0		
$x_1=2.72$	1	.58	
$x_2=7.39$	2	.214	-.057

$$f(x) = 0.58(x-1) - 0.057(x-1)(x-2.72)$$

Then calculate

$f(2)=0.58(2-1)-0.057(2-1)(2-2.72)$

$= 0.621$

(* Approx. value for ln(2) *)

[TRUE ln(2) = 0.6931]

Copyright © 2006 The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

Yet Another Example:

Given:

x_0=1 $f(x_0)$=ln(1)=0

x_1=4 $f(x_1)$=ln(4)=1.386

x_2=6 $f(x_2)$=ln(6)=1.791

Estimate ***ln(2) = ?***

using **interpolation**

Find $f(x)$ first

x_i	$f(x_i)$	$f[x_i,x_j]$	$f[x_i,x_j,x_k]$
x_0=1	0		
x_1=4	1.386	.462	
x_2=6	1.79	.2025	-.052

$$f(x) = 0.462(x-1) - 0.052(x-1)(x-4)$$

Then calculate

$f(2)$=0.462*(2-1) -0.052*(2-1)(2-4)

= 0.566

[TRUE ln(2) = 0.693]

Copyright © 2006 The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

Error Estimation in Newton's Interpolating Polynomials

- Structure of *interpolating polynomials* is similar to the **Taylor series** expansion, i.e. finite divided differences are added sequentially to capture the higher order derivatives.

- For an n^{th}-order interpolating polynomial, an analogous relationship for the **error** is:

$$R_n = f_{n+1}(x) - f_n(x) \quad \text{in other words} \Rightarrow \quad f_{n+1}(x) = f_n(x) + R_n$$

- If an additional point $(x_{n+1}, f(x_{n+1}))$ is available, then

$$R_n \cong f[x_{n+1}, x_n, x_{n-1}, \ldots, x_0](x - x_0)(x - x_1)\cdots(x - x_n)$$
$$\cong b_{n+1}(x - x_0)(x - x_1)\cdots(x - x_n)$$

- This result is based on the assumption that the series is strongly **convergent**. i.e. *(n+1)th-order* prediction is closer to the true value than the *nth-order* prediction.

Copyright © 2006 The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

Previous Example:

$x_0=1 \quad f(x_0)=\ln(1) = 0$

$x_1=e \quad f(x_1)=\ln(2.72) = 1$

$x_2=e^2 \quad f(x_2)=\ln(7.39) = 2$

$$f(x) = 0.58(x-1) - 0.057(x-1)(x-2.72)$$

$f(2) = 0.621$

Example (for computing error):

In the previous Example, add a fourth point

$x_3=e^{0.5} = 1.6487 \quad f(x_3) = 0.5$

And estimate the approximate error in computing $ln(2)$

2nd order polynomial estimation: $f(2) = 0.621$

True error $= ln(2) - f(2) = 0.693 - 0.621 = \mathbf{0.072}$

Approximate Error:

$R_2 = f_3(x) - f_2(x) = b_3 (x- x_0)(x- x_1)(x- x_2)$

or

$R_2 = 0.0208*(x-1)(x-2.72)(x-7.39)$

If we plug-in $x=2$ in the above expression, we get:

$R_2 = 0.0208*(2-1)(2-2.72)(2-7.39) = \mathbf{0.0807}$

which is close to the true error (the same order of magnitude).

Copyright © 2006 The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

Lagrange Interpolating Polynomials

- The Lagrange interpolating polynomial is simply a reformulation of the Newton's polynomial that avoids the computation of divided differences:

$$f_n(x) = \sum_{i=0}^{n} L_i(x) f(x_i)$$

$$L_i(x) = \prod_{\substack{j=0 \\ j \neq i}}^{n} \frac{x - x_j}{x_i - x_j}$$

$$f_1(x) = \frac{x - x_1}{x_0 - x_1} f(x_0) + \frac{x - x_0}{x_1 - x_0} f(x_1)$$

$$f_2(x) = \frac{(x - x_1)(x - x_2)}{(x_0 - x_1)(x_0 - x_2)} f(x_0) + \frac{(x - x_0)(x - x_2)}{(x_1 - x_0)(x_1 - x_2)} f(x_1) + \frac{(x - x_0)(x - x_1)}{(x_2 - x_0)(x_2 - x_1)} f(x_2)$$

- Above formula can be easily verified by plugging in x_0, x_1…in the equation one at a time and checking if the equality is satisfied.

Copyright © 2006 The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

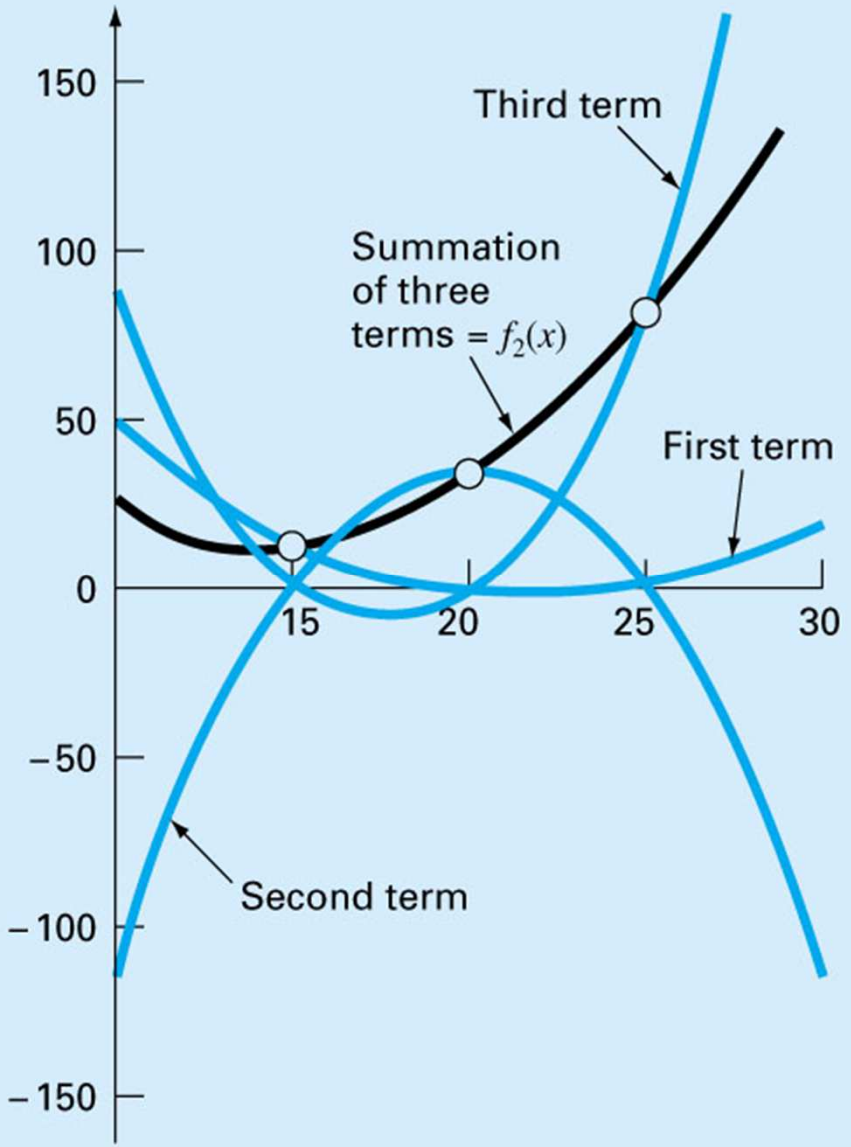
A visual depiction of the rationale behind the Lagrange polynomial . The figure shows a second order case:

$$f_2(x) = \frac{(x-x_1)(x-x_2)}{(x_0-x_1)(x_0-x_2)} f(x_0)$$

$$+ \frac{(x-x_0)(x-x_2)}{(x_1-x_0)(x_1-x_2)} f(x_1)$$

$$+ \frac{(x-x_0)(x-x_1)}{(x_2-x_0)(x_2-x_1)} f(x_2)$$

Each of the three terms passes through one of the data points and zero at the other two. The summation of the three terms must, therefore, be unique second order polynomial $f_2(x)$ that passes exactly through three points.



Copyright © 2006 The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

An example to demonstrate that 'order does not matter'

Use

$f(x) = 5 - 2x + x^2$

to demostrate that you get the same polynomial regardless of the order that the data points are listed

Use the following two tables and solve with :

(A) Nexton's divided differences

and

(B) Lagrange (using Table 1)

x_i	$f(x_i)$
x_0= 0	5
x_1= 1	4
x_2= 3	8

x_i	$f(x_i)$
x_0= 3	8
x_1= 1	4
x_2= 0	5

Copyright © 2006 The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

Coefficients of an Interpolating Polynomial

- Although both the Newton and Lagrange polynomials are well suited for determining intermediate values between points, they do not provide a polynomial in conventional form:

$$f(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_n x^n$$

- Since *n+1* data points are required to determine *n+1* coefficients, simultaneous linear systems of equations can be used to calculate "*a*"s.

$$\begin{aligned} f(x_0) &= a_0 + a_1 x_0 + a_2 x_0^2 \cdots + a_n x_0^n \\ f(x_1) &= a_0 + a_1 x_1 + a_2 x_1^2 \cdots + a_n x_1^n \\ &\vdots \\ f(x_n) &= a_0 + a_1 x_n + a_2 x_n^2 \cdots + a_n x_n^n \end{aligned}$$

$$\begin{bmatrix} 1 & x_0 & x_0^2 & \dots & x_0^n \\ 1 & x_1 & x_1^2 & \dots & x_1^n \\ 1 & x_2 & x_2^2 & \dots & x_2^n \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & x_n & x_n^2 & \dots & x_n^n \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ \vdots \\ a_n \end{bmatrix} = \begin{bmatrix} f(x_0) \\ f(x_1) \\ f(x_2) \\ \vdots \\ f(x_n) \end{bmatrix}$$

Where "*x*"s are the knowns and "*a*"s are the unknowns.

- Time complexity for Newton's: $O(n^2)$ vs. Gaussian Elim. $O(n^3)$
- Another reason: this approach results in a highly unstable (ill-conditioned) system of equations. Therefore, *Newton* or *Lagrange interpolation* is preferred.

Inverse Interpolation

Given $(x_0, f(x_0))$, $(x_1, f(x_1))$, $(x_2, f(x_2))$, …, $(x_n, f(x_n))$

and a function value $f(x_k)$, how do you determine (x_k) ?

1. Using one of the interpolation methods (Newton or Lagrange), obtain the function $f(x)$:

$$f(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_n x^n$$

2. Plug in the value of $f(x_k)$ in the above equation and find one of the real roots of the resultant equation using a root finding method:

$$a_0 + a_1 x + a_2 x^2 + \cdots + a_n x^n - f(x_k) = 0$$

Copyright © 2006 The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

Spline Interpolation

- There are cases where polynomials can lead to erroneous results because of round off error and overshoot.

- Alternative approach is to apply lower-order polynomials to subsets of data points. Such connecting polynomials are called ***spline functions***.

EXAMPLE:

spline fits for a set of 4 points:

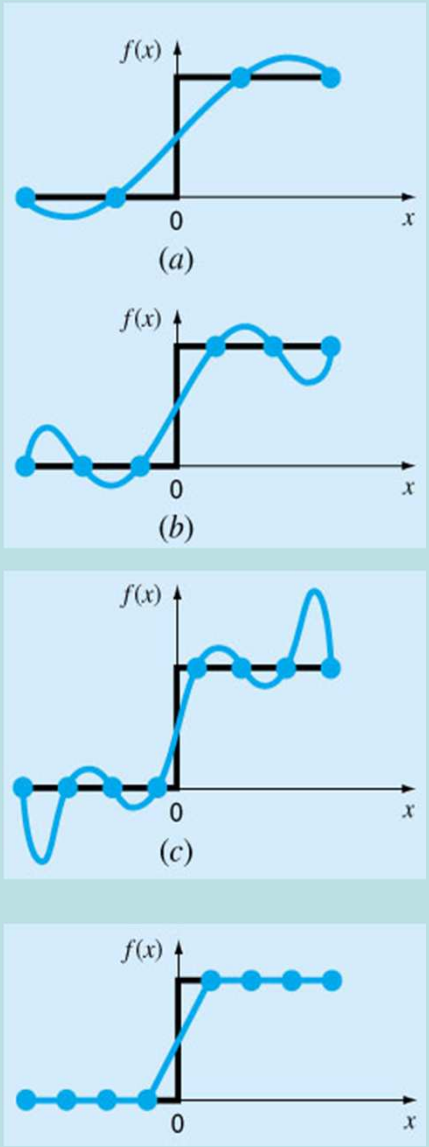
(a) linear

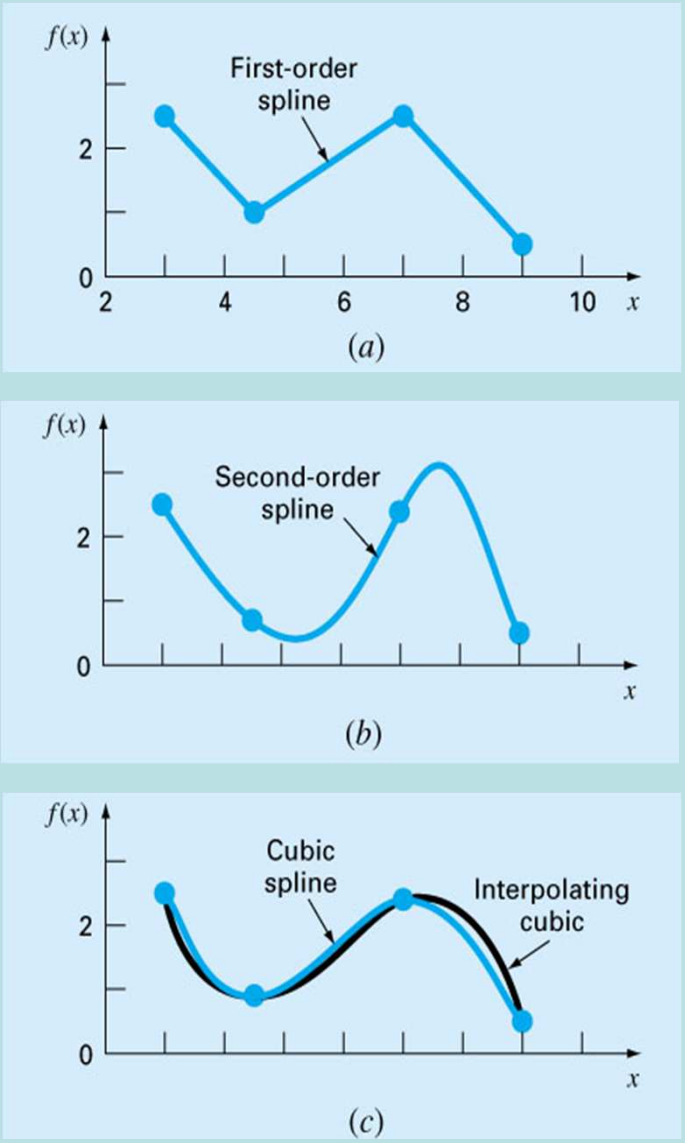
(b) quadratic

(c) cubic splines

- ***linear spline*** ➔➔➔

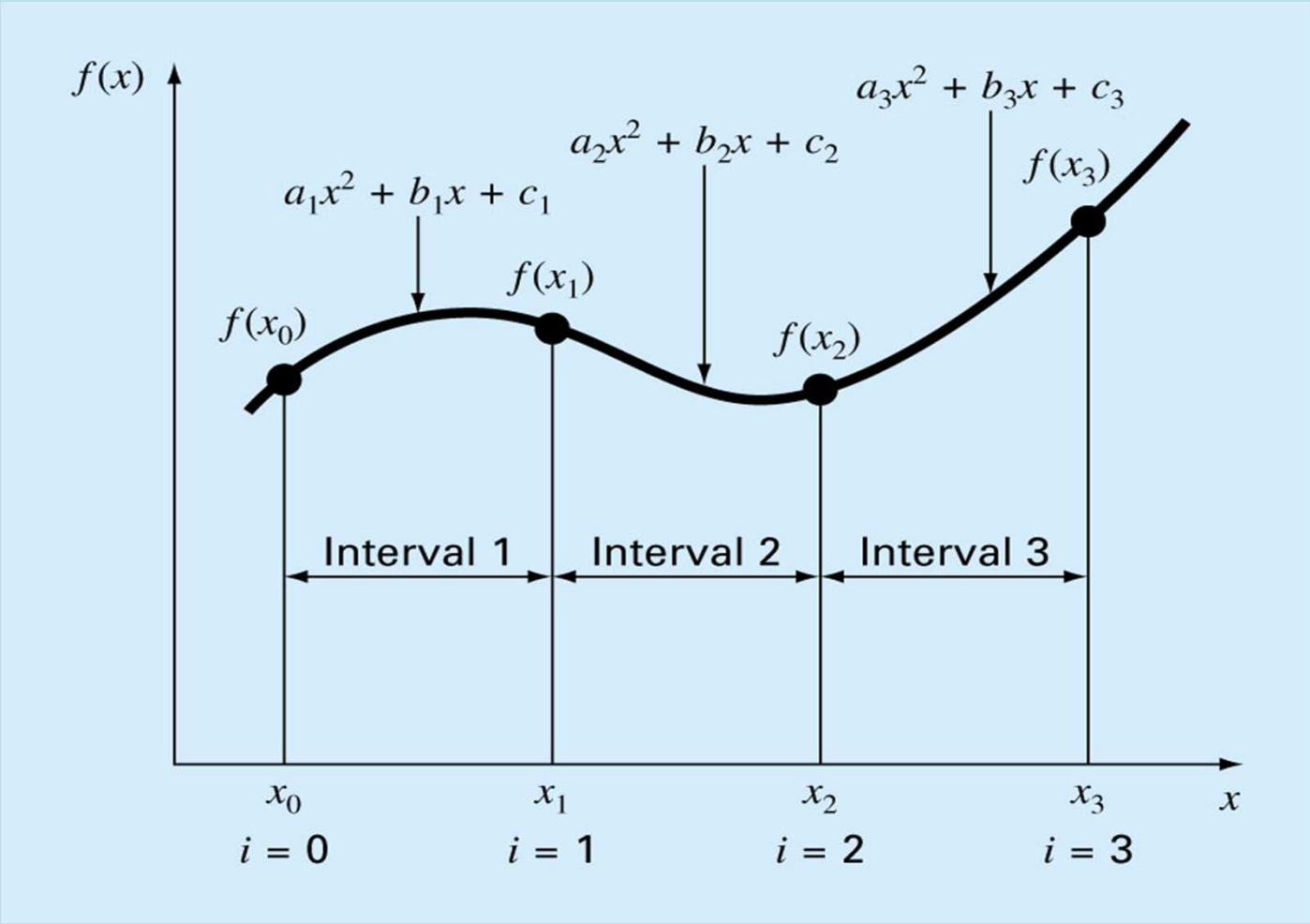
  is superior to higher-order interpolating polynomials (parts a, b, and c)





Copyright © 2006 The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

Quadratic Spline example

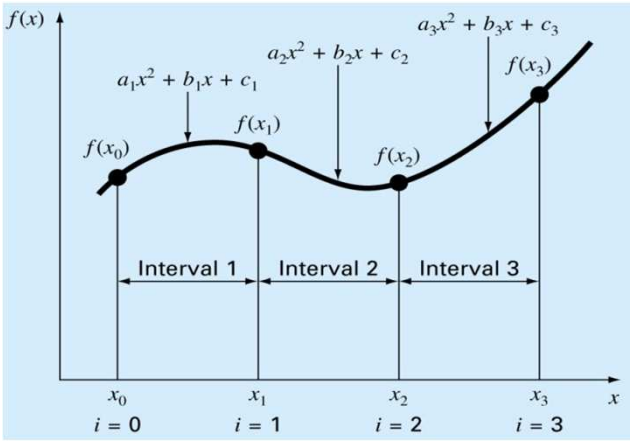


Copyright © 2006 The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

Quadratic Splines - Equations

Given (n+1) points, there are **3(n) coefficients** to find and hence, need to determine **3n equations** to solve:

1. Each polynomial must pass through the endpoints of the interval that they are associated with. And these conditions result in **2n equations**:



$$a_i x_i^2 + b_i x_i + c_i = f(x_i) \qquad i = 1,2,\ldots,n$$

$$a_i x_{i-1}^2 + b_i x_{i-1} + c_i = f(x_{i-1}) \qquad i = 1,2,\ldots,n$$

1. The first derivatives at the interior knots must be equal (**(n-1) equations**):

$$2a_i x_i + b_i = 2a_{i+1} x_i + b_{i+1} \qquad i = 1,2,\ldots,n-1$$

2. So far, we are one equation short. Since the last equation will be derived using only 2 points, it must be a line equation with only b_n and c_n to be determined (**$a_n = 0$**)

Copyright © 2006 The McGraw-Hill Companies, Inc. Permission required for reproduction or display.