Chapter 17

Polynomial Interpolation

Numerical Methods
Fall 2019

Polynomial Interpolation

- You will frequently have occasions to estimate intermediate values between precise data points.
- The function you use to interpolate must pass through the actual data points—this makes interpolation more restrictive than fitting.
- The most common method for this purpose is polynomial interpolation, where an (n-1)th order polynomial is solved that passes through n data points:

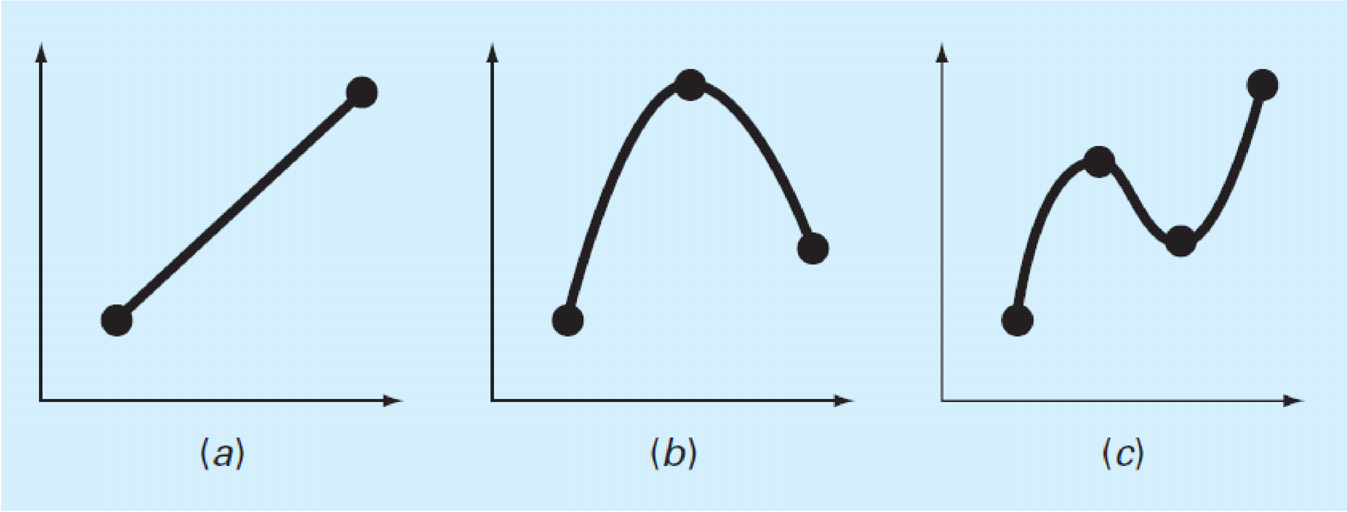
$$f(x) = a_1 + a_2x + a_3x^2 + \cdots + a_nx^{n-1}$$

MATLAB version:

$$f(x) = p_1x^{n-1} + p_2x^{n-2} + \cdots + p_{n-1}x + p_n$$

Determining Coefficients

- Since polynomial interpolation provides as many basis functions as there are data points (n), the polynomial coefficients can be found exactly using linear algebra.
- MATLAB's built in polyfit and polyval commands can also be used—all that is required is making sure the order of the fit for n data points is n–1.



Polynomial Interpolation Problems

- One problem that can occur with solving for the coefficients of a polynomial is that the system to be inverted is in the form:

$$\begin{bmatrix} x_1^{n-1} & x_1^{n-2} & \cdots & x_1 & 1 \\ x_2^{n-1} & x_2^{n-2} & \cdots & x_2 & 1 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ x_{n-1}^{n-1} & x_{n-1}^{n-2} & \cdots & x_{n-1} & 1 \\ x_n^{n-1} & x_n^{n-2} & \cdots & x_n & 1 \end{bmatrix} \begin{Bmatrix} p_1 \\ p_2 \\ \vdots \\ p_{n-1} \\ p_n \end{Bmatrix} = \begin{Bmatrix} f(x_1) \\ f(x_2) \\ \vdots \\ f(x_{n-1}) \\ f(x_n) \end{Bmatrix}$$

- Matrices such as that on the left are known as *Vandermonde matrices*, and they are very ill-conditioned—meaning their solutions are very sensitive to round-off errors.
- The issue can be minimized by scaling and shifting the data.

EXAMPLE 17.1

Newton Interpolating Polynomials, 1

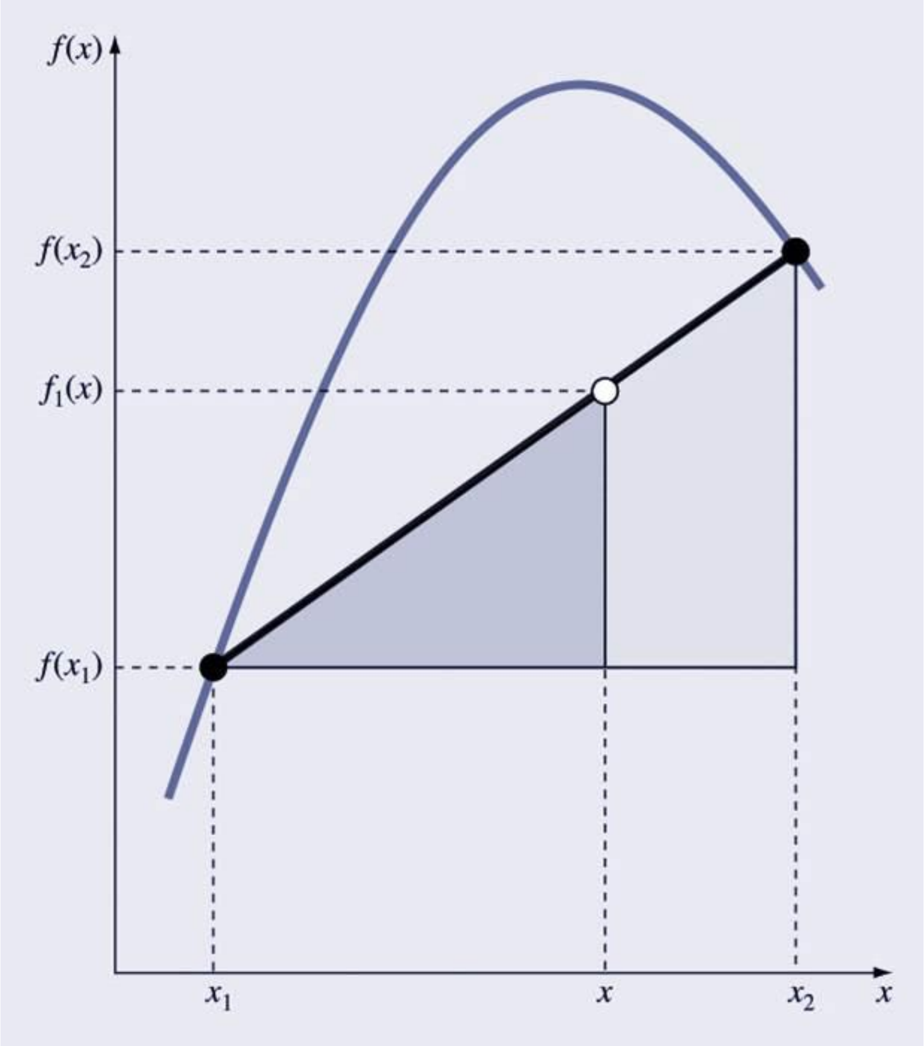
- Another way to express a polynomial interpolation is to use *Newton's interpolating polynomial*.
- The differences between a simple polynomial and Newton's interpolating polynomial for first and second order interpolations are:

Order	Simple	Newton
1st	$f_1(x) = a_1 + a_2 x$	$f_1(x) = b_1 + b_2(x - x_1)$
2nd	$f_2(x) = a_1 + a_2 x + a_3 x^2$	$f_2(x) = b_1 + b_2(x - x_1) + b_3(x - x_1)(x - x_2)$

Newton Interpolating Polynomials, 2

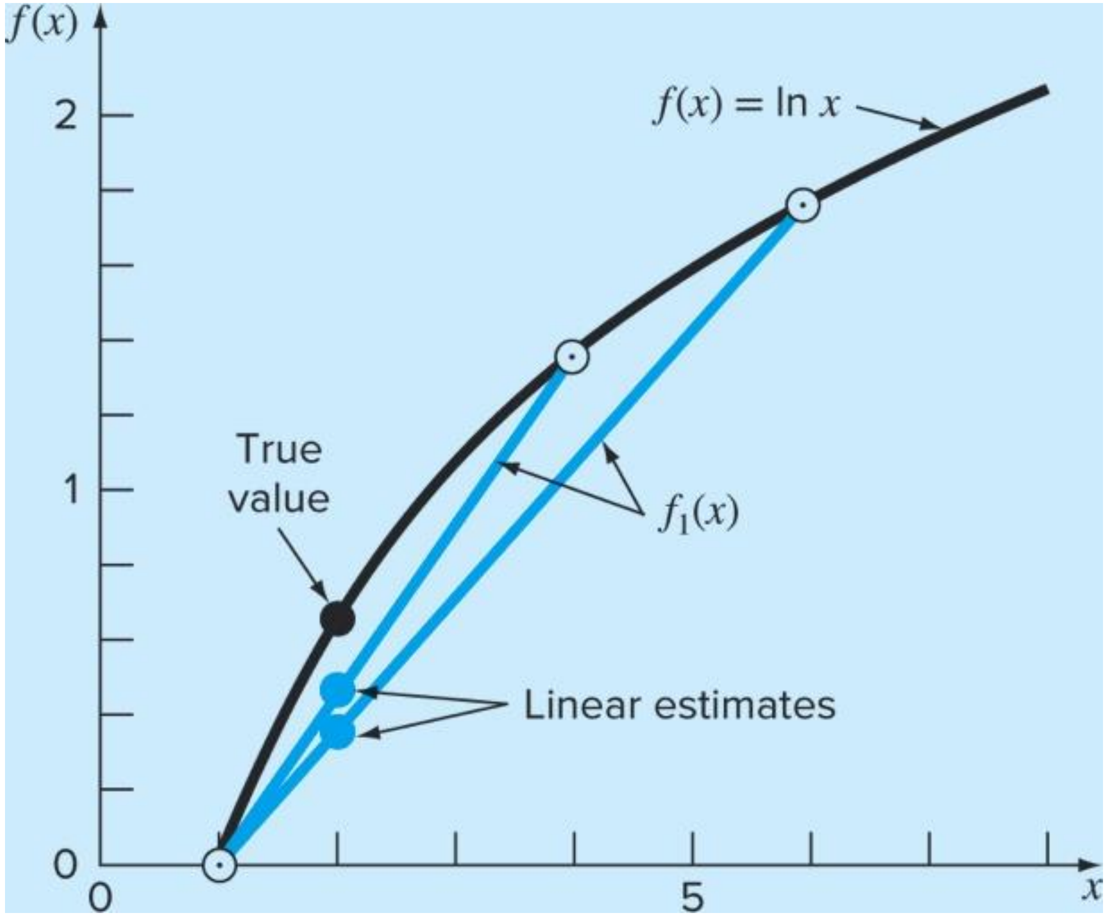
- The first-order Newton interpolating polynomial may be obtained from linear interpolation and similar triangles, as shown.
- The resulting formula based on known points x_1 and x_2 and the values of the dependent function at those points is:

$$f_1(x) = f(x_1) + \frac{f(x_2) - f(x_1)}{x_2 - x_1}(x - x_1)$$



Example

- Example 17.2
  - Estimate the natural logarithm of 2 using linear interpolation
  - What is $f_1(2)$?



1) interpolating between ln 1 = 0 and ln 6 = 1.791759

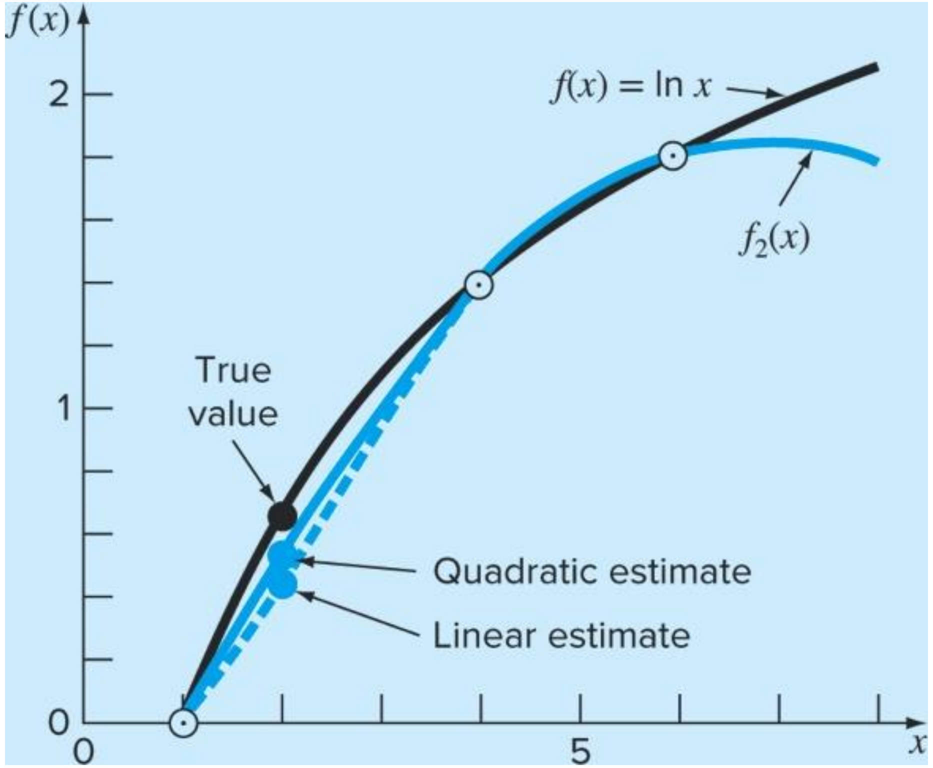
$$f_1(2) = 0 + \frac{1.791759 - 0}{6 - 1}(2 - 1) = 0.3583519$$

2) interpolating between ln 1 = 0 and ln 4 = 1.386294

$$f_1(2) = 0 + \frac{1.386294 - 0}{4 - 1}(2 - 1) = 0.4620981$$

Newton Interpolating Polynomials, 3

- The second-order Newton interpolating polynomial introduces some curvature to the line connecting the points, but still goes through the first two points.
- The resulting formula based on known points x_1, x_2, and x_3 and the values of the dependent function at those points is:



$$f_2(x) = f(x_1) + \frac{f(x_2) - f(x_1)}{x_2 - x_1}(x - x_1) + \frac{\dfrac{f(x_3) - f(x_2)}{x_3 - x_2} - \dfrac{f(x_2) - f(x_1)}{x_2 - x_1}}{x_3 - x_1}(x - x_1)(x - x_2)$$

EXAMPLE 17.3

Newton Interpolating Polynomials, 4

- The preceding analysis can be generalized to fit an (n − 1)th-order polynomial to n data points.
- The general formula is:

$$f_{n-1}(x) = b_1 + b_2(x - x_1) + \cdots + b_n(x - x_1)(x - x_1)\cdots(x - x_1)$$

where

$$\begin{aligned} b_1 &= f(x_1) \\ b_2 &= f[x_2, x_1] \\ b_3 &= f[x_3, x_2, x_1] \\ &\vdots \\ b_n &= f[x_n, x_{n-1}, \cdots, x_2, x_1] \end{aligned}$$

and the $f[\ldots]$ represent *divided differences*.

Divided Differences

- Divided difference are calculated as follows:

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

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

$$f[x_n, x_{n-1}, \cdots, x_2, x_1] = \frac{f[x_n, x_{n-1}, \cdots, x_2] - f[x_{n-1}, x_{n-2}, \cdots, x_1]}{x_n - x_1}$$

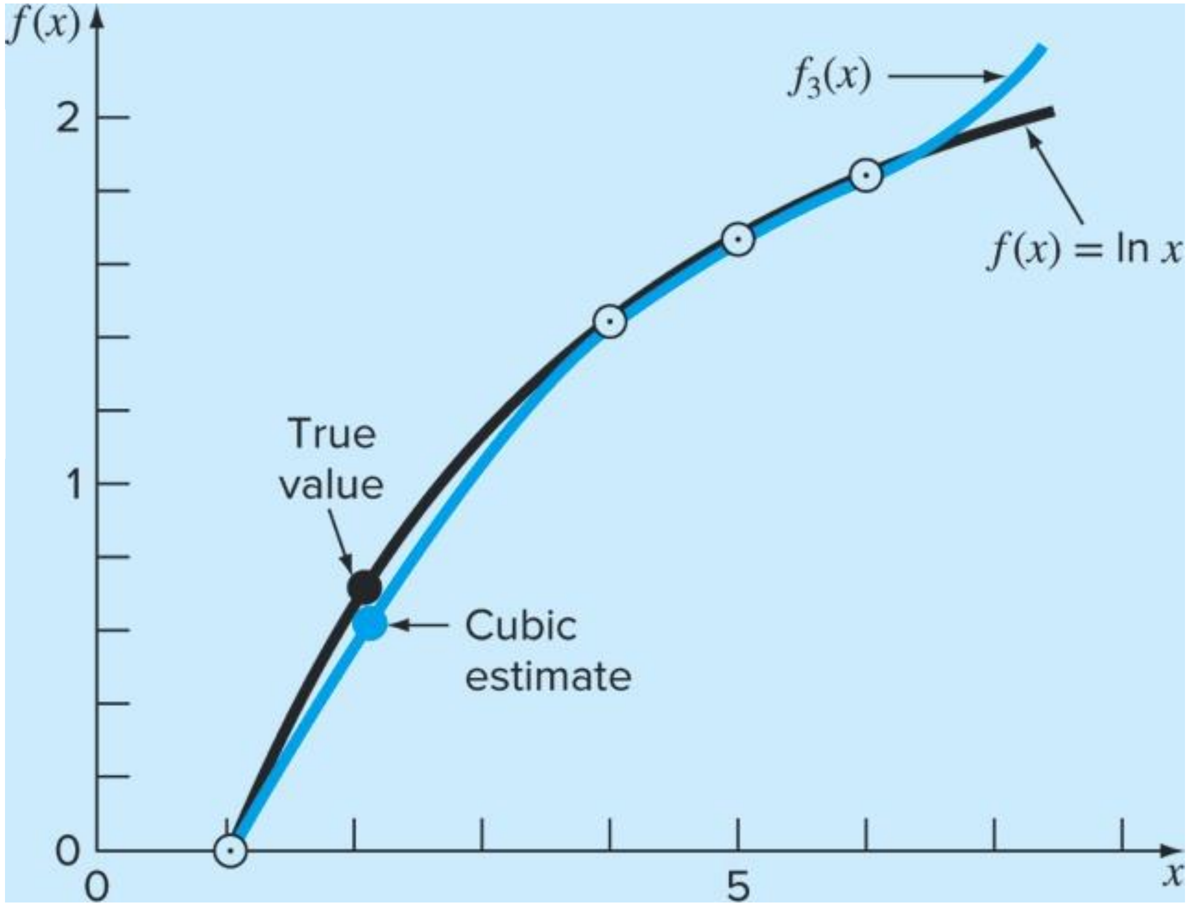
- Divided differences are calculated using the divided difference of the previous lower order:

x_i	$f(x_i)$	First	Second	Third
x_1	$f(x_1)$	$f[x_2, x_1]$	$f[x_3, x_2, x_1]$	$f[x_4, x_3, x_2, x_1]$
x_2	$f(x_2)$	$f[x_3, x_2]$	$f[x_4, x_3, x_2]$	
x_3	$f(x_3)$	$f[x_4, x_3]$		
x_4	$f(x_4)$			

Example

- Example 17.4

x_i	$f(x_i)$	First	Second	Third
1	0	0.4620981	−0.05187311	0.007865529
4	1.386294	0.2027326	−0.02041100	
6	1.791759	0.1823216		
5	1.609438			



Lagrange Interpolating Polynomials, 1

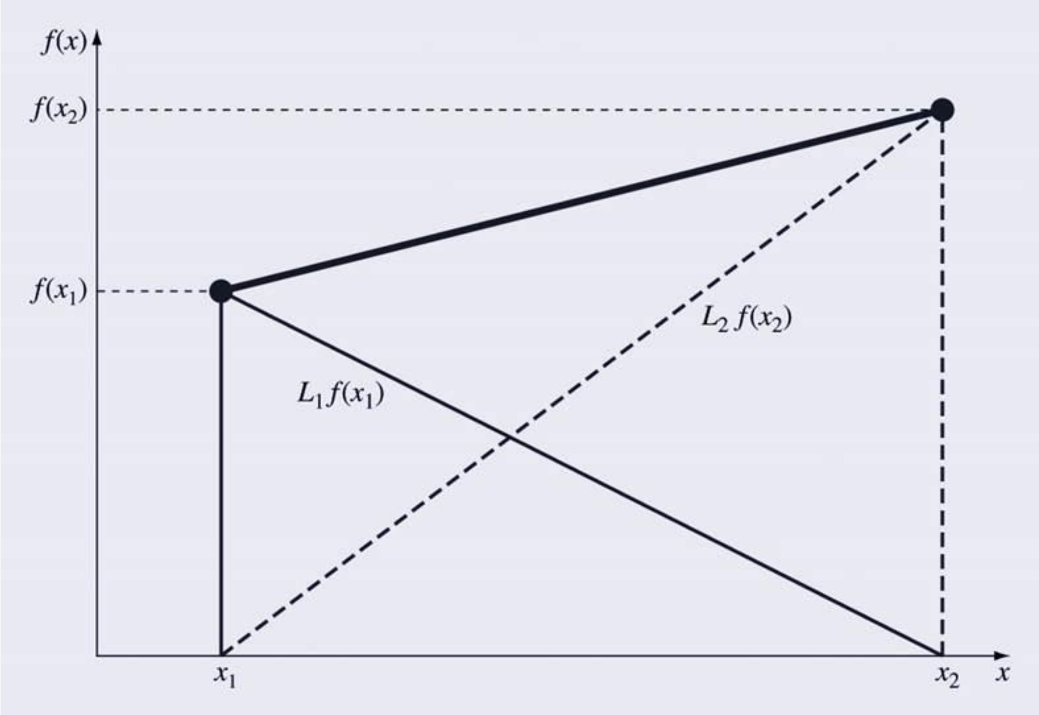
- Another method that uses shifted value to express an interpolating polynomial is the *Lagrange interpolating polynomial*.
- The differences between a simply polynomial and Lagrange interpolating polynomials for first and second order polynomials is:

Order	Simple	Lagrange
1st	$f_1(x) = a_1 + a_2 x$	$f_1(x) = L_1 f(x_1) + L_2 f(x_2)$
2nd	$f_2(x) = a_1 + a_2 x + a_3 x^2$	$f_2(x) = L_1 f(x_1) + L_2 f(x_2) + L_3 f(x_3)$

where the L_i are weighting coefficients that are functions of x.

Lagrange Interpolating Polynomials, 2

- The first-order Lagrange interpolating polynomial may be obtained from a weighted combination of two linear interpolations, as shown.
- The resulting formula based on known points x_1 and x_2 and the values of the dependent function at those points is:



$$f_1(x) = L_1 f(x_1) + L_2 f(x_2)$$

$$L_1 = \frac{x - x_2}{x_1 - x_2}, L_2 = \frac{x - x_1}{x_2 - x_1}$$

$$f_1(x) = \frac{x - x_2}{x_1 - x_2} f(x_1) + \frac{x - x_1}{x_2 - x_1} f(x_2)$$

Linear Lagrange Interpolating Polynomial.

Lagrange Interpolating Polynomials, 3

- Second-order Lagrange interpolating polynomial.

$$f_2(x) = \frac{(x-x_2)(x-x_3)}{(x_1-x_2)(x_1-x_3)} f(x_1) + \frac{(x-x_1)(x-x_3)}{(x_2-x_1)(x_2-x_3)} f(x_2) + \frac{(x-x_1)(x-x_2)}{(x_3-x_1)(x_3-x_2)} f(x_3)$$

- In general, the Lagrange polynomial interpolation for *n* points is:

$$f_{n-1}(x_i) = \sum_{i=1}^{n} L_i(x) f(x_i)$$

where L_i is given by:

$$L_i(x) = \prod_{\substack{j=1 \\ j\neq i}}^{n} \frac{x - x_j}{x_i - x_j}$$

EXAMPLE 17.5

Inverse Interpolation

- Interpolation general means finding some value $f(x)$ for some x that is between given independent data points.
- Sometimes, it will be useful to find the x for which $f(x)$ is a certain value—this is *inverse interpolation*.
- Rather than finding an interpolation of x as a function of $f(x)$, it may be useful to find an equation for $f(x)$ as a function of x using interpolation and then solve the corresponding roots problem: $f(x) - f_{\text{desired}} = 0$ for x.

Extrapolation

- *Extrapolation* is the process of estimating a value of $f(x)$ that lies outside the range of the known base points $x_1, x_2, \ldots, x_n$.

- Extrapolation represents a step into the unknown, and extreme care should be exercised when extrapolating!