Revision

- Create an array, *x*, starting at 25 decrementing by 7 until -100
- Create a matrix **A** with 3 columns, the first column is x, the second column is x+5,the third is x^2
- Plot **A** using the function `imagesc`
- Find all the elements in **A** that are less than 0
- Set the elements of **A** less than 0 to NaN
- Open a new figure and replot **A**
- Add a color bar to **A**

Interpolation and Extrapolation

Interpolation and Extrapolation

We sometimes know the value of a function $f(x)$ at a set of points $x_0, x_1, \ldots, x_{N-1}$ (say, with $x_0 < \ldots < x_{N-1}$), but we don't have an analytic expression for $f(x)$ that lets us calculate its value at an arbitrary point. For example, the $f(x_i)$'s might result from some physical measurement or from long numerical calculation that cannot be cast into a simple functional form. Often the x_i's are equally spaced, but not necessarily.

The task now is to estimate $f(x)$ for arbitrary x by, in some sense, drawing a smooth curve through (and perhaps beyond) the x_i. If the desired x is in between the largest and smallest of the x_i's, the problem is called *interpolation*; if x is outside that range, it is called *extrapolation*, which is considerably more hazardous (as many former investment analysts can attest).

Interpolate. From *inter* meaning between and *pole*, the points or nodes. Any means of calculating a new point between two or more existing data points is interpolation.

Extrapolation is the process of constructing new data points ***outside*** a discrete set of known data points. It is similar to the process of interpolation, but the results of extrapolations are often less meaningful, and are subject to greater uncertainty.

Local interpolation Using just the M nearest neighbors. May not have continuous first or higher order derivatives.

The number M of points used in an interpolation scheme, minus 1, is called the order of the interpolation. Higher order does not necessarily mean higher accuracy.

Functional forms used for interpolation

- Linear
- Polynomials
- Spline
- Rational functions (quotients of polynomials)
- Trigonometric functions (fourier methods)

- Non-parametric:
    - Neural Networks
    - Support Vector Machines
    - Gaussian Process Models
    - Decision Trees
    - Random Forests

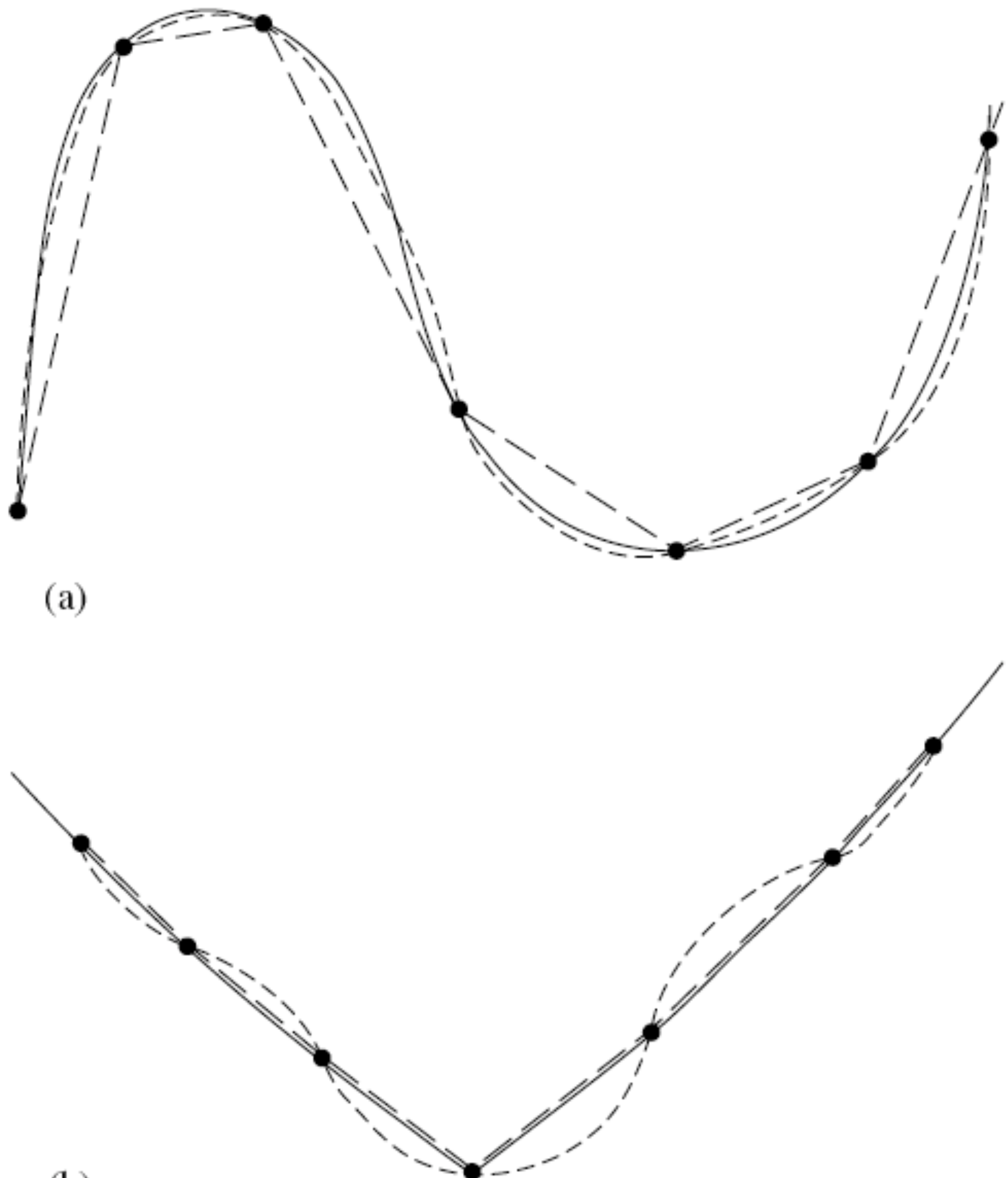


Figure 3.0.1. (a) A smooth function (solid line) is more accurately interpolated by a high-order polynomial (shown schematically as dotted line) than by a low-order polynomial (shown as a piecewise linear dashed line). (b) A function with sharp corners or rapidly changing higher derivatives is *less* accurately approximated by a high-order polynomial (dotted line), which is too "stiff," than by a low-order polynomial (dashed lines). Even some smooth functions, such as exponentials or rational functions, can be badly approximated by high-order polynomials.

Searching an Ordered Table

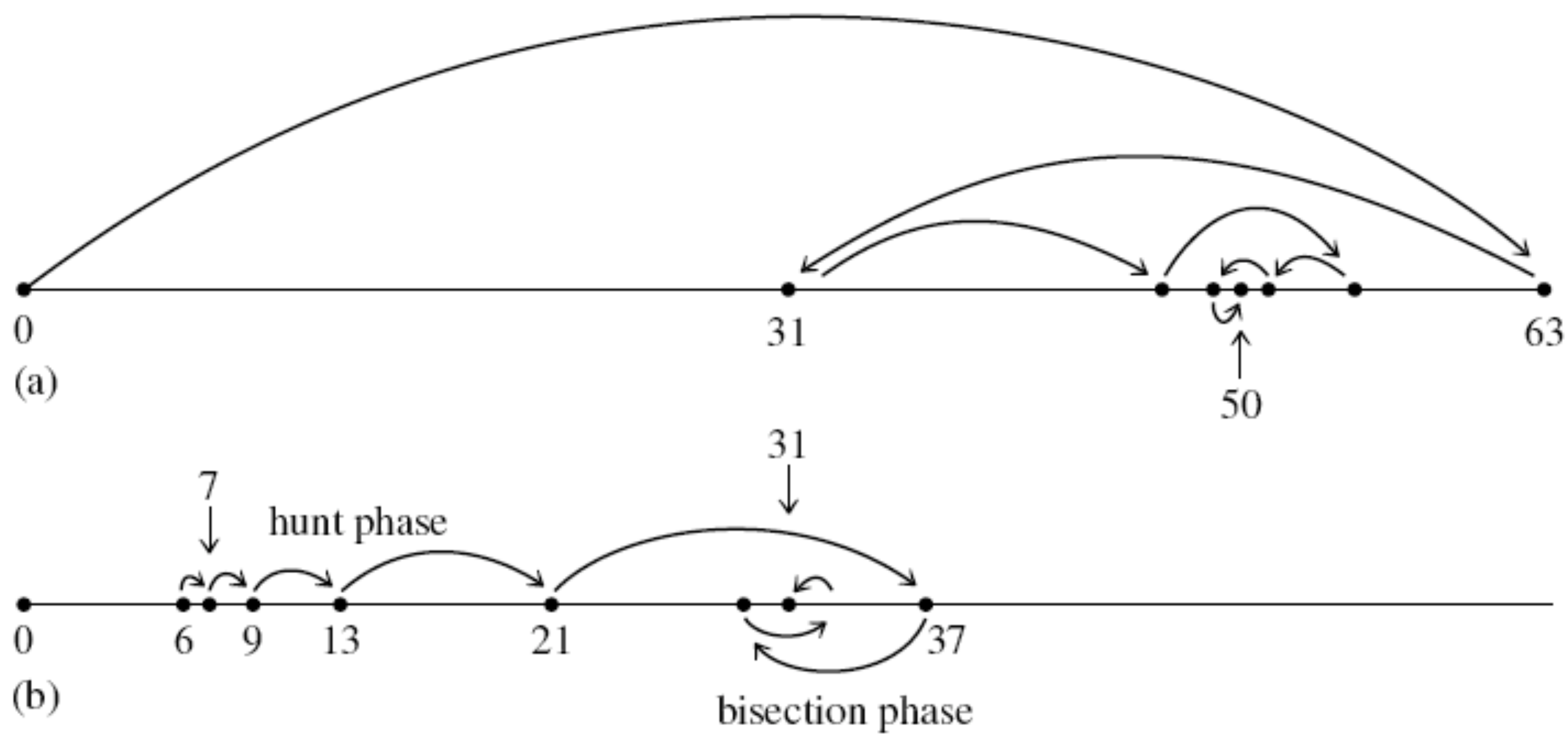


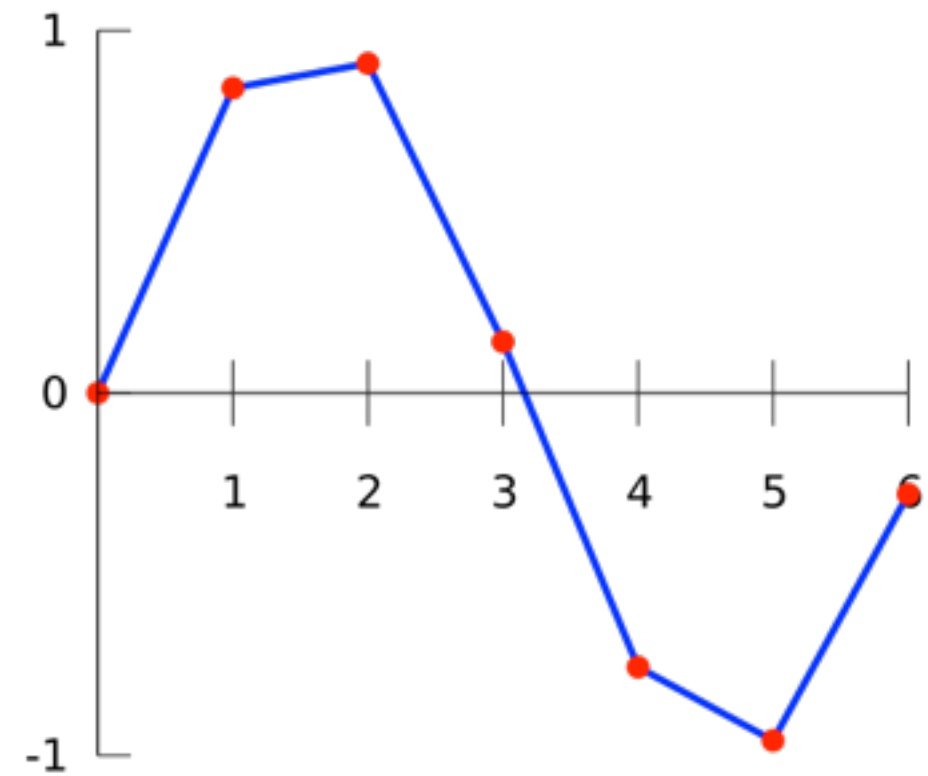
Figure 3.1.1. Finding a table entry by bisection. Shown here is the sequence of steps that converge to element 50 in a table of length 64. (b) The routine `hunt` searches from a previous known position in the table by increasing steps and then converges by bisection. Shown here is a particularly unfavorable example, converging to element 31 from element 6. A favorable example would be convergence to an element near 6, such as 8, which would require just three "hops."

Linear Interpolation

Generally, linear interpolation takes two data points, say (x_a,y_a) and (x_b,y_b), and the interpolant is given by:

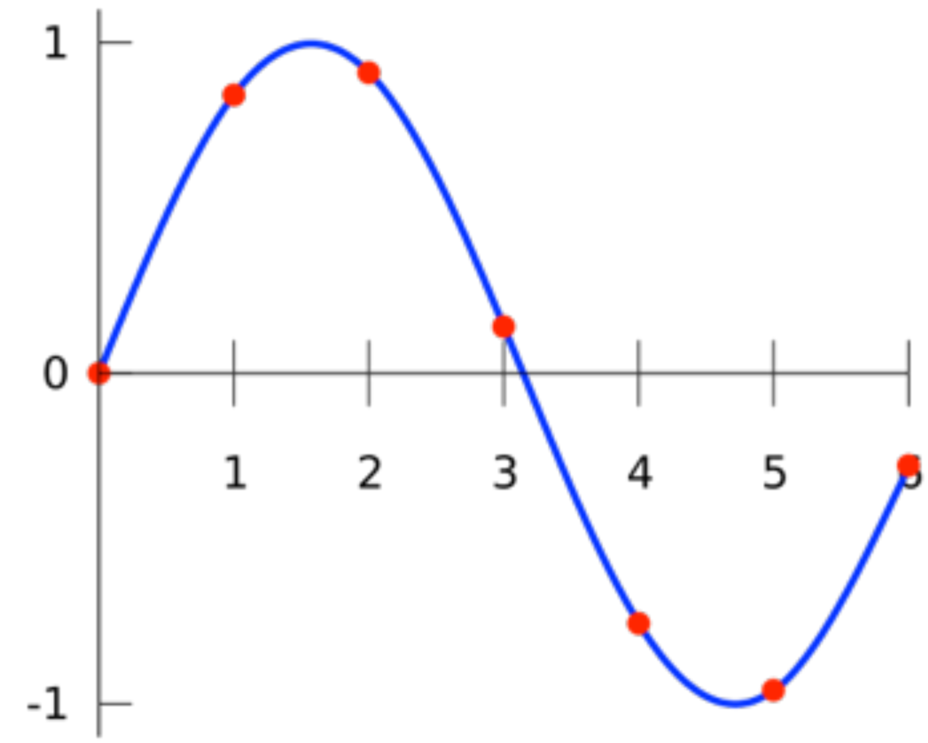
$$y = y_a + \frac{(x - x_a)(y_b - y_a)}{(x_b - x_a)} \text{ at the point } (x,y).$$

Linear interpolation is quick and easy, but it is not very precise. Another disadvantage is that the interpolant is not differentiable at the point x_k.



Spline Interpolation

Remember that linear interpolation uses a linear function for each of intervals $[x_k, x_{k+1}]$. Spline interpolation uses low-degree polynomials in each of the intervals, and chooses the polynomial pieces such that they fit smoothly together. The resulting function is called a spline.



Example

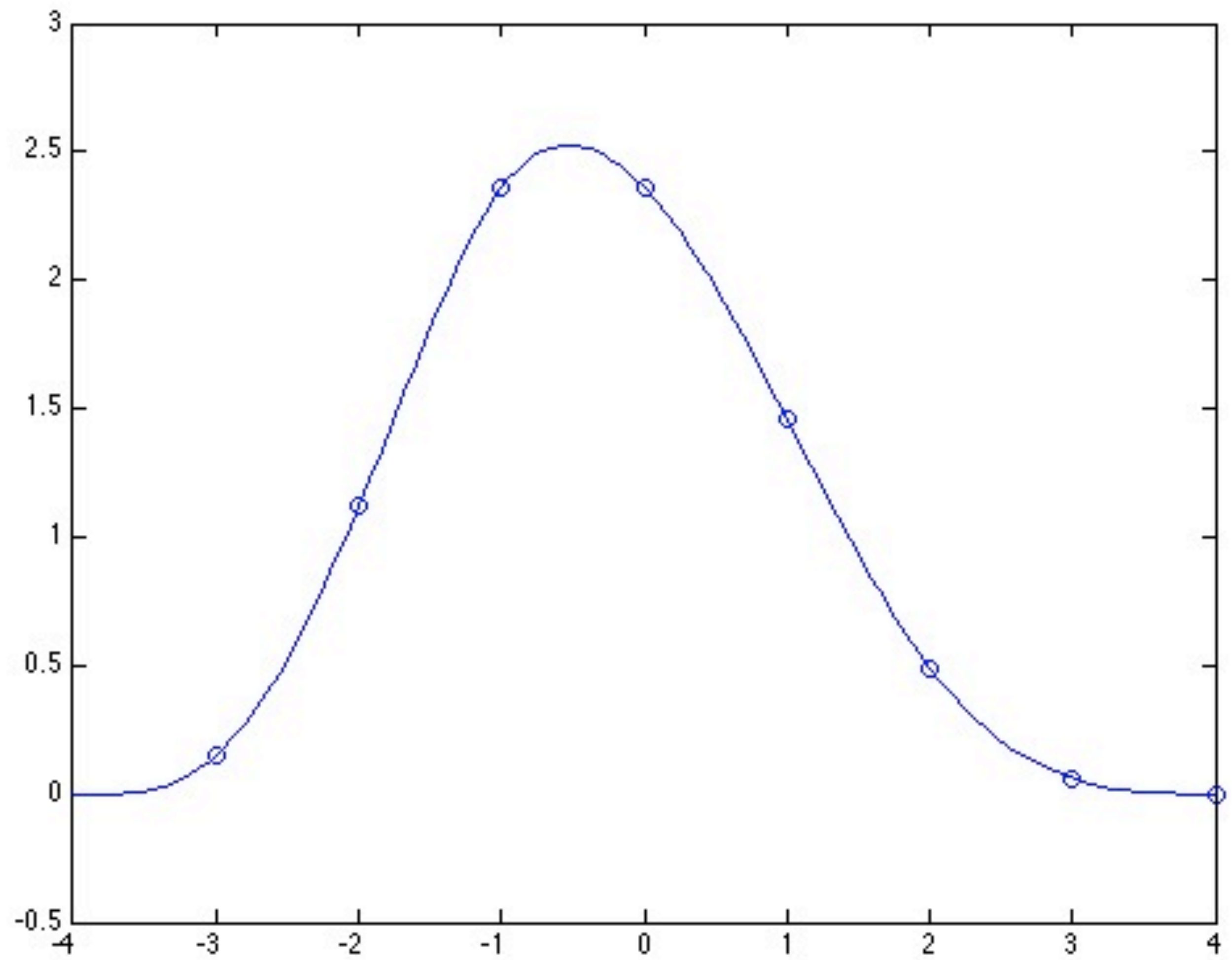
interpol.m

```
     % Generate a dataset
x = -4:4;
y = [0 .15 1.12 2.36 2.36 1.46 .49 .06 0];
plot(x,y,'o')
hold on

%--------------------------------------------------
% Cubic spline interpolation
cs = spline(x,[0 y 0]);
xx = linspace(-4,4,101);
plot(xx,ppval(cs,xx),'-');

%--------------------------------------------------

hold off
```



Example

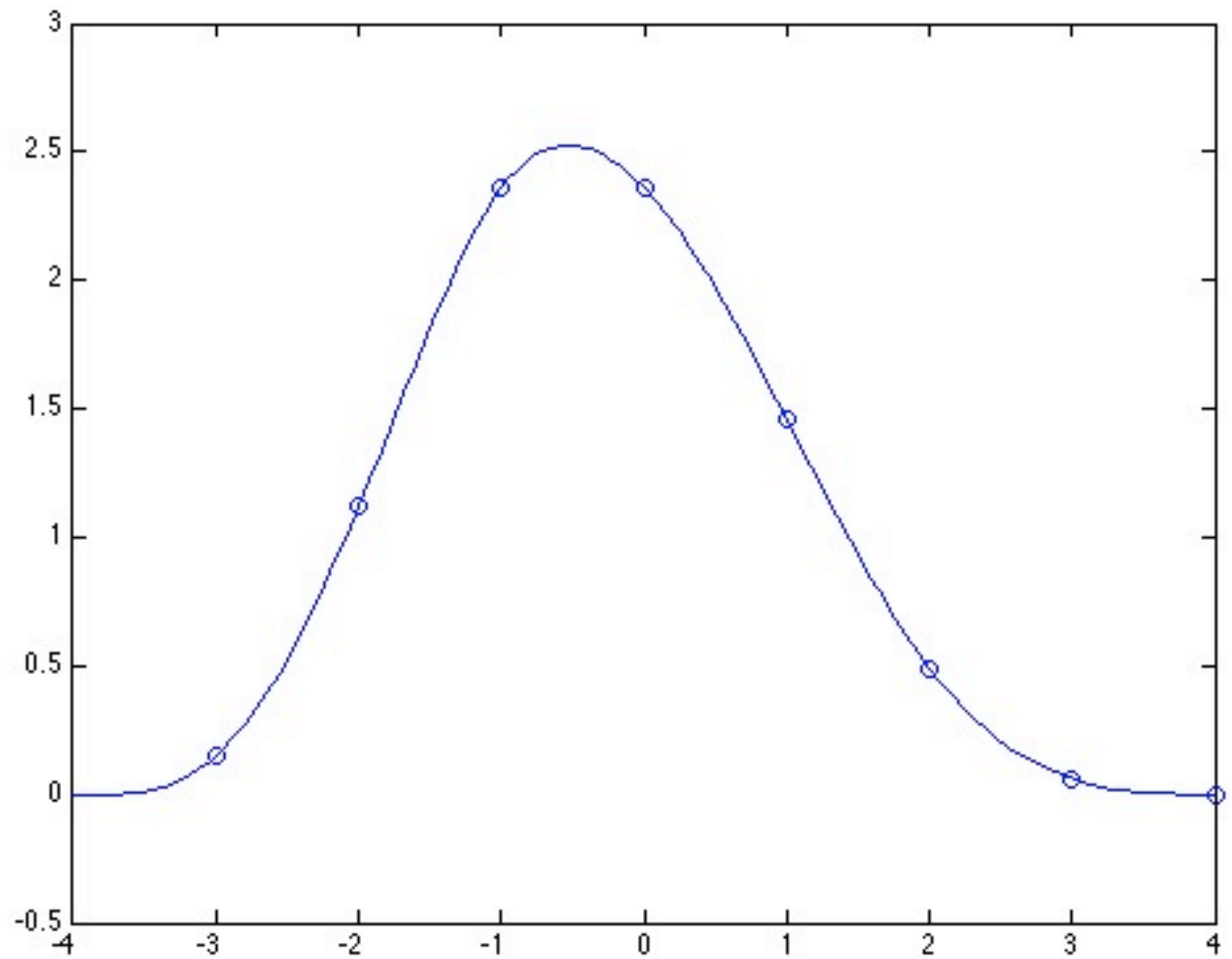
interpol.m

```
     % Generate a dataset
x = -4:4;
y = [0 .15 1.12 2.36 2.36 1.46 .49 .06 0];
plot(x,y,'o')
hold on

%--------------------------------------------------
% Cubic spline interpolation
cs = spline(x,[0 y 0]);
xx = linspace(-4,4,101);
plot(xx,ppval(cs,xx),'-');

%--------------------------------------------------

hold off
```



In the Figure window click on tools "basic fitting"

Rationals

A rational is the quotient of two numbers

$$y = \frac{\sum_{i=1}^{n+1} p_i x^{n+1-i}}{x^m + \sum_{i=1}^{m} q_i x^{m-1}}$$

Rationals

A rational is the quotient of two numbers

$$y = \frac{\sum_{i=1}^{n+1} p_i x^{n+1-i}}{x^m + \sum_{i=1}^{m} q_i x^{m-1}}$$

Open the Help Browser and in “Search Results” type “Fitting Data” and go through all the examples given.

Interpolation in Multiple Dimensions

In two dimensions, we imagine that we are given a matrix of functional values y_{ij}, with $i = 0, \ldots, M-1$ and $j = 0, \ldots, N-1$. We are also given an array of x_1 values x_{1i}, and an array of x_2 values x_{2j}, with i and j as just stated. The relation of these input quantities to an underlying function $y(x_1, x_2)$ is just

$$y_{ij} = y(x_{1i}, x_{2j}) \tag{3.6.1}$$

We want to estimate, by interpolation, the function y at some untabulated point (x_1, x_2).

An important concept is that of the *grid square* in which the point (x_1, x_2) falls, that is, the four tabulated points that surround the desired interior point. For convenience, we will number these points from 0 to 3, counterclockwise starting from the lower left. More precisely, if

$$\begin{aligned} x_{1i} &\le x_1 \le x_{1(i+1)} \\ x_{2j} &\le x_2 \le x_{2(j+1)} \end{aligned} \tag{3.6.2}$$

defines values of i and j, then

$$\begin{aligned} y_0 &\equiv y_{ij} \\ y_1 &\equiv y_{(i+1)j} \\ y_2 &\equiv y_{(i+1)(j+1)} \\ y_3 &\equiv y_{i(j+1)} \end{aligned} \tag{3.6.3}$$

Interpolation in Multiple Dimensions

The simplest interpolation in two dimensions is *bilinear interpolation* on the grid square. Its formulas are

$$t \equiv (x_1 - x_{1i})/(x_{1(i+1)} - x_{1i})$$
$$u \equiv (x_2 - x_{2j})/(x_{2(j+1)} - x_{2j}) \tag{3.6.4}$$

(so that t and u each lie between 0 and 1) and

$$y(x_1, x_2) = (1-t)(1-u)y_0 + t(1-u)y_1 + tuy_2 + (1-t)uy_3 \tag{3.6.5}$$

Interpolation of Scattered Data in Multiple-Dimensions

- Radial Basis Functions
- Krigging

Reading Assignment
Numerical Recipes
Chapter 3