Interpolation i Matlab

Dag Lindbo, 2011-01-31

clear all, close all

X = [1 4 5]';
Y = [1 3 1]';

% ekvationssystemet
C = [ones((size(X))) X X.^2]'\Y
f = @(x) C(1) + C(2)*x + C(3)*x.^2;

% derivera: df = c(2) + 2*c(3)*x = 0 =>
xmax = -c(2)/(2*c(3))

x = linspace(1,5,100);
plot(X,Y,'r.',x,f(x),'b',xmax,f(xmax),'rx','MarkerSize',20),
axis([0 6 0 4]), grid on

c =
-2.333333333333333  
4.000000000000000  
-0.666666666666667

xmax =
3
Med inbyggda funktioner

\begin{verbatim}
X = [1 4 5 7 8]';
Y = [1 3 1 4 5]';

% anpassa polynom
C = polyfit(X,Y,4)
x = linspace(1,8,100);
y = polyval(C,x); % evaluera polynomet i punkterna x

figure()
plot(X,Y,'r.',x,y,'MarkerSize',20), grid on

% styckvis kubisk
help pchip

figure()
C = pchip(X,Y)
y = ppval(C,x);
plot(X,Y,'r.',x,y,'MarkerSize',20), grid on
\end{verbatim}

\begin{verbatim}
c =
Columns 1 through 3
-0.091269841269840 1.857142857142832 -12.757936507936355
Columns 4 through 5
33.214285714285381 -21.222222222222076

PCHIP Piecewise Cubic Hermite Interpolating Polynomial.
PP = PCHIP(X,Y) provides the piecewise polynomial form of a certain
shape-preserving piecewise cubic Hermite interpolant, to the values
Y at the sites X, for later use with PPVAL and the spline utility UNMKPP.
X must be a vector.
If Y is a vector, then Y(j) is taken as the value to be matched at X(j),
therefore Y must be of the same length as X.
If Y is a matrix or ND array, then Y(:,...,:,j) is taken as the value to
be matched at X(j), therefore the last dimension of Y must equal length(X).

YY = PCHIP(X,Y,XX) is the same as YY = PPVAL(PCHIP(X,Y),XX), thus
providing, in YY, the values of the interpolant at XX.

The PCHIP interpolating function, p(x), satisfies:
On each subinterval, X(k) <= x <= X(k+1), p(x) is the cubic Hermite
interpolant to the given values and certain slopes at the two endpoints.
Therefore, p(x) interpolates Y, i.e., p(X(j)) = Y(:,j), and
the first derivative, Dp(x), is continuous, but
D^2p(x) is probably not continuous; there may be jumps at the X(j).
The slopes at the X(j) are chosen in such a way that
p(x) is "shape preserving" and "respects monotonicity". This means that,
on intervals where the data is monotonic, so is p(x);
at points where the data have a local extremum, so does p(x).

Comparing PCHIP with SPLINE:
The function s(x) supplied by SPLINE is constructed in exactly the same way,
except that the slopes at the X(j) are chosen differently, namely to make
even D^2s(x) continuous. This has the following effects.
SPLINE is smoother, i.e., D^2s(x) is continuous.
SPLINE is more accurate if the data are values of a smooth function.
PCHIP has no overshoots and less oscillation if the data are not smooth.
PCHIP is less expensive to set up.
The two are equally expensive to evaluate.

Example:
X = -3:3;
\end{verbatim}
\[ y = [-1 -1 -1 0 1 1 1]; \]
\[ t = -3:.01:3; \]
\[ plot(x,y,'o',t,[pchip(x,y,t); spline(x,y,t)]) \]
\[ legend('data','pchip','spline',4) \]

Class support for inputs x, y, xx:
float: double, single

See also INTERP1, SPLINE, PPVAL, UNMKPP.

Reference page in Help browser
doc pchip

c =
form: 'pp'
brakes: [1 4 5 7 8]
coefs: [4x4 double]
pieces: 4
order: 4
dim: 1
Published with MATLAB® 7.10