Cubic Splines in Matlab

D. Hundley

November 1, 2003

To work with piecewise defined polynomials efficiently, Matlab uses a data structure to compactly hold all of the necessary ingredients. When you construct a cubic spline, this data structure is what will be returned, and this is the information it contains:

- `form` For cubic splines, this will always be `pp` for piecewise polynomial.
- `breaks` A vector with the domain data. Using n points will mean that this vector stores n values.
- `coefs` An array that will be $n-1 \times 4$ for cubic splines, storing the coefficients of each of the $n-1$ cubic polynomials (the coefficients for the cubic term are in the first column).
- `pieces` The number of polynomials used (for cubic splines, this is always $n-1$).
- `order` This is always 4 for cubic splines.
- `dim` The dimension of the output; for cubic splines, this is always 1.

Here's an example that will construct a spline (using the "Not-a-knot" condition):

```
x = -4:4;
y = [0 .15 1.12 2.36 2.36 1.46 .49 .06 0];
cs = spline(x,y);

xx = linspace(-4,4,101);
yy = ppval(cs,xx);
plot(x,y,'o',xx,yy,'-');
```

The first three lines constructs the domain, the range, and puts the cubic spline into the data structure `cs`. If you type `cs`, Matlab will show you what is stored there.
In the last three lines, we sample the domain at many more points, evaluate the cubic splines at these data points (note the use of the command `ppval` to do this), and finally it plots the original data together with the cubic splines.
Additionally, one can define the derivatives at the endpoints (remember those two extra equations)- To do this is Matlab, use:

```
cs = spline(x, [ a, y, b]);
```

where a, b are the desired slopes.
Alternatively, one can use the GUI tools. To do this, first plot your data, then in the Figure, click on: `Tools` → `Basic Fitting`. The Basic Fitting GUI is now shown.
Next, click on either `spline interpolant` or the related `shape preserving interpolant`. To put the piecewise polynomial data structure into your workspace, go to the second panel of the basic fitting GUI (use the arrow at the bottom right), and choose the `Save to Workspace` button.
Back in the command window, you can take a look at what this is. For example, if I had Matlab save the structure as `fit1`, the data structure corresponding to `cs` in our previous examples is stored as `fit1.coeff`. Here's an example:

```
>> plot(x,y,'o')
Basic Fitting GUI created variables in the current workspace.
>> fit1
fit1 =
     type: 'spline'
    coeff: [1x1 struct]

>> fit1.coeff
ans =
      form: 'pp'
    breaks: [-4 -3 -2 -1 0 1 2 3 4]
     coefs: [8x4 double]
    pieces: 8
     order: 4
       dim: 1
```

"Spline Interpolant" (SI) or "Shape Preserving Interpolant" (SPI)?

They are both piecewise cubic functions, however, the SPI may not give a continuous second derivative. Internally, for SI, the Basic Fitting menu will call on `spline`, and for SPI, Matlab will call `pchip` (for more information, use Matlab's help feature).
Here are some differences (from Matlab's website) between `spline` and `pchip`, together with an illustrative example:

- `spline` produces a smoother result, i.e. is continuous.
- `spline` produces a more accurate result if the data consists of 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.

```
x = -3:3;
y = [-1 -1 -1 0 1 1 1];
t = -3:.01:3;
p = pchip(x,y,t);
s = spline(x,y,t);
plot(x,y,'o',t,p,'-',t,s,'-.')
legend('data','pchip','spline',4)
```

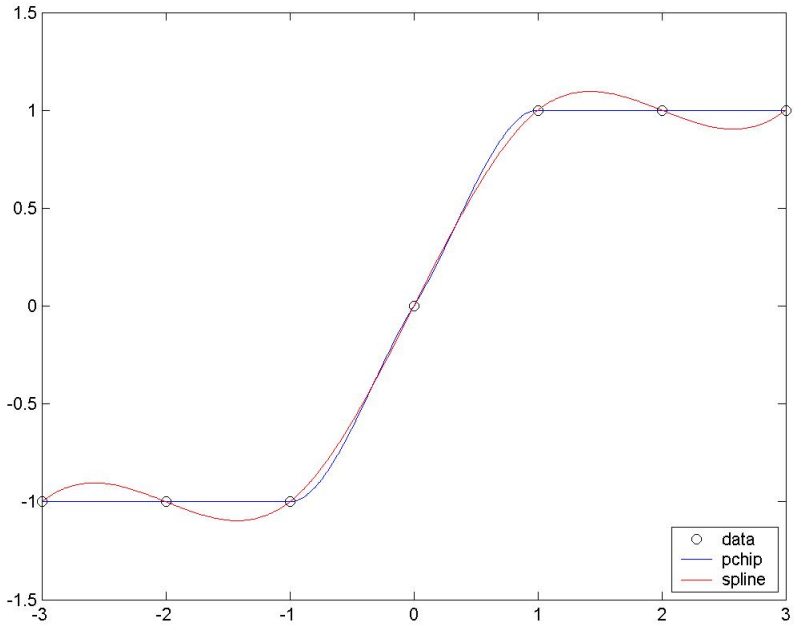

Figure 1: Figure for the last example. Note that the spline interpolant "overshoots" the data, while the "shape preserving" interpolant does not.