Programs for Natural Cubic Spline Interpolation

Mike Renfro

November 2, 2004

Programs for Natural Cubic Spline Interpolation

Part I

Programs for Natural Cubic Spline Interpolation

Review of Natural Cubic Spline Method

Given a series of points $(x_0, f(x_0)) \cdots (x_n, f(x_n))$, find a cubic equation for each of the n intervals that:

- passes through the endpoints $(x, f(x))$ at each end of the interval
- has continuous first derivatives from one interval to the next
- has continuous second derivatives from one interval to the next
- has second derivatives at x_0 and x_n of exactly 0.

Data from Example 5.12

```
% Example 5.12 data
x=[2 3 6.5 8 12];
f=[14 20 17 16 23];
n=length(x)-1;
```

Method 1, p.395–396

Equation 5.95 provides a system of $n-1$ equations describing the second derivative of the spline function at each knot:

$$f''(x_{i-1})(x_i - x_{i-1}) + 2f''(x_i)(x_{i+1} - x_{i-1}) + f''(x_{i+1})(x_{i+1} - x_i)$$
$$= 6\left\{\frac{f(x_{i+1}) - f(x_i)}{x_{i+1} - x_i} - \frac{f(x_i) - f(x_{i-1})}{x_i - x_{i-1}}\right\} \quad i = 1, 2, \cdots, n-1.$$

Since all the x and $f(x)$ values are known, the unknowns are the $f''(x)$ values. This is a linear algebraic system of equations. Also, remember that $f''(x_0) = 0$ and $f''(x_n) = 0$.

Method 1, Matrix Form

Solve $[A]\{y\} = \{p\}$, where

$$[A] = \begin{bmatrix} 2(x_2 - x_0) & x_2 - x_1 & 0 & \cdots & 0 \\ x_2 - x_1 & 2(x_3 - x_1) & x_3 - x_2 & \ddots & \vdots \\ 0 & x_3 - x_2 & 2(x_4 - x_2) & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & x_{n-1} - x_{n-2} \\ 0 & \cdots & 0 & x_{n-1} - x_{n-2} & 2(x_n - x_{n-2}) \end{bmatrix}$$

$$\{y\} = \left\{ \begin{array}{c} f''(x_1) \\ \vdots \\ f''(x_{n-1}) \end{array} \right\}$$

$$\{p\} = 6 \left\{ \begin{array}{c} \frac{f(x_2)-f(x_1)}{x_2-x_1} - \frac{f(x_1)-f(x_0)}{x_1-x_0} \\ \vdots \\ \frac{f(x_n)-f(x_{n-1})}{x_n-x_{n-1}} - \frac{f(x_{n-1})-f(x_{n-2})}{x_{n-1}-x_{n-2}} \end{array} \right\}$$

Method 1, Alternate Matrix Form

Solve $[A]\{y\} = \{p\}$, where

$$[A] = \begin{bmatrix} 2(h_1 + h_2) & h_2 & 0 & \cdots & 0 \\ h_2 & 2(h_2 + h_3) & h_3 & \ddots & \vdots \\ 0 & h_3 & 2(h_3 + h_4) & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & h_{n-1} \\ 0 & \cdots & 0 & h_{n-1} & 2(h_{n-1} + h_n) \end{bmatrix}$$

$$\{y\} = \left\{ \begin{matrix} f''(x_1) \\ \vdots \\ f''(x_{n-1}) \end{matrix} \right\}$$

$$\{p\} = 6 \left\{ \begin{matrix} \frac{f(x_2)-f(x_1)}{h_2} - \frac{f(x_1)-f(x_0)}{h_1} \\ \vdots \\ \frac{f(x_n)-f(x_{n-1})}{h_n} - \frac{f(x_{n-1})-f(x_{n-2})}{h_{n-1}} \end{matrix} \right\}, \quad h_i = x_i - x_{i-1}$$

Notice the Patterns in the Matrices

Notice that the first row of the $\{p\}$ vector uses the first, second, and third $f(x)$ values, as well as the first and second h values. The second row uses the second, third, and fourth $f(x)$ values, and also the second and third h values. We should be able to write a program to fill out all these matrices automatically.

Math-to-MATLAB Translation, Part 1

We have vectors of x and $f(x)$ values in MATLAB already. We can refer to any element of those vectors using a subscript in parentheses, for example:

```
>> x(1) % is the first element of the x vector
ans =
     2
>> x(2) % is the second element of the x vector
ans =
     3
>> f(end) % is the last element of the f vector
ans =
    23
```

Note that MATLAB does not have a zeroth element of a vector; all vector locations start at 1. So x_0 in math (the first x value) is the same as `x(1)` in MATLAB.

Math-to-MATLAB Translation, Part 2

MATLAB also allows us to perform vectorized calculations very quickly:

```
>> h=x(2:end)-x(1:end-1)
h =
    1.0000    3.5000    1.5000    4.0000
```

so now `h(1)` in MATLAB is the same as $h_1 = x_1 - x_0$ in math.

Filling Out the $\{p\}$ Vector

We can easily create a `for` loop in MATLAB to fill out the $\{p\}$ vector. The first element of $\{p\}$ is calculated using $f(x_0)$, $f(x_1)$, $f(x_2)$, h_1, and h_2. These correspond to `f(1)`, `f(2)`, `f(3)`, `h(1)`, and `h(2)` in MATLAB. For rows other than the first, the h and f positions are simply incremented.

MATLAB Code to Fill Out $\{p\}$ Vector

One version:

```
p=zeros(n-1,1);
for i=2:n
    p(i-1)=6*((f(i+1)-f(i))/h(i)-...
        (f(i)-f(i-1))/h(i-1));
end
```

Other variants are certainly possible — if `i` looped from `1` to `n-1`, you'd just have to shift the `i`, `i-1`, and `i+1` expressions up to `i+1`, `i`, and `i+2`, respectively.

Filling out the [A] Matrix

The [A] matrix is symmetric, and consists largely of zeros. Only the elements along the diagonal, and the elements immediately to the left or the right of the diagonal are filled.

The ith element on the diagonal is built up from h_i and h_{i+1}. The ith element immediately to the left or right of the diagonal are made up solely from h_{i+1}.

MATLAB Code to Fill Out $[A]$ Matrix

One version:

```
A=zeros(n-1);
for i=1:(n-1)
    A(i,i)=2*(h(i)+h(i+1));
end
for i=2:(n-1)
    A(i-1,i)=h(i);
    A(i,i-1)=h(i);
end
```

Elements of matrices in MATLAB are referenced similarly to elements of vectors. Matrix elements can be selected by using two subscripts in parentheses; the first subscript corresponds to the row of the matrix, and the second corresponds to the column.

Solving the System of Equations

```
ypp=A\p;
```

The values in ypp represent $f''(x_1) \cdots f''(x_n)$. Remember that the natural cubic spline end conditions require that $f(x_0)$ and $f(x_n)$ be exactly 0. To complete the list of $f''(x)$ values, let's add zeros to the beginning and end of the ypp variable:

```
ypp=[0; ypp; 0]
```

Results for Example 5.12, Part 1

```
>> cubicspline
ypp =
         0
   -5.2707
    1.7981
    1.0730
         0
```

The above $f''(x_i)$ values can be substituted into Equation 5.93

$$f_i(x) = f''(x_{i-1})\frac{(-x_i + x)^3}{6(-x_i + x_{i-1})} + f''(x_i)\frac{(x - x_{i-1})^3}{6(x_i - x_{i-1})} + \left\{\frac{f(x_{i-1})}{x_i - x_{i-1}} - f''(x_{i-1})\left(\frac{x_i - x_{i-1}}{6}\right)\right\}(x_i - x) + \left\{\frac{f(x_i)}{x_i - x_{i-1}} - f''(x_i)\left(\frac{x_i - x_{i-1}}{6}\right)\right\}(x - x_{i-1})$$

Results for Example 5.12, Part 2

Therefore,

$$\begin{aligned}
f_1(x) &= -0.878446(x-2)^3 + 14(3-x) + 20.878446(x-2) \\
f_2(x) &= 0.250985(x-6.5)^3 + 0.0856217(x-3)^3 \\
&\quad + 8.788847(6.5-x) + 3.808278(x-3) \\
f_3(x) &= -0.199784(x-8)^3 + 0.119223(x-6.5)^3 + 10.883822(8-x) \\
&\quad + 10.3984149(x-6.5) \\
f_4(x) &= -0.044709(x-12)^3 + 3.284659(12-x) + 5.75(x-8)
\end{aligned}$$

General Form of the Equations

For a given interval i, where $i = 1 \cdots n$ and $x_{i-1} \leq x \leq x_i$,

$$f_i(x) = a_i + b_i x + c_i x^2 + d_i x^3$$

This adds up to n cubic equations, with $4n$ unknowns. We need to find a total of $4n$ independent equations to solve for those unknowns.

First Set of Independent Equations

$2n$ equations can be derived from the continuity requirement:

$$
\begin{aligned}
(1)a_1 + (x_0)b_1 + (x_0^2)c_1 + (x_0^3)d_1 &= y_0 \\
(1)a_1 + (x_1)b_1 + (x_1^2)c_1 + (x_1^3)d_1 &= y_1 \\
(1)a_2 + (x_1)b_2 + (x_1^2)c_2 + (x_1^3)d_2 &= y_1 \\
(1)a_2 + (x_2)b_2 + (x_2^2)c_2 + (x_2^3)d_2 &= y_2 \\
(1)a_3 + (x_2)b_3 + (x_2^2)c_3 + (x_2^3)d_3 &= y_2 \\
&\vdots \\
(1)a_n + (x_{n-1})b_n + (x_{n-1}^2)c_n + (x_{n-1}^3)d_n &= y_{n-1} \\
(1)a_n + (x_n)b_n + (x_n^2)c_n + (x_n^3)d_n &= y_n
\end{aligned}
$$

Second Set of Independent Equations

$n-1$ equations can be derived from the requirement of continuous first derivatives:

$$\begin{aligned}
(1)b_1 + (2x_1)c_1 + (3x_1^2)d_1 & \\
+(-1)b_2 + (-2x_1)c_2 + (-3x_1^2)d_2 &= 0 \\
(1)b_2 + (2x_2)c_2 + (3x_2^2)d_2 & \\
+(-1)b_3 + (-2x_2)c_3 + (-3x_2^2)d_3 &= 0 \\
&\vdots \\
(1)b_{n-1} + (2x_{n-1})c_{n-1} + (3x_{n-1}^2)d_{n-1} & \\
+(-1)b_n + (-2x_{n-1})c_n + (-3x_{n-1}^2)d_n &= 0
\end{aligned}$$

Third Set of Independent Equations

$n-1$ equations can be derived from the requirement of continuous second derivatives:

$$\begin{aligned}
(2)c_1 + (6x_1)d_1 + (-2)c_2 + (6x_1)d_2 &= 0 \\
(2)c_2 + (6x_2)d_2 + (-2)c_3 + (6x_2)d_3 &= 0 \\
&\vdots \\
(2)c_{n-1} + (6x_{n-1})d_{n-1} + (-2)c_n + (6x_{n-1})d_n &= 0
\end{aligned}$$

Fourth Set of Independent Equations

2 equations can be derived from the natural cubic spline end conditions ($f''(x_0) = f''(x_n) = 0$):

$$(2)c_1 + (6x_0)d_1 = 0$$
$$(2)c_n + (6x_n)d_n = 0$$

Notice the Patterns in the Equations and Matrices

If we assume that our vector of unknowns is

$$\left\{ \begin{array}{c} a_1 \\ b_1 \\ c_1 \\ d_1 \\ \vdots \\ a_n \\ b_n \\ c_n \\ d_n \end{array} \right\}$$

we can establish a few patterns in the equations and the matrix version.

Patterns from First Set of Equations

$$\begin{bmatrix} 1 & x_0 & x_0^2 & x_0^3 & 0 & \cdots & & & & 0 \\ 1 & x_1 & x_1^2 & x_1^3 & 0 & \cdots & & & & 0 \\ 0 & 0 & 0 & 0 & 1 & x_1 & x_1^2 & x_1^3 & 0 & \cdots \\ 0 & 0 & 0 & 0 & 1 & x_2 & x_2^2 & x_2^3 & 0 & \cdots \\ \vdots & & & & & & & & & \end{bmatrix} \begin{Bmatrix} a_1 \\ b_1 \\ c_1 \\ d_1 \\ \vdots \end{Bmatrix} = \begin{Bmatrix} y_0 \\ y_1 \\ y_1 \\ y_2 \\ \vdots \end{Bmatrix}$$

On the first and second row, column 1 has a 1 in it. On the third and fourth row, column 5 has a 1 in it. On the fifth and sixth row, the 1 is in column 9, etc.

How to Insert the 1s into the Coefficient Matrix

```
A=zeros(4*n);
row=1;
for col=1:4:4*n
    A(row,col)=1;
    A(row+1,col)=1;
    row=row+2;
end
```

This code sets the `col` variable to values of $1, 5, 9, \cdots$ up to a limit not exceeding $4n$. When `col` is 1, we insert a 1 into the $(1,1)$ element of A, and then insert a 1 into the $(2,1)$ element of A. At the end of the loop, we increment the `row` counter by 2 and continue. This loop automatically terminates at row $2n$.

How to Insert the x Values into the Coefficient Matrix

```
row=1;
index=1;
for col=2:4:4*n
    A(row,col)=x(index);
    A(row,col+1)=x(index)^2;
    A(row,col+2)=x(index)^3;
    index=index+1;
    row=row+2;
end
```

This code takes care of the upper rows of the x, x^2, and x^3 values. Remember that the number x_0 in math is the same as `x(1)` in MATLAB. So on the first loop, we insert `x(1)`, `x(1)^2`, and `x(1)^3` into the first row, columns 2, 3, and 4. These values correspond to x_0, x_0^2, and x_0^3 from the original equation.

How to Insert the x Values into the Coefficient Matrix

```
row=2;
index=2;
for col=2:4:4*n
    A(row,col)=x(index);
    A(row,col+1)=x(index)^2;
    A(row,col+2)=x(index)^3;
    index=index+1;
    row=row+2;
end
```

This code takes care of the lower rows of the x, x^2, and x^3 values, and all the coefficients from the first $2n$ equations are inserted now.

Other Parts of the Coefficent Matrix

The first-derivative continuity requirement is satisfied with the following two loops. The 1 term is applied to the b_i variables stored in columns $2, 6, 10, \cdots$. The $2x$ term is is applied to the c_i variables in columns $3, 7, 11, \cdots$. The $3x^2$ term is applied to the d_i variables in columns $4, 8, 12, \cdots$.

```
row=2*n+1;
index=2;
for col=2:4:4*n-4
    A(row,col)=1;
    A(row,col+1)=2*x(index);
    A(row,col+2)=3*x(index)^2;
    index=index+1;
    row=row+1;
end
```

Other Parts of the Coefficient Matrix

First-derivative continuity requirement (continued). The -1 term is on the b_i variables, the $-2x$ term on the c_i variables, and the $-3x^2$ term on the d_i variables.

```
row=2*n+1;
index=2;
for col=6:4:4*n
    A(row,col)=-1;
    A(row,col+1)=-2*x(index);
    A(row,col+2)=-3*x(index)^2;
    index=index+1;
    row=row+1;
end
```

Other Parts of the Coefficient Matrix

Second-derivative continuity requirement. The 2 term is applied to the c_i variables, and the $6x$ term is applied to the d_i variables.

```
row=3*n;
index=2;
for col=3:4:4*n-4
    A(row,col)=2;
    A(row,col+1)=6*x(index);
    index=index+1;
    row=row+1;
end
```

Other Parts of the Coefficient Matrix

Second-derivative continuity requirement (continued). The -2 term is applied to the c_i variables, and the $-6x$ term is applied to the d_i variables.

```
row=3*n;
index=2;
for col=7:4:4*n
    A(row,col)=-2;
    A(row,col+1)=-6*x(index);
    index=index+1;
    row=row+1;
end
```

Other Parts of the Coefficient Matrix

The last two equations, the natural cubic spline end conditions, applied to the c_1, d_1, c_n, and d_n variables:

```
A(4*n-1,3)=2;
A(4*n-1,4)=6*x(1);
A(4*n,4*n-1)=2;
A(4*n,4*n)=6*x(n+1);
```

The Right-Hand Side Vector

The following code fills out the right-hand side of the matrix equation. The first $2n$ values are $y_0, y_1, y_1, y_2, y_2, \cdots, y_{n-1}, y_{n-1}, y_n$, and the remaining values are all 0.

```
knowns=zeros(4*n,1);
knowns(1)=f(1);
for i=2:2:(2*n-2)
    knowns(i)=f(i/2+1);
    knowns(i+1)=f(i/2+1);
end
knowns(2*n)=f(end)
```

The [A] Matrix

$$[A]=\begin{bmatrix}
1 & 2 & 4 & 8 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
1 & 3 & 9 & 27 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 1 & 3 & 9 & 27 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 1 & 6\frac{1}{2} & 42\frac{1}{4} & 274\frac{5}{8} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 6\frac{1}{2} & 42\frac{1}{4} & 274\frac{5}{8} & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 8 & 64 & 512 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 8 & 64 & 512\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 12 & 144 & 1728\\
0 & 1 & 6 & 27 & 0 & -1 & -6 & -27 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 1 & 13 & 126\frac{3}{4} & 0 & -1 & -13 & -126\frac{3}{4} & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 16 & 192 & 0 & -1 & -16 & -192\\
0 & 0 & 2 & 18 & 0 & 0 & -2 & -18 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 2 & 39 & 0 & 0 & -2 & -39 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 48 & 0 & 0 & -2 & -48\\
0 & 0 & 2 & 12 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 72
\end{bmatrix}$$

The $\{p\}$ Vector

$$\{p\} = \begin{Bmatrix} 14 \\ 20 \\ 20 \\ 17 \\ 17 \\ 16 \\ 16 \\ 23 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{Bmatrix}$$

The Cubic Coefficients

```
>> coefficients=A\knowns
coefficients =
    7.2707
   -3.6629
    5.2707
   -0.8784
  -25.5358
   29.1435
   -5.6648
    0.3366
   89.0302
  -23.7331
    2.4701
   -0.0806
   70.6713
  -16.8485
    1.6095
   -0.0447
```

These coefficients are in the order a_1, b_1, c_1, d_1, $\cdots$, a_n, b_n, c_n, d_n.

Homework

Continue with all problems assigned from Chapter 5. Have them all ready to turn in on Thursday, November 4.