Splines

Text Reference: Section 1.2, p. 23

Note: The MATLAB M-file **splinedat.m** *contains data for the examples and the questions. Save the accompanying file* **splinedat.m** *as a text file and with the .m extension. Set the path browser in MATLAB to the folder where the file* **splinedat.m** *is located. Then type* **splinedat** *in the MATLAB command window to access the data.*

The purpose of this set of exercises is to show how to use a system of linear equations to fit a piecewise-polynomial curve through a set of points.

Consider the problem of fitting a curve $y = f(t)$ to a given set of data points (t_1, y_1), (t_2, y_2), … , (t_n, y_n). In another project it is shown that a single polynomial function which passes through each of these points may be found, but sometimes this approach is unwise given the conditions of the problem. There is another way to proceed which still results in a curve passing through all the data points. Consider taking each pair of consecutive data points and fitting a polynomial curve through them. This process creates what is sometimes called a "piecewise-polynomial" curve, but more often is called a **spline**.

Example: The following data from Car and Driver magazine[1] shows the elapsed time it took a Honda CR-V starting at rest to accelerate to 30, 60, and 90 m.p.h.

Honda CR-VEX	Time	0	3.1	10.3	30.1
	Velocity	0	30	60	90

To approximate how long it would take the CR-V to accelerate to 50 m.p.h. or to approximate the distance it would take the CR-V to accelerate to 90 m.p.h., an explicit velocity function $v(t)$ is needed. Such a $v(t)$ could be found by fitting a piecewise-polynomial curve to the data. The easiest approach would be to fit lines between each consecutive pair of data points; the result is Figure 1.

[1] *Car and Driver*, May 1998, p. 102

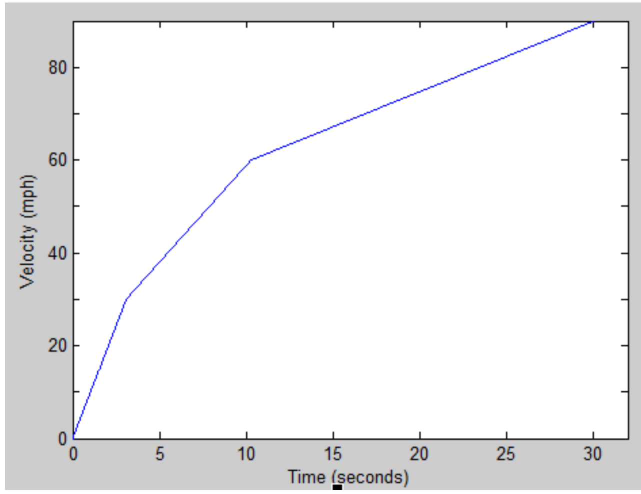

Figure 1: Piecewise-Linear Fit

The drawback to this approach is that the curve is not smooth; that is, its slope changes abruptly at the data points. The expectation is that the velocity function should indeed be smoother than that produced by the line fitting. In order to ensure that the velocity function $v(t)$ is as smooth as needed, assume that $v'(t)$ and $v''(t)$ are continuous functions. In order to make these assumptions feasible, fit a third-degree polynomial to each consecutive pair of data points. This piecewise-polynomial curve is called a **cubic spline**.

To fit a cubic spline $v(t)$ to the velocity data, assume that on each of the three intervals [0,3.1], [3.1,10.3], and [10.3,30.1] the formula for $v(t)$ is given by a cubic polynomial whose coefficients must be determined. It is convenient to write the formulas as follows:

$$v(t) = \begin{cases} a_1(t-0)^3 + a_2(t-0)^2 + a_3(t-0) + a_4 & \text{if } 0 \le t \le 3.1 \\ b_1(t-3.1)^3 + b_2(t-3.1)^2 + b_3(t-3.1) + b_4 & \text{if } 3.1 \le t \le 10.3 \\ c_1(t-10.3)^3 + c_2(t-10.3)^2 + c_3(t-10.3) + c_4 & \text{if } 10.3 \le t \le 30.1 \end{cases}$$

Since $v(0)=0$, $v(3.1)=30$, $v(10.3)=60$, and $v(30.1)=90$,

$$a_4 = 0 \tag{1}$$

$$(3.1)^3 a_1 + (3.1)^2 a_2 + 3.1\, a_3 + a_4 = 30 \tag{2}$$

$$b_4 = 30 \tag{3}$$

$$(7.2)^3 b_1 + (7.2)^2 b_2 + 7.2\, b_3 + b_4 = 60 \tag{4}$$

$$c_4 = 60 \tag{5}$$

$$(19.8)^3 c_1 + (19.8)^2 c_2 + 19.8\, c_3 + c_4 = 90 \tag{6}$$

Consider the derivative $v'(t)$:

$$v'(t) = \begin{cases} 3a_1(t-0)^2 + 2a_2(t-0) + a_3 & \text{if } 0 \le t \le 3.1 \\ 3b_1(t-3.1)^2 + 2b_2(t-3.1) + b_3 & \text{if } 3.1 \le t \le 10.3 \\ 3c_1(t-10.3)^2 + 2c_2(t-10.3) + c_3 & \text{if } 10.3 \le t \le 30.1 \end{cases}$$

Since $v'(t)$ is supposed to be continuous at $t = 3.1$ and $t = 10.3$, it must be true that

$$3(3.1)^2 a_1 + 2(3.1)a_2 + a_3 = b_3$$
$$3(7.2)^2 b_1 + 2(7.2)b_2 + b_3 = c_3$$

which may be rewritten as

$$3(3.1)^2 a_1 + 2(3.1)a_2 + a_3 - b_3 = 0 \qquad (7)$$
$$3(7.2)^2 b_1 + 2(7.2)b_2 + b_3 - c_3 = 0 \qquad (8)$$

Further consider the second derivative $v''(t)$:

$$v''(t) = \begin{cases} 6a_1(t-0) + 2a_2 & \text{if } 0 \le t \le 3.1 \\ 6b_1(t-3.1) + 2b_2 & \text{if } 3.1 \le t \le 10.3 \\ 6c_1(t-10.3) + 2c_2 & \text{if } 10.3 \le t \le 30.1 \end{cases}$$

To make $v''(t)$ continuous at $t = 3.1$ and $t = 10.3$, set

$$6(3.1)a_1 + 2a_2 - 2b_2 = 0 \qquad (9)$$
$$6(7.2)b_1 + 2b_2 - 2c_2 = 0 \qquad (10)$$

And so there are 10 linear equations relating the 12 variables. Two more equations are needed to hope for a unique solution, and there are several ways to do this. One way is to choose to assume that $v''(0) = v''(30.1) = 0$; these assumptions give the final two equations:

$$2a_2 = 0 \qquad (11)$$
$$6(19.8)c_1 + 2c_2 = 0 \qquad (12)$$

The augmented matrix A for this system of equations is

$$
\begin{pmatrix}
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
29.79 & 9.61 & 3.1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 30 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 30 \\
0 & 0 & 0 & 0 & 373.248 & 51.84 & 7.2 & 1 & 0 & 0 & 0 & 0 & 60 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 60 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 7762.392 & 392.04 & 19.8 & 1 & 90 \\
28.83 & 6.2 & 1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 155.52 & 14.4 & 1 & 0 & 0 & 0 & -1 & 0 & 0 \\
18.6 & 2 & 0 & 0 & 0 & -2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 43.2 & 2 & 0 & 0 & 0 & -2 & 0 & 0 & 0 \\
0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 118.8 & 2 & 0 & 0 & 0
\end{pmatrix}
$$

where the columns correspond to a_1, a_2, a_3, a_4, b_1, b_2, b_3, b_4, c_1, c_2, c_3, and c_4 .

Here is the matrix *A* in MATLAB:

```
A= [0 0   0   1   0   0   0   0   0   0   0   0   0 ;
    29.791    9.61  3.1    1  0   0   0   0   0   0   0   0   30;
    0   0   0   0   0   0   0   1   0   0   0   0   30 ;
    0   0   0   0   373.248   51.84 7.2    1  0   0   0   0   60 ;
    0   0   0   0   0   0   0   0   0   0   0   1   60 ;
    0   0   0   0   0   0   0   0   7762.392 392.04    19.8   1   90 ;
    28.83 6.2    1   0   0   0   -1 0   0   0   0   0   0 ;
    0   0   0   0   155.52    14.4   1   0   0   0   -1 0   0;
    18.6   2   0   0   0   -2 0   0   0   0   0   0   0;
    0   0   0   0   43.2   2   0   0   0   -2 0   0   0;
    0   2   0   0   0   0   0   0   0   0   0   0   0;
    0   0   0   0   0   0   0   0   118.8 2   0   0   0]
```

If the accompanying M-file *splinedat.m* is available to MATLAB's working path, you can type **`splinedat`** and then **`A`** to get the matrix *A*.

Row reducing A produces the matrix

```
» B=rref(A)

B =

  Columns 1 through 7

    1.0000         0         0         0         0         0         0
         0    1.0000         0         0         0         0         0
         0         0    1.0000         0         0         0         0
         0         0         0    1.0000         0         0         0
         0         0         0         0    1.0000         0         0
         0         0         0         0         0    1.0000         0
         0         0         0         0         0         0    1.0000
         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         0         0
         0         0         0         0         0         0         0

  Columns 8 through 13

         0         0         0         0         0   -0.0847
         0         0         0         0         0    0.0000
         0         0         0         0         0   10.4914
         0         0         0         0         0         0
         0         0         0         0         0    0.0345
         0         0         0         0         0   -0.7878
         0         0         0         0         0    8.0494
    1.0000         0         0         0         0   30.0000
         0    1.0000         0         0         0    0.0007
         0         0    1.0000         0         0   -0.0423
         0         0         0    1.0000         0    2.0731
         0         0         0         0    1.0000   60.0000
```

so the unique solution of the system is found to be

$$\begin{aligned} &a_1 = -0.0847, a_2 = 0,\ a_3 = 10.4914,\ a_4 = 0 \\ &b_1 = 0.0345,\ b_2 = -0.7878,\ b_3 = 8.0494,\ b_4 = 30 \\ &c_1 = 0.0007,\ c_2 = -0.0423,\ c_3 = 2.0731,\ c_4 = 60 \end{aligned}$$

Thus the velocity function is

$$v(t) = \begin{cases} \text{-0.0847 } (t-0)^3 + 10.4914(t-0) & \mathit{if}\ 0 \le t \le 3.1 \\ 0.0345(t-3.1)^3 + -0.7878(t-3.1)^2 + 8.0494(t-3.1) + 30 & \mathit{if}\ 3.1 \le t \le 10.3 \\ 0.0007(t-10.3)^3 + -0.0423(t-10.3)^2 + 2.0731(t-10.3) + 60 & \mathit{if}\ 10.3 \le t \le 30.1 \end{cases}$$

The function $v(t)$ may be input into MATLAB using the Symbolic Toolbox. First we will gather the coefficients for the piecewise functions:

```
s1=B([1 2 3 4],13); s2=B([5 6 7 8],13);
s3 = B([9 10 11 12],13);
```

Next we will convert these functions to symbolic polynomials using the MATLAB Symbolic Toolbox:

```
s1=poly2sym(s1); s2=poly2sym(s2); s3=poly2sym(s3);
```

However, the latter two polynomials need to be translated:

```
syms x; p1=s1; p2=subs(s2, x-3.1); p3=subs(s3, x-
10.3);
```

We can then return these symbolic polynomials back to coefficient polynomials:

```
r1=sym2poly(p1); r2=sym2poly(p2); r3=sym2poly(p3);
```

Then graph the piecewise function

```
t= [0 3.1 10.3 30.1]; vel=[0 30 60 90];
xx1=0:.1:3.1; v1=polyval(r1,xx1);
xx2=3.1:.1:10.3; v2=polyval(r2,xx2);
xx3=10.3:.1:30.1; v3=polyval(r3,xx3);
hold on
plot(t,vel,'o',xx1,v1,xx2,v2,xx3,v3)
```

The functions `p1, p2, p3` are symbolic objects whereas `r1, r2,` and `r3` are numerical objects. Checking the numerical coefficients:

```
>>r1, r2, r3
r1 =

   -0.0847    0.0000   10.4914         0

r2 =

    0.0345   -1.1087   13.9285   -3.5517

r3 =

    0.0007   -0.0643    3.1704   33.3845
```

We see that the function is

$$v(t) = \begin{cases} -0.0847\,t^3 + 10.4914\,t & \text{if } 0 \le t \le 3.1 \\ 0.0345\,t^3 + -1.1087\,t^2 + 13.9285\,t - 3.5517 & \text{if } 3.1 \le t \le 10.3 \\ 0.0007\,t^3 + -0.0643\,t^2 + 3.1704\,t + 33.3845 & \text{if } 10.3 \le t \le 30.1 \end{cases}$$

as given above.

A graph of $v(t)$ is available as Figure 2. Notice how smooth the graph appears compared with Figure 1. This $v(t)$ can be used to answer the questions posed earlier.

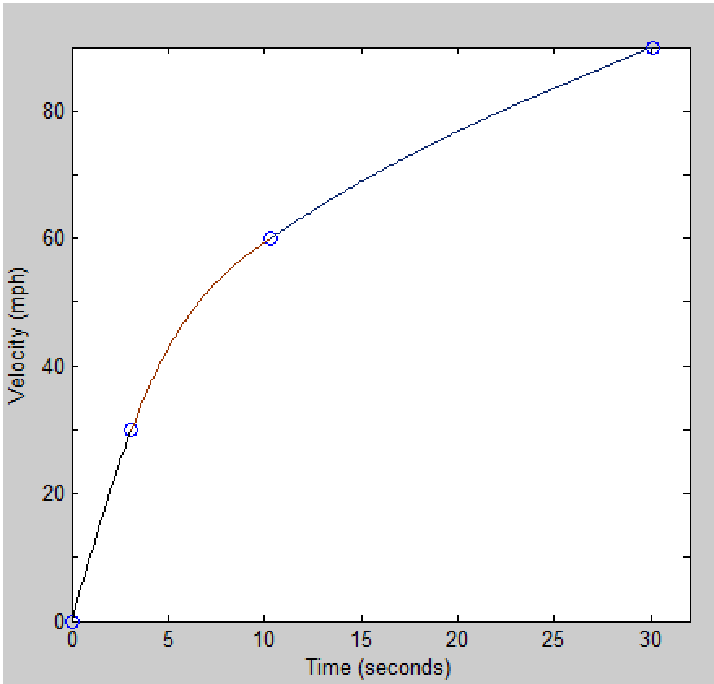

Figure 2: Cubic Spline Fit

1. How long will it take the CR-V to accelerate to 50 m.p.h.?

This event will happen some time between 3.1 and 10.3 seconds, so MATLAB can be used to solve the equation. We will find the roots for the symbolic function p2-50. We use the **`double`** command to find a numerical result:

```
» double(solve(p2-50))

    ans =

          6.5993
          12.7627 - 8.4988i
          12.7627 + 8.4988i
```

We take the one real solution obtaining t=6.5993 seconds.

2. How much distance will it take for the CR-V to accelerate to 90 m.p.h.?

First consider units of measure. Since $v(t)$ is measured in miles per hour and t is measured in seconds, $v(t)$ should be converted to miles per second before proceeding. The velocity function in miles per second is thus $v(t)/3600$. The distance traveled is

$$d = \int_0^{30.1} v(t)/3600dt$$

(Why?) This integral may be evaluated numerically using MATLAB, giving

```
>>f=int(p1/3600,0,3.1)+int(p2/3600,3.1,10.3)+int(p3/3600,10.3,30.1)
>>double(f)

            ans =

                0.5307
```

so the desired distance is .530727 miles.

Questions:

1. Using cubic splines, answer the above two questions for two of the sport utility vehicles in the Table below. Choose which two vehicles you want to study, and construct a cubic spline function for each vehicle. Write the coefficients to three significant figures as in the example above.

Jeep Cherokee SE	Time	0	3.2	12	38.2
	Velocity	0	30	60	90
Kia Sportage	Time	0	4.2	12.8	38.7
	Velocity	0	30	60	90
Subaru Forester L	Time	0	2.8	9.5	22.7
	Velocity	0	30	60	90
Toyota RAV4	Time	0	3	10.2	31.7
	Velocity	0	30	60	90

2. Which of the two vehicles you chose in Question 1 requires the longer distance to reach 90 m.p.h.?