Mathematical Modelling & Simulation (MC-409) Lab

Experiment 5 - Fitting a Cubic Spline Curve Using `spline()` in MATLAB

Delhi Technological University

Dr. Vineet Srivastava

11th October 2020

Anish Sachdeva
DTU/2K16/MC/013

Cubic Spline

Cubic spline interpolation is a special case for Spline interpolation that is used very often to avoid the problem of Runge's phenomenon. This method gives an interpolating polynomial that is smoother and has smaller error than some other interpolating polynomials such as Lagrange polynomial and Newton polynomial.

Definition

Given a set of $n + 1$ data points (x_i, y_i) where no 2 x_i are the same and $a = x_0 < x_1 < \cdots x_n = b$, the spline $S(x)$ is a function satisfying:

1. $S(x) \in C^2[a, b]$
2. On each subinterval $[x_{i-1}, x_i]$, $S(x)$ is a polynomial of degree 3, where $i = 1, 2, 3, \cdots n$
3. $S(x_i) = y_i$, for all $i = 0, 1, 2, 3, \cdots n$

Let us assume that:

$$S(x) = \begin{cases} C_1(x), & x_0 \le x \le x_1 \\ \cdots & \\ C_i(x), & x_{i-1} < x \le x_i \\ \cdots & \\ C_n(x), & x_{n-1} < x \le x_n \end{cases}$$

Where each $C_i = a_i + b_i x + c_i x^2 + d_i x^3 \ (d_i \neq 0)$ is a cubic function for $i = 0, 1, 2, 3, \cdots n$.

Boundary Conditions

To determine the cubic spline $S(x)$, we need to determine a_i, b_i, c_i and d_i for each i by:

1. $C_i(x_{i-1}) = y_{i-1}$ and $C_i(x_i) = y_i \ \ i = 1, \cdots n$
2. $C'_i(x_i) = C'_{i+1}(x_i) \ \ i = 1, \cdots n$
3. $C''_i(x_i) = C''_{i+1}(x_i) \ \ i = 1 \cdots n$

We can see that there are $n + n + (n - 1) = 4n - 2$ conditions, but we need to determine $4n$ coefficients, so usually we add two boundary conditions to solve this problem. There are three types of common boundary conditions:

1. First Derivatives at the endpoints are known:

   $C'_1(x_0) = f'_0$ and $C'_n(x_n) = f'_n$

This is called clamped boundary conditions:

2. Second Derivative at the endpoints are known:

   $C''_1(x_0) = f''_0$ and $C'_n(x_n) = f'_n$.

   The special case $C''_1(x_0) = C''(x_n) = 0$ is called natural or simple boundary conditions.

3. When the exact function $f(x)$ is a periodic function with period $x_n - x_0$, $S(x)$ is a periodic function with period $x_n - x_0$ too. Thus:

   $C_1(x_0) = C_n(x_n),\ C'_1(x_0) = C'_n(x_n),$ and $C''_1(x_0) = C''_n(x_n)$.

   The spline functions $S(x)$ satisfying this type of boundary conditions are called periodic splines.

Method

There are several methods that can be used to find the spline function *S(x)* according to its corresponding conditions. Since there are $4n$ coefficients to determine with *4n* conditions, we can easily plug the values we know into the $4n$ conditions and then solve the system of equations. Note that all the equations are linear with respect to the coefficients, so this is workable and computers can do it quite well.

1. For type 1 boundary conditions, we are given $C'_1(x_0) = f'_0$ and $C'_n(x_0) = f'_n$. According to equations we can obtain:

   $2M_0 + M_1 = \frac{6}{h_n}(f'_n - f[x_{n-1},\ x_n]) = 6f[x_{n-1},\ x_n,\ x_n]$

   Therefore let $\lambda_0 = \mu_n = 1$, $d_0 = 6f[x_{n-1},\ x_n,\ x_n]$. So the system of equations we need to solve is:

$$\begin{bmatrix} 2 & \lambda_0 & & & & & \\ \mu_1 & 2 & \lambda_1 & & & & \\ & \ddots & \ddots & \ddots & & & \\ & & \ddots & \ddots & \ddots & & \\ & & & \ddots & \ddots & \ddots & \\ & & & & \mu_{n-1} & 2 & \lambda_{n-1} \\ & & & & & \mu_n & 2 \end{bmatrix} \begin{bmatrix} M_0 \\ M_1 \\ \vdots \\ \vdots \\ \vdots \\ M_{n-1} \\ M_n \end{bmatrix} = \begin{bmatrix} d_0 \\ d_1 \\ \vdots \\ \vdots \\ \vdots \\ d_{n-1} \\ d_n \end{bmatrix}.$$

2. For type 2 Boundary Condition we are given

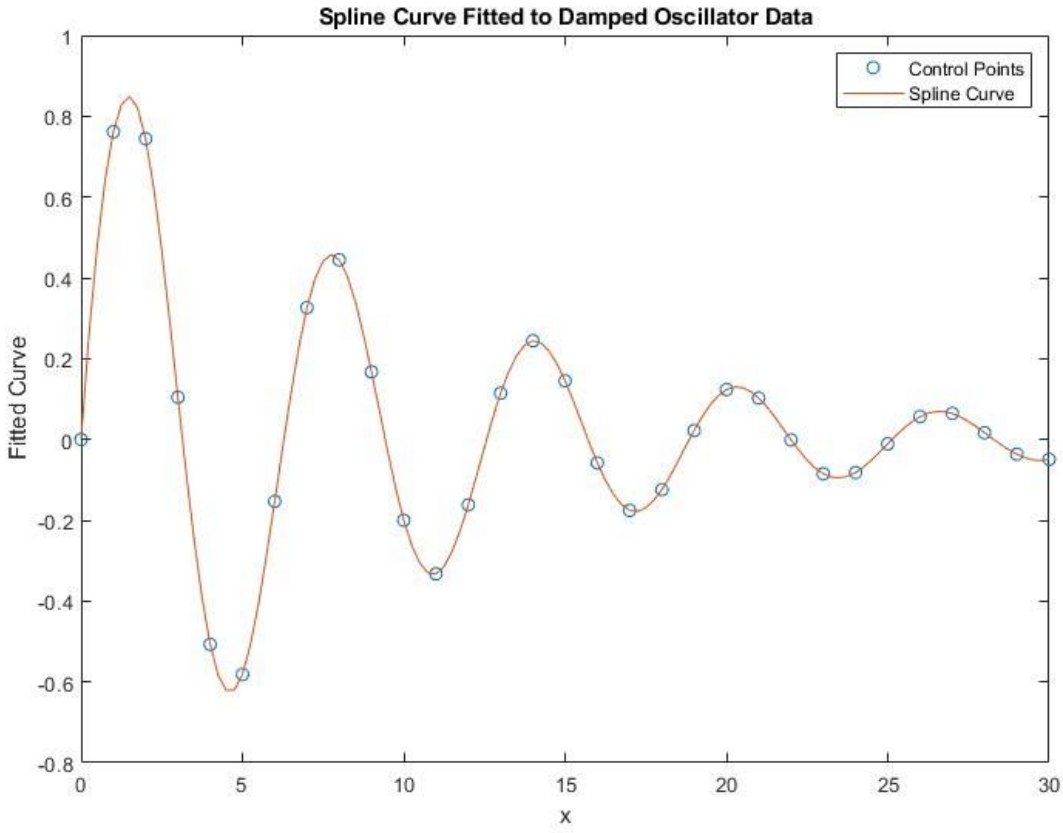
   $M_0 = f''_0$ and $M_n = f''_n$

Directly, so let $\lambda_0 = \mu_0 = 0$, $d_0 = 2f''_0$, and $d_n = 2f''_n$, and we need to solve this system of equations the same as above.

Code

```
% Spline Curve Fitting on Damped Oscillator

% Clearing all variables and Plots
clc;
clear;
close all;

% Creating Control Points
x = 0:30;
a = .1;
y = exp(-a * x) .* sin(x);

% Generating Data points to map spline function
xx = 0:.25:30;
yy = spline(x, y, xx);

% Plotting Spline Function
p1 = plot(x, y, 'o');
hold on;
p2 = plot(xx, yy);
legend([p1, p2], 'Control Points', 'Spline Curve');
title("Spline Curve Fitted to Damped Oscillator Data");
xlabel('x');
ylabel("Fitted Curve");
```

Output


Spline Curve Fitted to Damped Oscillator Data
Control Points
Spline Curve
Fitted Curve
x
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
0
5
10
15
20
25
30