8 Feb 2024

More about splines.

Image of a spline used in technical drawing

Splines of degree d through n points (or "knots") have $n(d+1)$ conditions, that is, we have n polynomials $\{p_1, p_2, \ldots, p_n\}$ passing through $n+1$ points (x_i, y_i) with the following restrictions:

- The polynomials must be continuous at the *2n* conditions given by the given points. That is
  $p_i(x_{i-1}) = y_{i-1}, \qquad p_i(x_i) = y_i \qquad i = 1, 2, \ldots, n$
- The derivatives up to $(d-1)^{st}$ order at each of the interior knots must agree, that is,

$$p_i^{(k)}\left(x_i\right)=p_{i+1}^{(k)}\left(x_i\right) \quad i=1,2,\ldots,n-1 \quad k=1,2,\ldots,d-1$$

$$n(d+1)$$

$$n$$

$$\{p_1, p_2, p_n, \ldots\}$$

$$n$$

$$x_i, y_i$$

$$p_i(x_{i-1})=y_{i-1}, false, 2,\ldots, n$$

$$(d-1)^{st}$$

$$false, 2,\ldots, n-k=1,2,\ldots, d-1 \tag{1}$$

Since n polynomials of degree d have n(d+1) coefficients and

$$n\cdot(d+1)-(2\cdot n+(n-1)\cdot(d-1))=d-1$$

- this leaves d-1 conditions to determine.

These are "end conditions", ie, specified by the derivatives at the two endpoints. There are several variations, two most common are

- A natural spline, which sets the the highest order derivatives to 0 (half of them can be specified) [this is the default]
- A periodic spline, forcing the derivatives at the first knot to agree with the derivatives at the final one.

$$n\ (d+1)-2\ n-(n-1)\ (d-1)=d-1 \tag{2}$$

I can either use **with(CurveFitting)** to load all the commands in the CurveFitting package, or refer to its "full name" using **CurveFitting:-Spline** or I can use **with(CurveFitting, Spline)** to just get that one.
For variety, let's just load the two we care about here.

```
> with(CurveFitting,Spline);
  with(CurveFitting,PolynomialInterpolation);
```

$$[Spline]$$

$$[PolynomialInterpolation] \tag{3}$$

Let's define some data, and play with it.

```
> knots:=[[0,1], [1,2], [2,0], [3,1], [4,0], [5,1]];
```

$$knots := [[0,1],[1,2],[2,0],[3,1],[4,0],[5,1]] \tag{4}$$

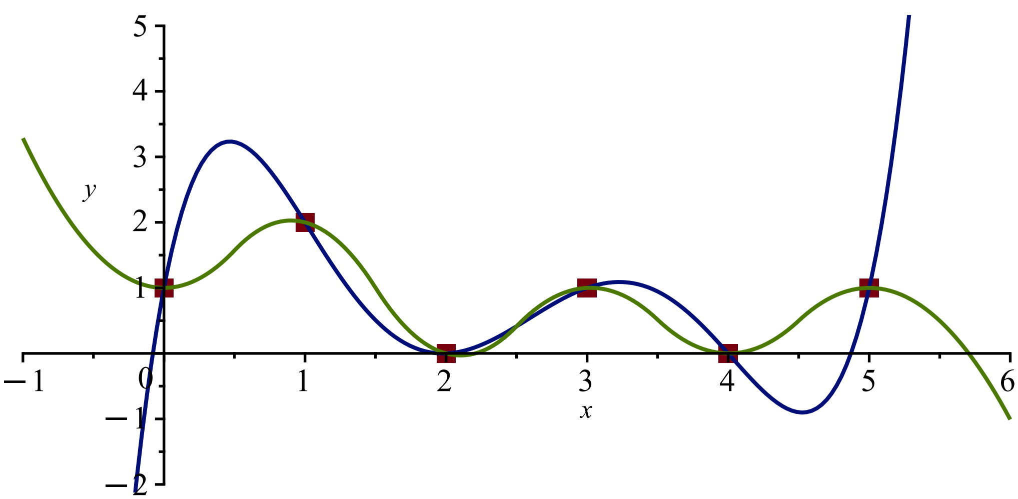
```
> polly:=PolynomialInterpolation(knots,x);
```

$$polly := \frac{1}{6}x^5-\frac{17}{8}x^4+\frac{115}{12}x^3-\frac{143}{8}x^2+\frac{45}{4}x+1 \tag{5}$$

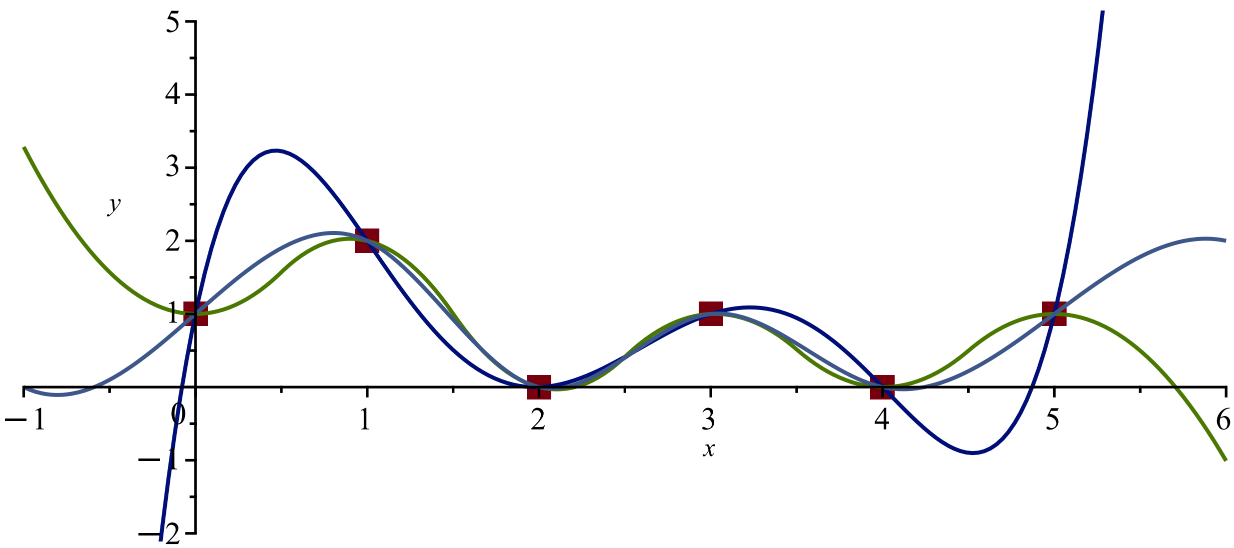
```
> sp2:=Spline(knots,x,degree=2)
```

$$sp2 := \begin{cases} \frac{2716\,x^2}{1189} + 1 & x < \frac{1}{2} \\ -\frac{3392}{1189}\,x^2 + \frac{6108}{1189}\,x - \frac{338}{1189} & x < \frac{3}{2} \\ \frac{3368}{1189}\,x^2 - \frac{14172}{1189}\,x + \frac{14872}{1189} & x < \frac{5}{2} \\ -\frac{2548}{1189}\,x^2 + \frac{15408}{1189}\,x - \frac{22103}{1189} & x < \frac{7}{2} \\ \frac{2408}{1189}\,x^2 - \frac{19284}{1189}\,x + \frac{38608}{1189} & x < \frac{9}{2} \\ -\frac{2388}{1189}\,x^2 + \frac{23880}{1189}\,x - \frac{58511}{1189} & \mathit{otherwise} \end{cases} \tag{6}$$

```
> plot([knots,polly,sp2], x=-1..6,y=-2..5,
  style=[point,line,line],symbolsize=20,symbol=solidbox, size=[.8,
  .4]);
```




```
> sp3:=Spline(knots,x,degree=3):
> plot([knots,polly,sp2,sp3], x=-1..6,y=-2..5,
  style=[point,line,line,line],symbolsize=20,
  symbol=solidbox, size=[.8,.4]);
```




```
> dsp3:=diff(sp3,x)
```

$$dsp3 := \begin{cases} -\frac{669\,x^2}{209} + \frac{432}{209} & x \le 1 \\ \frac{1464}{209}\,x^2 - \frac{4266}{209}\,x + \frac{135}{11} & x \le 2 \\ -\frac{75}{11}\,x^2 + \frac{7290}{209}\,x - \frac{8991}{209} & x \le 3 \\ \frac{1101}{209}\,x^2 - \frac{414}{11}\,x + \frac{13743}{209} & x \le 4 \\ -\frac{471}{209}\,x^2 + \frac{4710}{209}\,x - \frac{11409}{209} & 4 < x \end{cases} \tag{7}$$

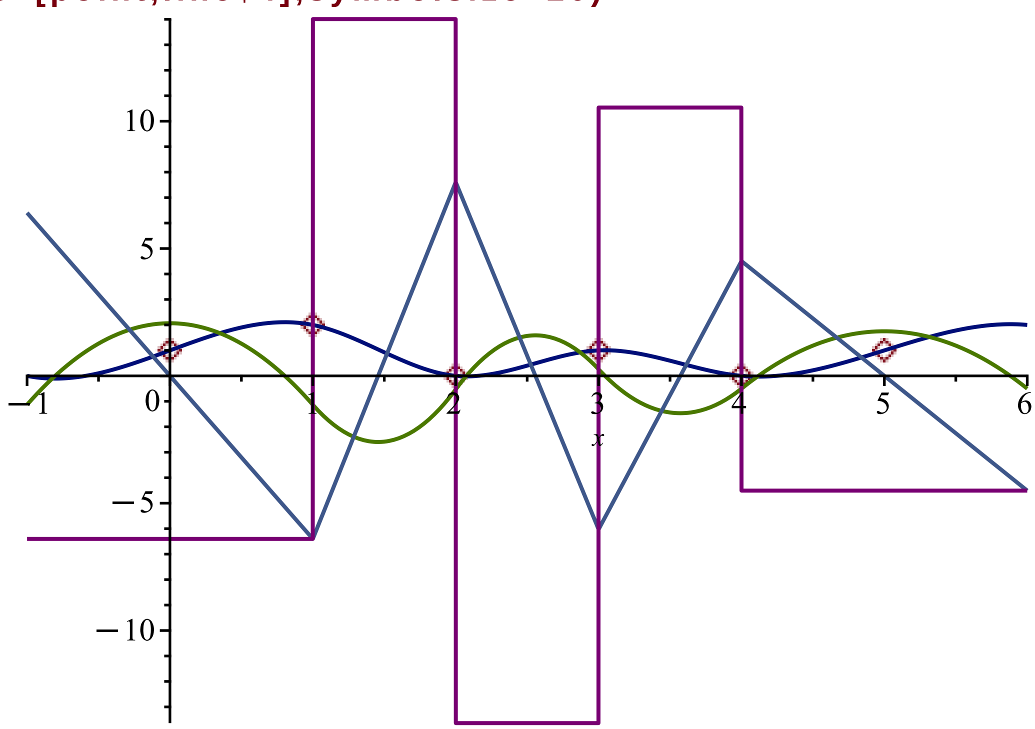
```
> ddsp3:=diff(%,x)
```

$$ddsp3 := \begin{cases} -\frac{1338\,x}{209} & x \le 1 \\ \frac{2928\,x}{209} - \frac{4266}{209} & x \le 2 \\ -\frac{150\,x}{11} + \frac{7290}{209} & x \le 3 \\ \frac{2202\,x}{209} - \frac{414}{11} & x \le 4 \\ -\frac{942\,x}{209} + \frac{4710}{209} & 4 < x \end{cases} \tag{8}$$

```
> dddsp3:=diff(ddsp3,x)
```

$$dddsp3 := \begin{cases} -\frac{1338}{209} & x < 1 \\ \mathit{undefined} & x = 1 \\ \frac{2928}{209} & x < 2 \\ \mathit{undefined} & x = 2 \\ -\frac{150}{11} & x < 3 \\ \mathit{undefined} & x = 3 \\ \frac{2202}{209} & x < 4 \\ \mathit{undefined} & x = 4 \\ -\frac{942}{209} & 4 < x \end{cases} \tag{9}$$

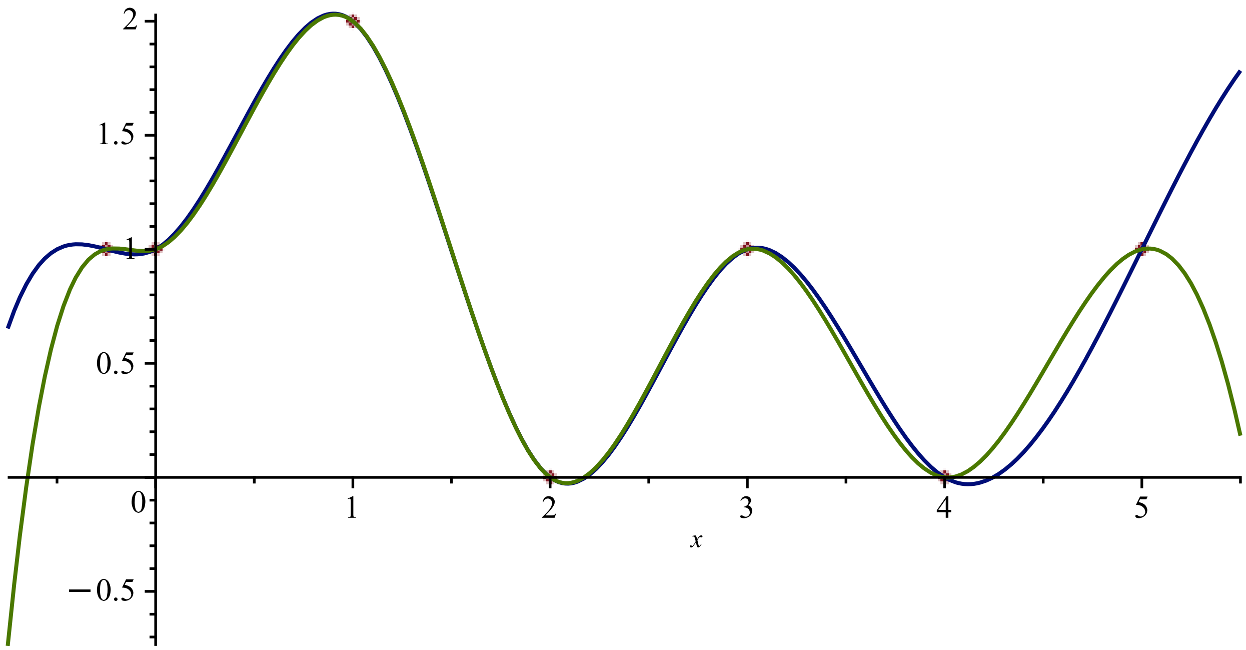
```
> plot([knots,sp3,dsp3,ddsp3,dddsp3], x=-1..6,
  style=[point,line$4],symbolsize=20)
```



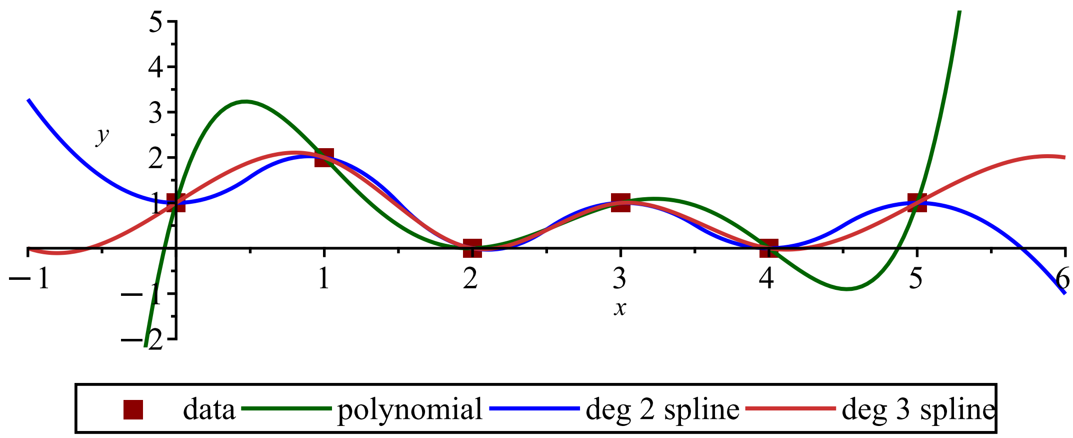

```
> ?spline
> moreknots:=[[-1/4,1],op(knots)];
```

$$moreknots := \left[\left[-\frac{1}{4}, 1\right], [0, 1], [1, 2], [2, 0], [3, 1], [4, 0], [5, 1]\right] \tag{10}$$

```
> nsp3:=Spline(moreknots,x,degree=3,endpoints=natural):
  psp3:=Spline(moreknots,x,degree=3,endpoints=periodic):
> plot([moreknots,  nsp3, psp3],
    x=-0.75..5.5,style=[point,line$3])
```

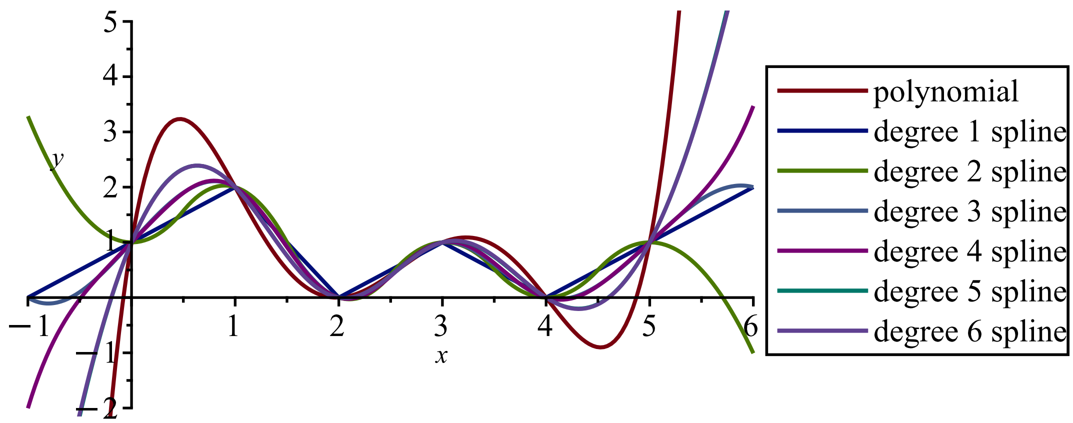



```
> plot([knots,polly,sp2,sp3], x=-1..6,y=-2..5,
  style=[point,line,line,line],symbolsize=20,
  symbol=solidbox, size=[.8,.4],
  color=["DarkRed","DarkGreen",blue,orange],
  legend=["data","polynomial","deg 2 spline","deg 3 spline"]);
```



Let's now do this for splines of all possible degrees, and add a legend. But typing them all is tedious, so let's "compute" the text of the legend.

```
> plot([polly,seq(Spline(knots,x,degree=k),k=1..6)], x=-1..6,y=-2.
  .5,
  style=line,symbolsize=20,
  symbol=solidbox, size=[.8,.4],
  legend=["polynomial",
   seq(sprintf("degree %d spline",k),k=1..6)],legendstyle=
  [location=right]);
```



The **sprintf** command lets me "print" a value into a string, providing some formatting (that's the f in the command)

```
> sayhi:=sprintf("hello %d",54)
```

$$sayhi := \text{"hello 54"} \qquad (11)$$

This is similar to **printf**, which let's me print out a formatted string, but for **sprintf**, the result is a string that I can use, instead of going to the standard output.

```
> printf("hello %d",54)
hello 54
> sayhi[2]
```

"e" (12)

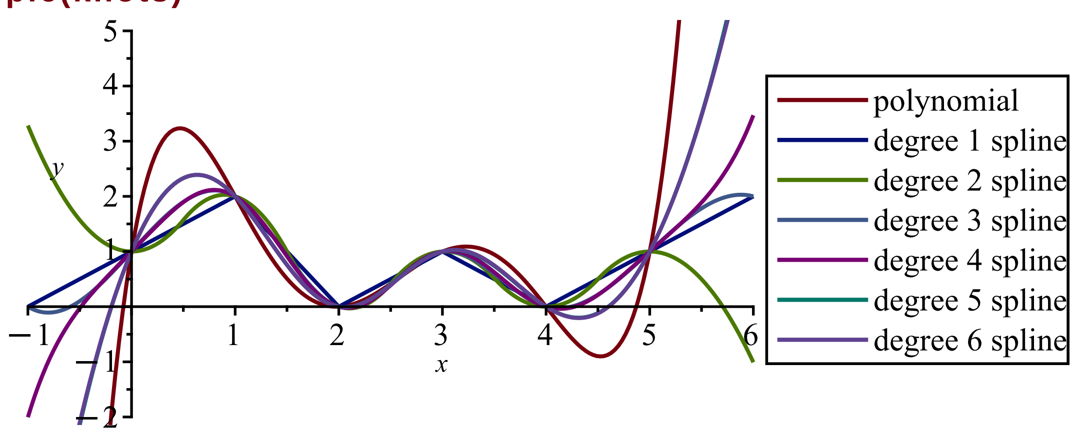
I can automate this into a command, which takes in my data, and gives out the relevant plot

```
> makepic:=data->plot([PolynomialInterpolation(data,x),seq(Spline
  (knots,x,degree=k),k=1..6)], x=-1..6,y=-2..5,
  style=line,symbolsize=20,
  symbol=solidbox, size=[.8,.4],
  legend=["polynomial",
   seq(sprintf("degree %d spline",k),k=1..6)],legendstyle=
  [location=right]);
```

Warning, (in makepic) `k` is implicitly declared local

makepic := data ↦ plot([CurveFitting:-PolynomialInterpolation(data, x), seq(CurveFitting:-Spline(knots, x, degree = k), k = 1 ..6)], x = −1 ..6, y = −2 ..5, style = line, symbolsize = 20, symbol = solidbox, size = [0.8, 0.4], legend = ["polynomial", seq(sprintf("degree %d spline", k), k = 1 ..6)], legendstyle = [location = right]) (13)

```
> makepic(knots)
```



Finally, let's discuss something about Bézier curves, which are related but different. These show up in a lot of computer graphics. See, for example, the Wikipedia page or any number of other references. We won't be using these, but I would be remiss if I didn't mention them.

```
> ?Task,BezierCurves
```

Generate Bézier Curves

Description

A Bézier curve is a polynomial determined by a set of points in such a way that it interpolates the first and last points, but has its shape determined by the remaining points. This task allows you to interactively define the points and view the curve.

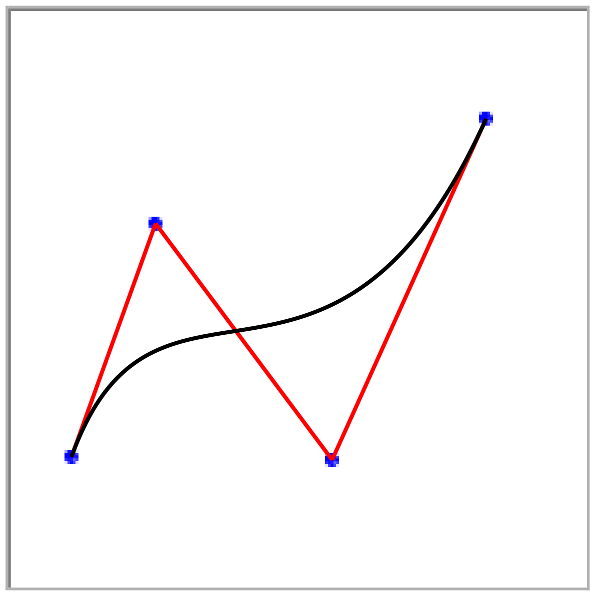
Bézier Curves

- Use the slider to select n, the degree of the Bézier curve.

$n =$ 2 3 4 5

- Press **Initialize** to initialize/clear the plot window.
- Click on the plot area and select the Click and Drag Manipulator ( ) from the Plot menu or plotting toolbar.
- Click to insert $n + 1$ control points $\mathbf{P}_k$, $k = 0, \ldots, n$.
- Drag control points to modify the Bézier curve.
- Below, find the Bézier curve

$$\mathbf{R} = \sum_{k=0}^{n} \binom{n}{k} (1 - u)^{n-k} u^k \, \mathbf{P}_k$$



$$\mathbf{R} = \begin{bmatrix} 1.090343 - 2.023484\,u^3 + 4.879494\,u^2 + 4.443258\,u \\ 2.180685 + 18.41744\,u^3 - 24.78334\,u^2 + 12.31509\,u \end{bmatrix}$$