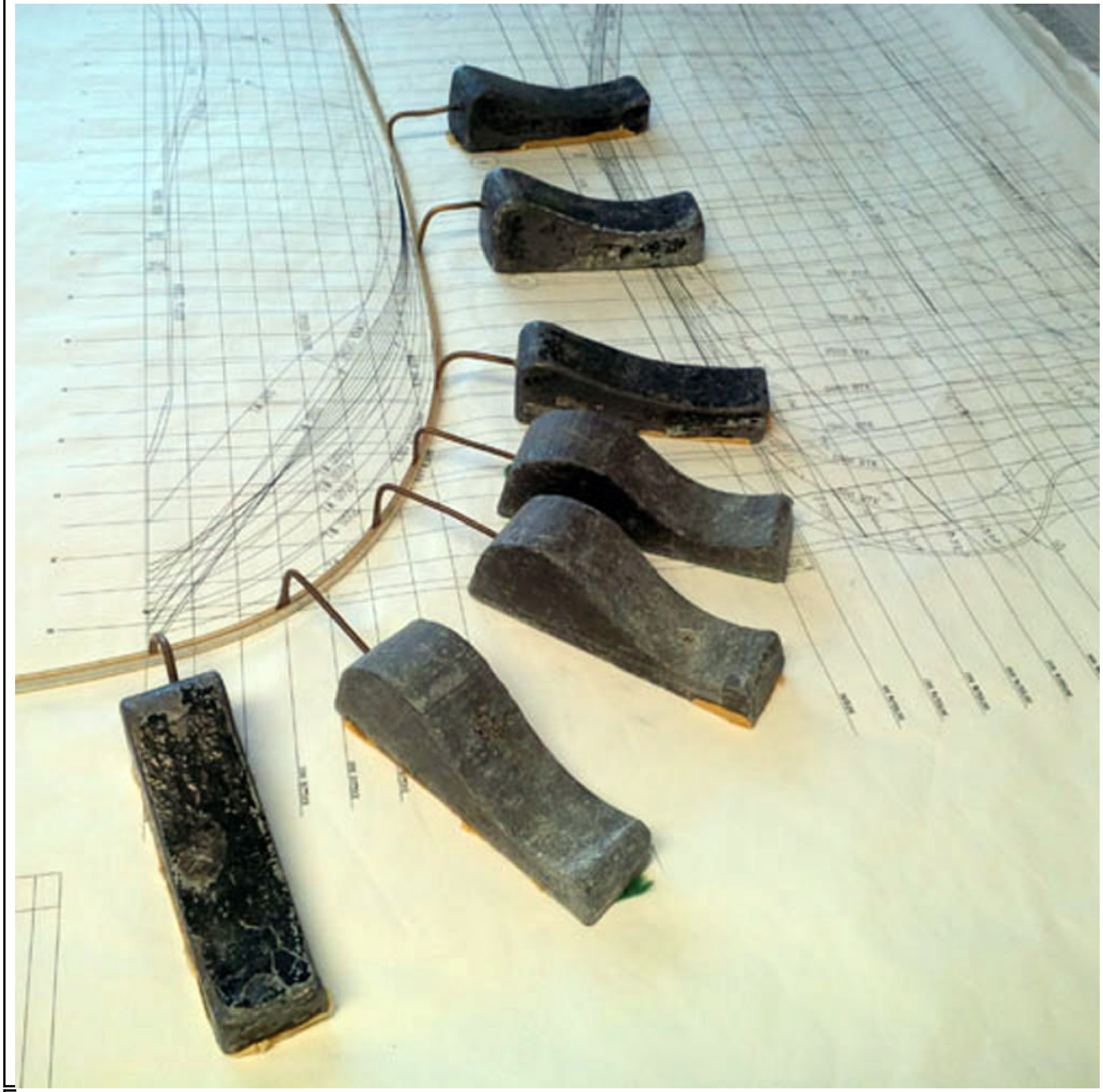
9-12-19 More on splines.

>

some questions from the class... you can ignore, or no.



Splines of degree d through n points (or "knots") have $n(d+1)$ conditions, that is, we have n polynomials $\{p_1, p_2, ..., p_n\}$ 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

$$p_i(x_{i-1}) = y_{i-1}, \qquad p_i(x_i) = y_i \qquad i = 1, 2, ..., n$$

• The derivatives up to $(d-1)^{st}$ order at each of the interior knots must agree, that is,

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

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)
• A periodic spline, forcing the derivatives at the first knot to agree with the derivatives at the final one.

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

$$knots := [[0,1],[1,2],[2,0],[3,1],[4,0],[5,-1]] \quad (1)$$

```
> PolynomialInterpolation(knots,x)
```

$$PolynomialInterpolation(knots, x) \quad (2)$$

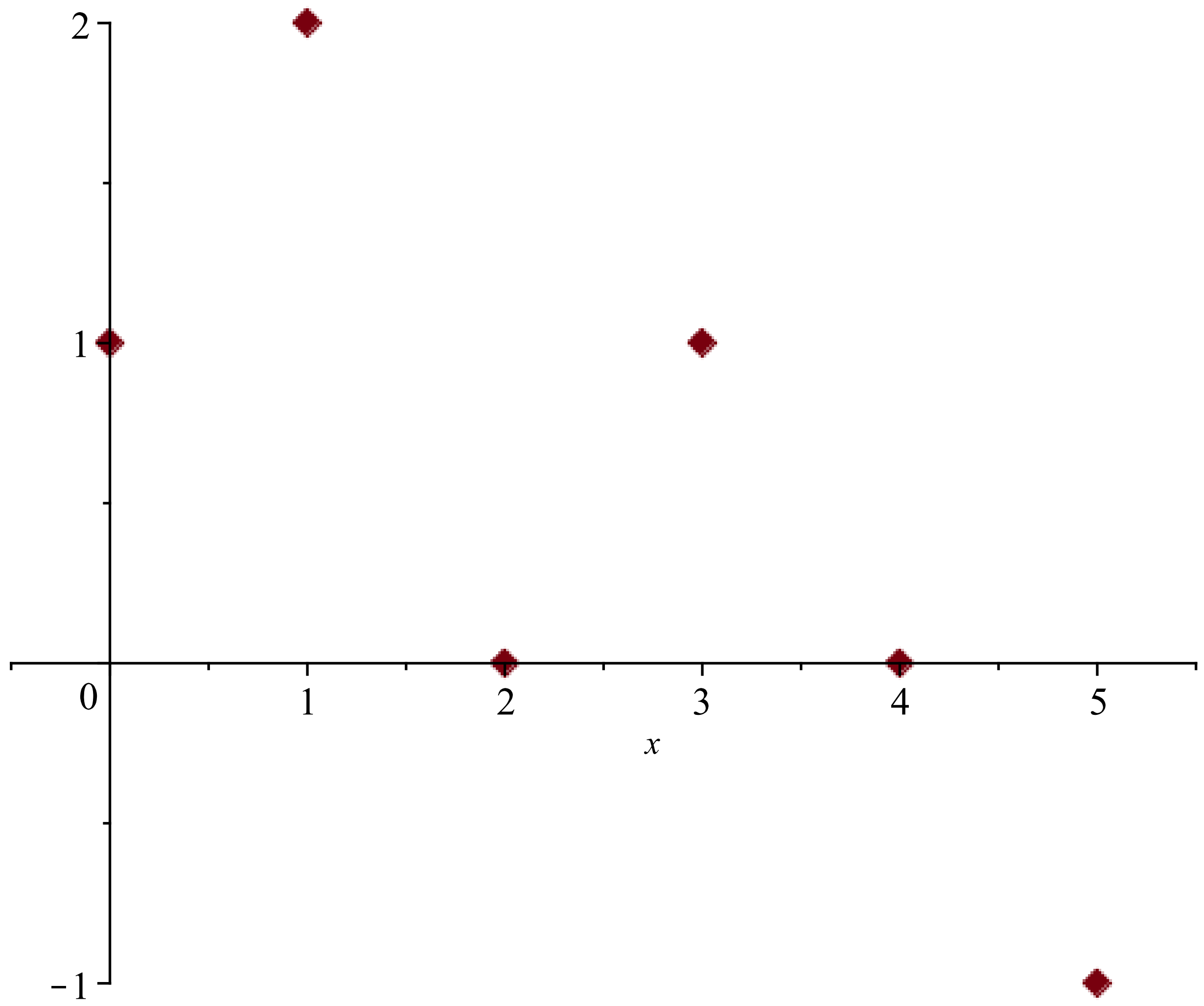
> $with(CurveFitting)$

$$[ArrayInterpolation, BSpline, BSplineCurve, Interactive, LeastSquares, Lowess, PolynomialInterpolation, RationalInterpolation, Spline, ThieleInterpolation] \quad (3)$$

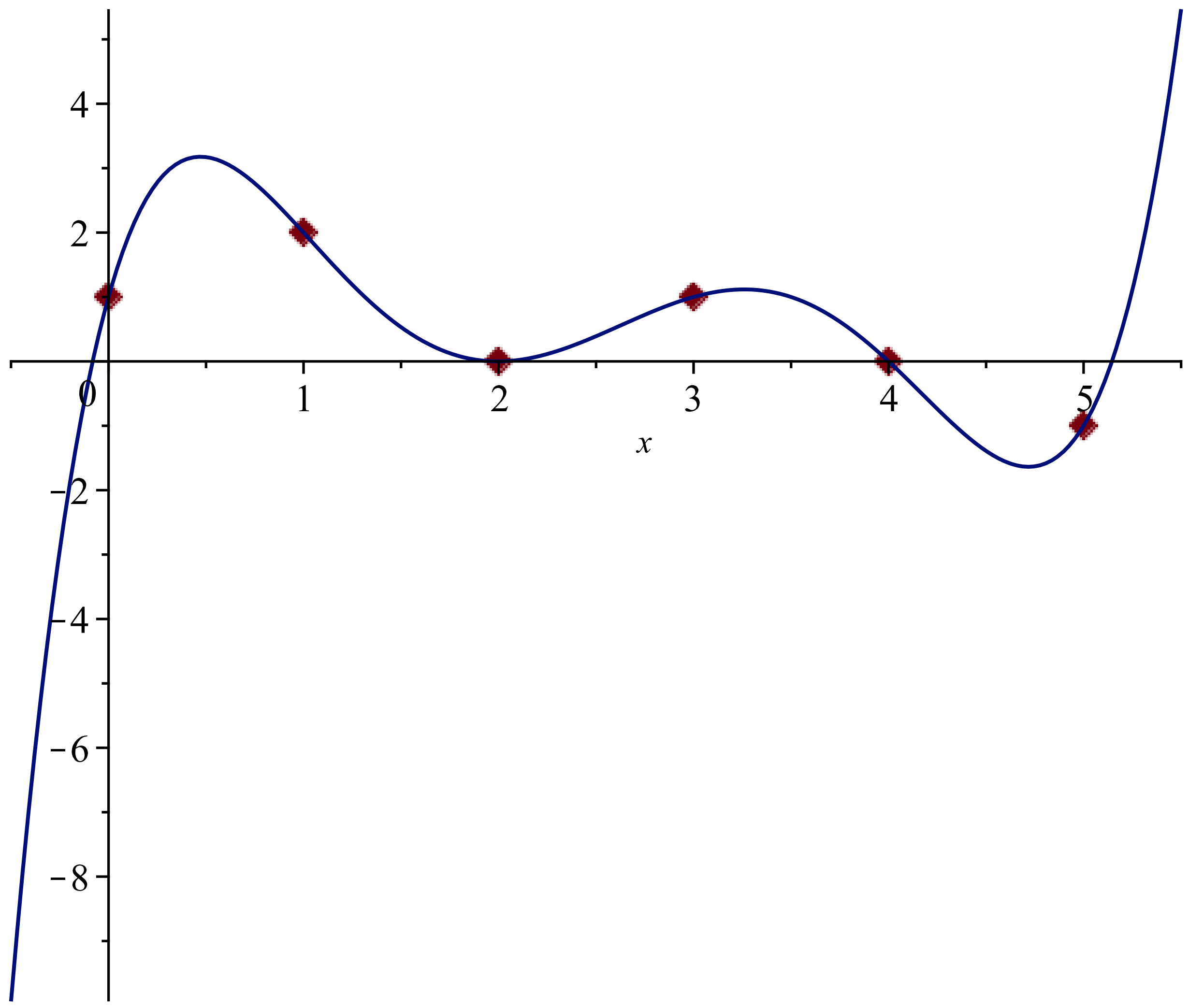
> $polly := PolynomialInterpolation(knots, x)$

$$polly := \frac{3}{20}x^5 - \frac{47}{24}x^4 + 9x^3 - \frac{409}{24}x^2 + \frac{217}{20}x + 1 \quad (4)$$

> $plot(knots, x = -0.5 .. 5.5, style = point, symbolsize = 20, symbol = soliddiamond)$



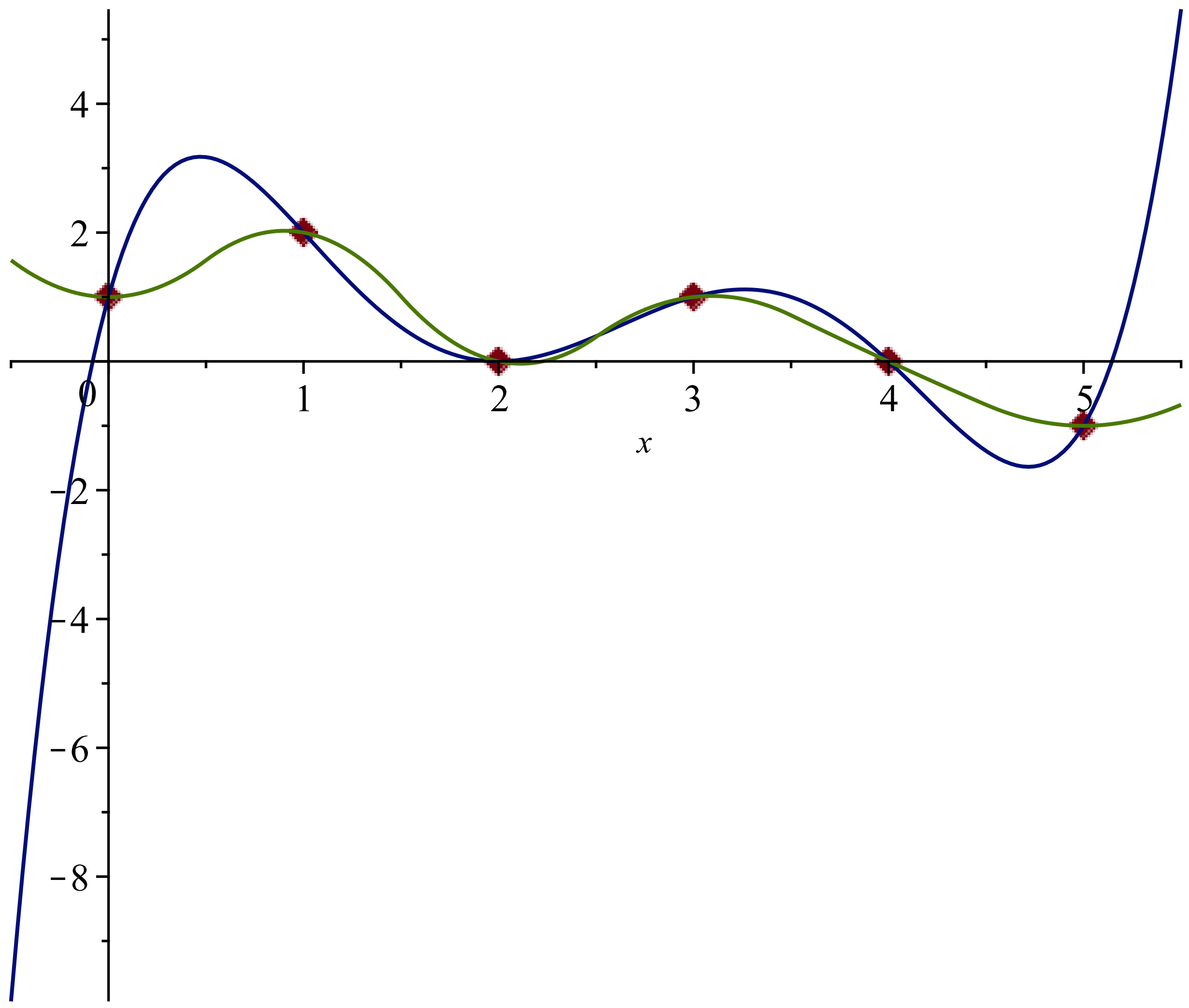
```
> plot([knots, polly], x =-0.5 ..5.5, style = [point, line], symbolsize = 20, symbol = soliddiamond)
```



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

$$sp2 := \begin{cases} \frac{2712\,x^2}{1189} + 1 & x < \frac{1}{2} \\ -\frac{3380}{1189}\,x^2 + \frac{6092}{1189}\,x - \frac{334}{1189} & x < \frac{3}{2} \\ \frac{3300}{1189}\,x^2 - \frac{13948}{1189}\,x + \frac{14696}{1189} & x < \frac{5}{2} \\ -\frac{2152}{1189}\,x^2 + \frac{13312}{1189}\,x - \frac{19379}{1189} & x < \frac{7}{2} \\ \frac{100}{1189}\,x^2 - \frac{2452}{1189}\,x + \frac{8208}{1189} & x < \frac{9}{2} \\ \frac{1552}{1189}\,x^2 - \frac{15520}{1189}\,x + \frac{37611}{1189} & \text{otherwise} \end{cases} \qquad \textbf{(5)}$$

```
> plot([knots, polly, sp2], x =-0.5 ..5.5, style = [point, line$4], symbolsize = 20, symbol
      = soliddiamond)
```



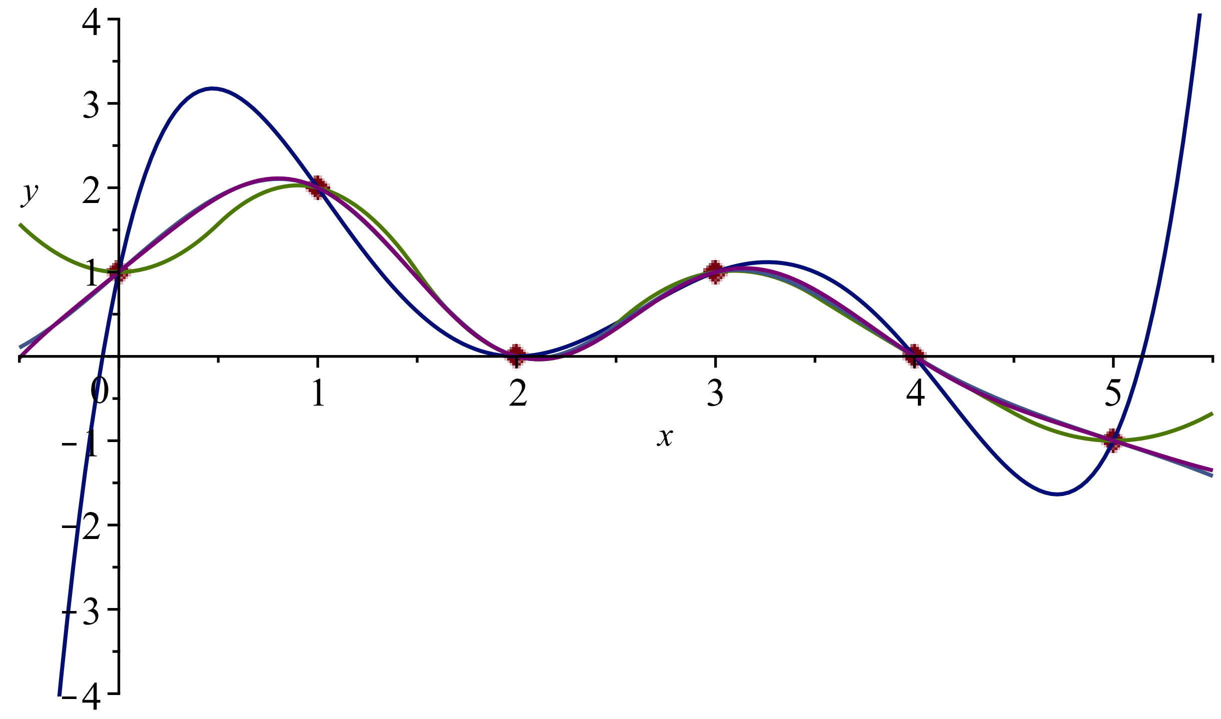
```
> line$4
```

line, line, line, line **(6)**

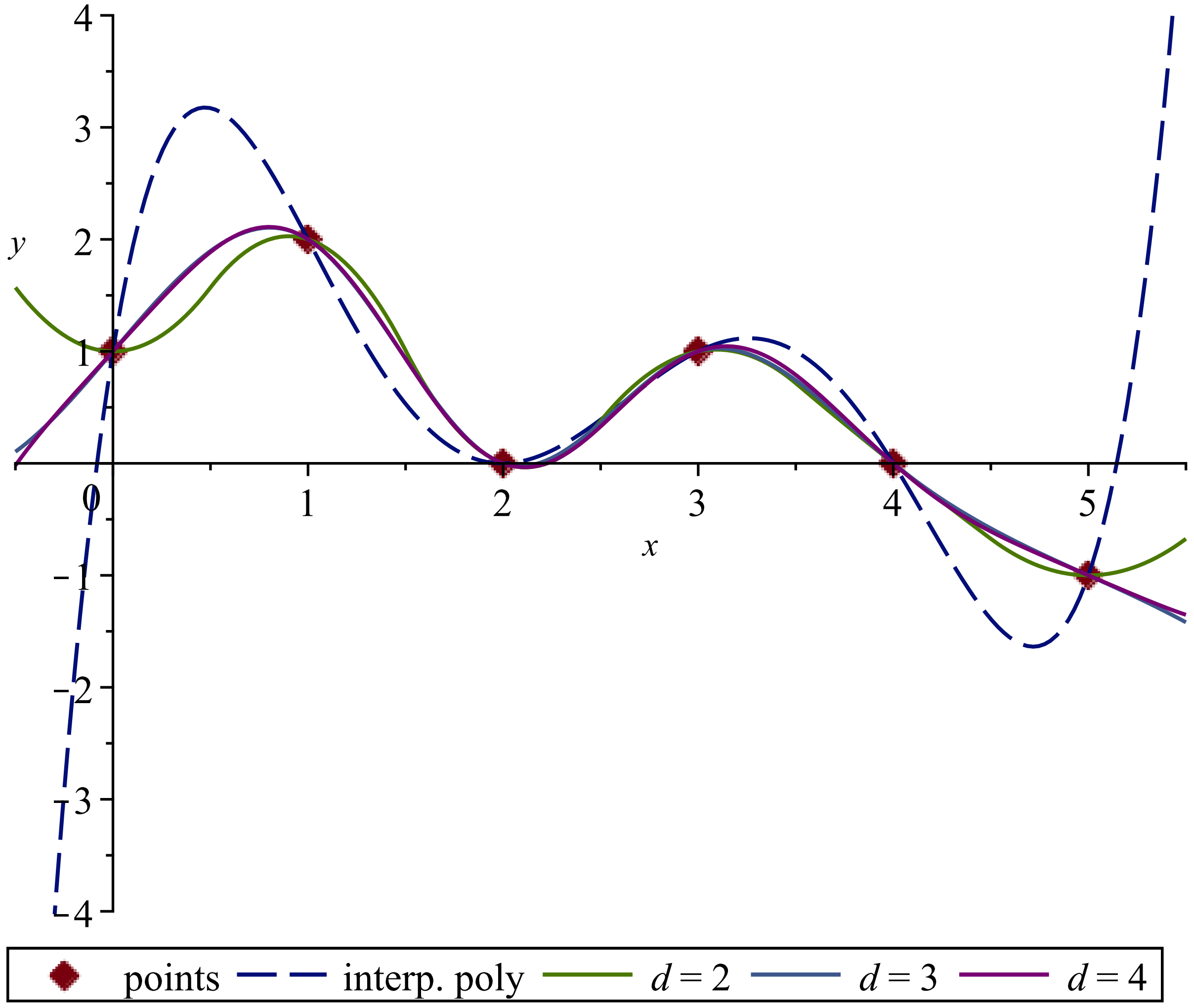
```
>
```

How would I make the earlier picture including splines of degree 2, 3, 4, etc. without retyping them one by one?

```
> plot([knots, polly, seq(Spline(knots, x, degree = n), n = 2 ..4)], x =-0.5 ..5.5, y =-4 ..4,
        style = [point, line$4], symbolsize = 20, symbol = soliddiamond)
```



```
> plot([knots, polly, seq(Spline(knots, x, degree = n), n = 2 ..4)], x =-0.5 ..5.5, y =-4 ..4,
      style = [point, line$4], symbolsize = 20, symbol = soliddiamond, linestyle = [dot, dash, solid
      $4],
      legend = ["points", "interp. poly", d = 2, d = 3 , d = 4])
```



```
> 
> sp4 := Spline(knots, x, degree = 4) :
> diff(x^4, x)
```

$$4\,x^3 \quad (7)$$

```
> diff(diff(x^4, x), x)
```

$$12\,x^2 \quad (8)$$

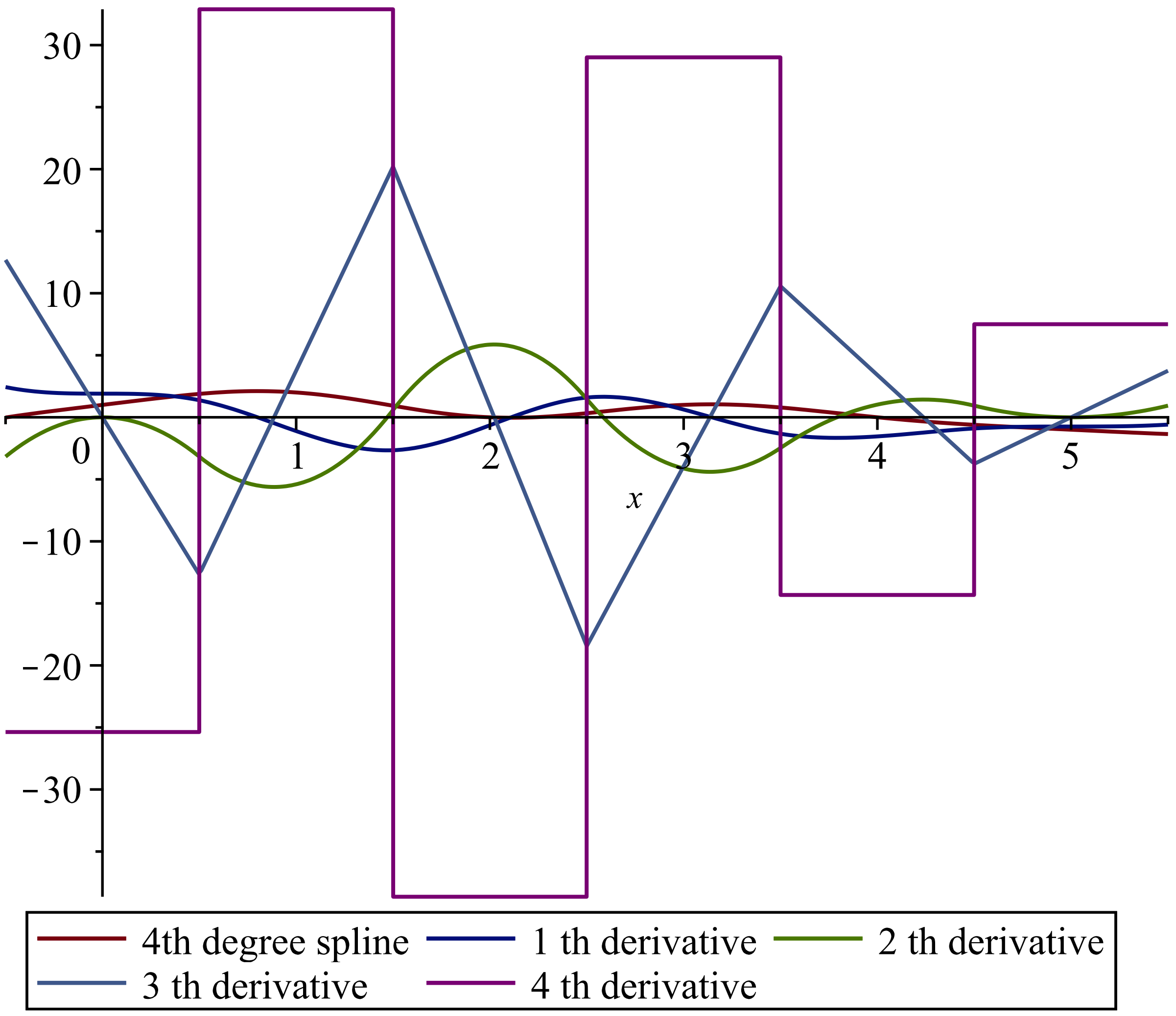
```
> diff(x^4, x$2)
```

$$12\,x^2 \quad (9)$$

```
> diff(x^4, x$4)
```

$$24 \quad (10)$$

```
> plot( [sp4, seq( diff(sp4, x$n), n = 1 ..4) ], x = -0.5 ..5.5,
    legend = ["4th degree spline", seq(sprintf("%d th derivative", n), n = 1 ..4) ])
```



what is this **sprintf** thingy?

> *print*(Pi, x^2, "yo mama")

π, x^2, "yo mama" **(11)**

> *printf*(" how I want to format it, maybe %d is an integer, %f is a floating point, %s is a string", 1234, 27.429, "Yo mama")

```
 how I want to format it, maybe 1234 is an integer, 27.429000 is
a floating point, Yo mama is a string
```

> *mystr* := *sprintf*(" how I want to format it, maybe %d is an integer, %f is a floating point, %s is a string", 1234, 27.429, "Yo mama")

mystr := **(12)**

" how I want to format it, maybe 1234 is an integer, 27.429000 is a floating point, Yo mama is a string"

> *mystr*[15 ..27]

" format it, m" **(13)**

>

Detour on Bezier curves.

> ?*Task,BezierCurves*

copy and past contents of window that came up.

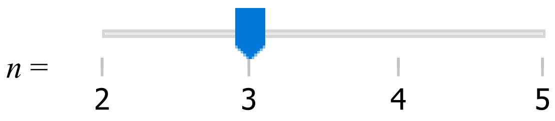
Generate Bézier Curves

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.

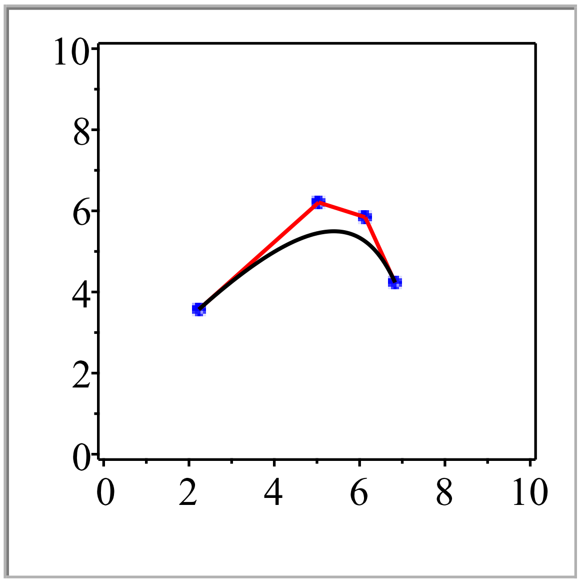
Bézier Curves

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



- 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

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



$$\mathbf{R} = \begin{bmatrix} 2.230971 + 1.327470\, u^3 - 5.151242\, u^2 + 8.423970\, u \\ 1.755122\, (u + 0.3240081)\, (u - 1.649364)\, (u - 3.805699) \end{bmatrix}$$

Commands Used

binomial, plot

See Also

Bézier curve

```
>
```