

Bezier Curves

nalib

```
> restart;
> libname:="c:/nalib",libname;
libname := "/nalib", "/Library/Frameworks/Maple.framework/Versions/15/lib",
"/Library/Frameworks/Maple.framework/Versions/15/toolbox/NAG/lib"
> with(numanal);
[SOR, SOR_dir, adaptq, adaptq_dir, bezier, bezier_dir, bisection, bisection_dir, chop, chop_dir,
clamped_spline, clamped_spline_dir, divided_diff, divided_diff_dir, extrap, extrap_dir,
falseposition, falseposition_dir, fixedpoint, fixedpoint_dir, gaussseidel, gaussseidel_dir, hermite,
hermite_dd, hermite_dd_dir, hermite_dir, horner, horner_dir, jacobi, jacobi_dir, muller,
muller_dir, natural_spline, natural_spline_dir, newton, newton_dir, romberg, romberg_dir,
secant, secant_dir, steffensen, steffensen_dir]
```

Page 17, Problem 3c

We want to construct and graph the bezier curve going from point (0,0) with guidepoint (0.5,0.5) to point (4,6) with entering guidepoint (3.5,7) and exiting guidepoint (4.5,5) to point (6,1) with guidepoint (7,2).

We first look at the directions for **bezier**.

```
> bezier_dir();
bezier returns a list where each component is a list
containing a starting point, an ending point, and the two
cubic parametric equations between the points.
```

The arguments for bezier are:

- (1) the list of points $[x[i], y[i]]$ for $i=0..n$
- (2) the list of left (or exiting) guideposts $[x[i+], y[i+]]$ for $i=0..n-1$
- (3) the list of right (or entering) guideposts $[x[i]-, y[i]-]$ for $i=1..n$
- (4) the variable for returning the bezier curve

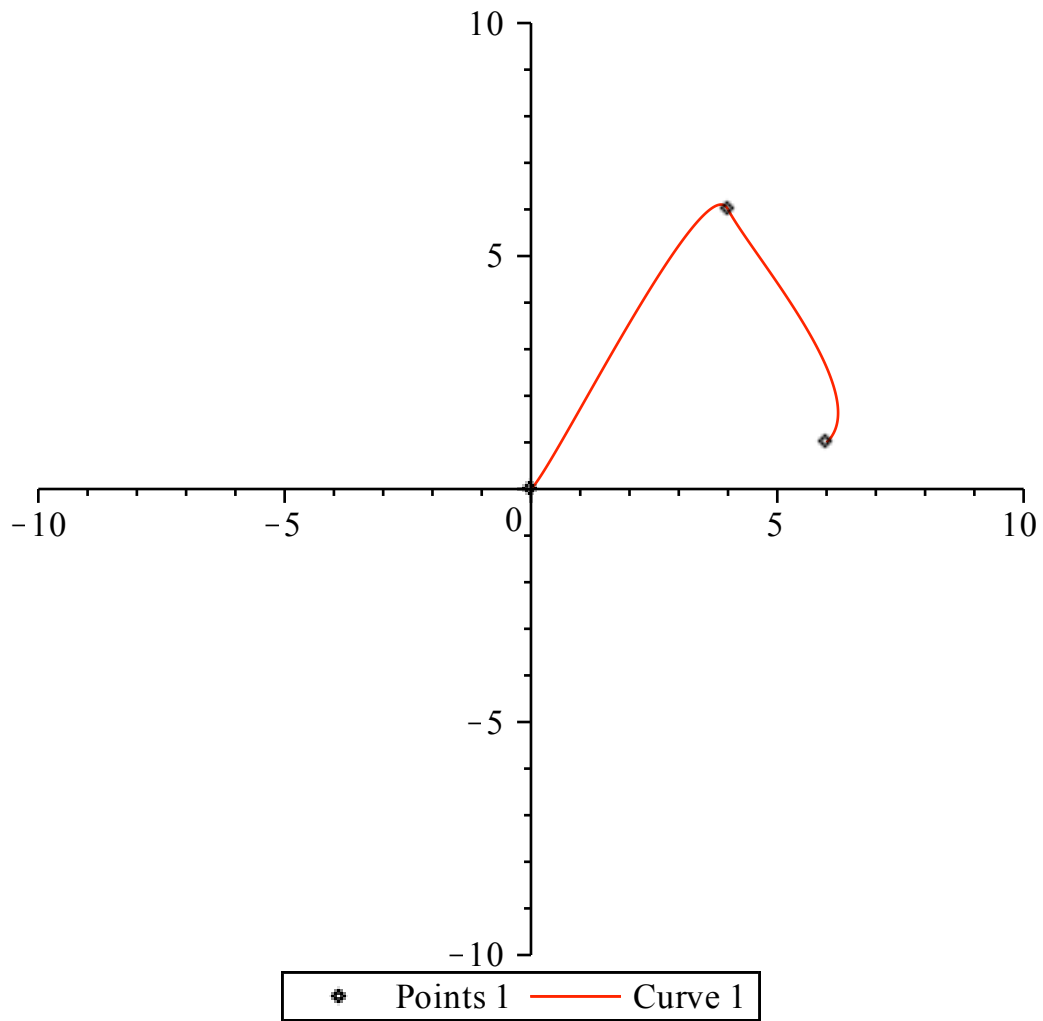
If assigning the result to a variable, have the variable and the 4th argument the same.

If b is the variable for returning the bezier curve and has already been given a value, the procedure should be preceded by the statement:
 $b := 'b'$

```
> b:=bezier([[0,0],[4,6],[6,1]],[[.5,.5],[4.5,5]],[[3.5,7],[7,2]],b);
b := [[ [0, 0], [4, 6],  $x_0(t) = -5.0 t^3 + 7.5 t^2 + 1.5 t$ ,  $y_0(t) = -13.5 t^3 + 18.0 t^2 + 1.5 t$  ], [ [4, 6],
[6, 1],  $x_1(t) = -5.5 t^3 + 6.0 t^2 + 1.5 t + 4$ ,  $y_1(t) = 4 t^3 - 6 t^2 - 3 t + 6 ] ]]$ 
```

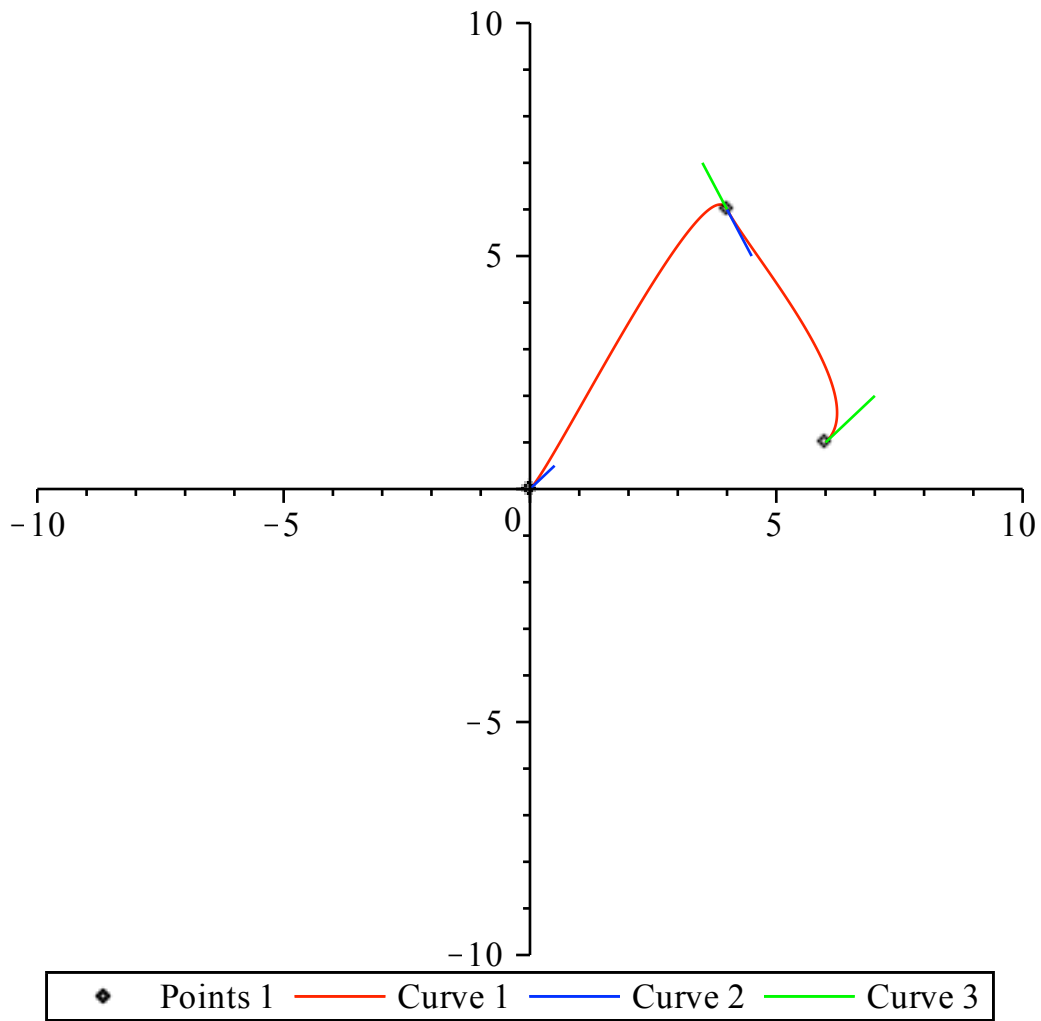
We do a [parametric plot](#) of the bezier curve one piece at a time.

```
> p1:=plot([rhs(op(3,b[1])),rhs(op(4,b[1])),t=0..1],[-10..10,-10..10]);
> p2:=plot([rhs(op(3,b[2])),rhs(op(4,b[2])),t=0..1],[-10..10,-10..10]);
> with(plots):
> p3:=pointplot({[0,0],[4,6],[6,1]});
> display(p1,p2,p3);
```



Let's add in the guideposts.

```
> p4:=pointplot({[0,0],[.5,.5]},style=line,color=blue):
> p5:=pointplot({[4,6],[4.5,5]},style=line,color=blue):
> p6:=pointplot({[4,6],[3.5,7]},style=line,color=green):
> p7:=pointplot({[6,1],[7,2]},style=line,color=green):
> display(p1,p2,p3,p4,p5,p6,p7);
```



Generate Bézier Curves

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.

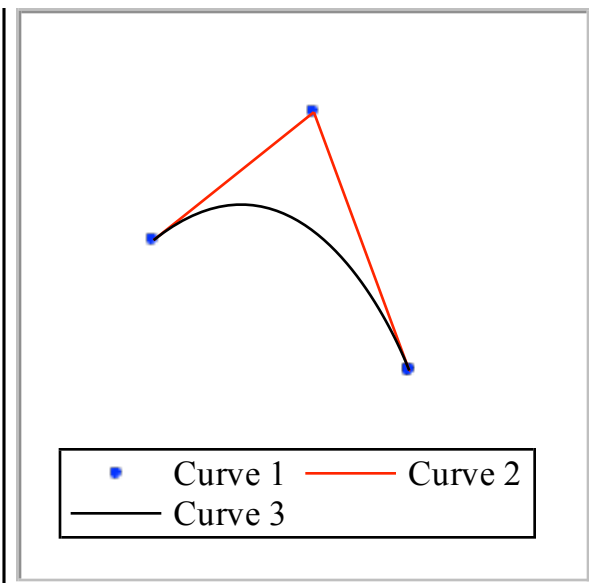
• Click on the plot area and select the Click and Drag Manipulator () from the Plot menu or plotting toolbar.

• Click to insert $n + 1$ control points $\mathbf{P}_k, k = 0, \dots, n$.

• Drag control points to modify the Bézier curve.

• Below, find the Bézier curve

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



R =

$$\begin{bmatrix} 2.265625 + 9.062500 u - 3.671875 u^2 - 0.5729167 u^3 \\ 5.027778 + 9.000000 u - 27.16667 u^2 + 15.11111 u^3 \end{bmatrix}$$