# Lab G: Parametric Equations

## `ParametricPlot[]`

`ParametricPlot[]` is the *Mathematica* command that does parametric plots. You have seen several examples of this used in class. Here is an example:

```
ParametricPlot[{Sin[t],Cos[2t]}, {t, 0, 4*Pi}, AspectRatio->Automatic]
```

You can use `?ParametricPlot` to get additional information about using this command.

**1.** Give the example above a try. You should see a closed curve. How many times is it traced for $t \in [0, 4\pi]$? In which direction is the curve traced? How do you know?

## Bézier Curves

Bézier Curves are used in computer-aided design and are named after a mathematician who used them in work for the automotive industry. These curves make "smooth" paths between two specified points. Of course, one could just use a straight line, but that would be pretty boring. To modify the path between two points, Bézier curves specify in addition to the endpoints additional *control points.* In this lab we will learn a little bit about Bézier curves that use four control points (two endpoints and two other points).

If the four control points are $P_0 = (x_0, y_0), P_1 = (x_1, y_1), P_2 = (x_2, y_2), P_3 = (x_3, y_3)$, then the parametric equations for the curve are given by

$$x = x_0(1-t)^3 + 3x_1t(1-t)^2 + 3x_2t^2(1-t) + x_3t^3 \quad (1)$$

$$y = y_0(1-t)^3 + 3y_1t(1-t)^2 + 3y_2t^2(1-t) + y_3t^3 \quad (2)$$

where $0 \le t \le 1$.

**2.** Explain why the curve always starts at $P_0$ and ends at $P_3$.

**3.** Notice that the forms of the functions $x(t)$ and $y(t)$ are identical – the only difference is the parameters used to set the coordinates of the control points. Define a function `bez[a_,b_,c_,d_,t_]` so that

```
x[t_]:= bez[0,1,2,1,t]
y[t_]:= bez[0,1,-1,0,t]
```

can be used to define $x(t)$ and $y(t)$ for a Bezier curve with control points $(0, 0)$, $(1, 1)$, $(2, -1)$, and $(1, 0)$. Plot this Bézier curve using `ParametericPlot[]`.

**4.** What point on the curve corresponds to $t = \frac{1}{2}$? What is the slope of the tangent line at that point?[1]

**5.** Find all points on this curve that have a horizontal or vertical tangent. (Use `Solve[]` and/or `NSolve[]`.)

**6.** What is the length of the curve above? (Use `NIntegrate[]`.)

**7.** Now graph the Bézier curve together with the three line segments connecting the four control points on the same graph.[2] (Remember: it is easy to get a parametric equation for the graph of a line segment using $t \in [0, 1]$.)

You can put this all into one function if you are clever (and learn a new thing about *Mathematica*.) Here is an

[1] You can use `x'[t]` and `y'[t]` to get the derivatives.

[2] You can probably guess how to get multiple parametric plots on one graph. If not, try `?ParametricPlot`.

example that just plots the curve (without the line segments):

```
bezier[x0_,y0_,x1_,y1_,x2_,y2_,x3_,y3_]:=
(
   Clear[bez,x,y];
   bez[a_,b_,c_,d_,t_]:= <put your definition here> ;
   x[t_] := bez[x0,x1,x2,x3,t];
   y[t_] := bez[y0,y1,y2,y3,t];
   ParametricPlot[{x[t],y[t]}, {t,0,1}]
)
```

The parenthesized list of commands separated by semicolons tells *Mathematica* to execute the sequence of commands whenever you call `bezier`.

**8.** How does the slope of the Bézier curve compare with the slopes of the line segments? Prove that what you observe is correct for any Bèzier curve.

**9.** One application of Bézier curves is to represent letters and other symbols that are printed by laser printers. Experiment with control points until you find a reasonable representation of the letter 'C'.

**10.** How can you paste two Bézier curves together so that there is not a "corner" where they meet? Use this to get a Bézier curve that looks like the letter 'S'.

**11.** Can you get a single Bézier curve to have a loop? If so, do it; if not, explain why this is impossible.

## `Do[] and Animations`

You can check your answer to the problem 1 using the following command.

```
Do[
  ParametricPlot[{Sin[t],Cos[2t]}, {t, 0, max},
    AspectRatio->Automatic, PlotRange->{{-1.5,1.5},{-1.5,1.5}}],
  {max, 0, 4*Pi, Pi/4}
]
```

`Do[]` is a command we haven't seen before. Basically `Do[]` does things repeatedly. You need to tell it what to do (in this case make plots) and what paramters to modify for each try (in this case the end of the interval for $t$). When you execute the command above you should see a bunch of graphs.

**12.** How many graphs were generated by the command above? Why?

Now for the fun part: Double click on the last graph and you will get an animation. There are controls that let you stop, start and change the speed of the animation. Any sequence of graphs generated with `Do[]` can be animated. Usually you will want to specify the `PlotRange` so that they are all to the same scale and use the same viewing rectangle.

**13.** Generate a parametric plot that fits in the box $[-3, 3] \times [-2, 2]$ and is "aesthetically pleasing" or interesting is some way. Hint: You can use trigonometric functions to keep things inside the box.