Parametric and Polar Curves with Maple[1]

Preliminaries

Start Maple by clicking on the icon that appears either on your desktop or the dock. Go to the **Maple** menu and select **Preferences**. A dialog window will pop up, click on **Display** and select **window** for **Plot display**, then click the **Apply to Session** button at the bottom of the box (this will make each plot appear on a separate window).

At the prompt, type:

```
> with(plots):
```

This will allow you to use different plotting commands available in Maple.

Don't forget to type a semicolon (;) (or a colon (:) if you don't want to see the output) at the end of each command!!!

Plotting Parametric Curves

The `plot` command in Maple allows you to plot parametric curves using the syntax:

> `plot([x(t),y(t),t=t_0..t_f],` *options*`);`

This will produce the parametric curve defined by `(x(t),y(t))` when the parameter `t` ranges from `t_0` to `t_f`. Different *options* are available for color, axis, scaling, etc. Try this:

```
> plot([cos(t), sin(t), t=0..2*Pi], scaling=constrained);
```

Now, let's define some functions of `t` and produce different parametric plots with them:

```
> f1 := t-> 3*cos(2*t) ; g1 := t-> sin(4*t) ;
> plot([f1(t), g1(t), t=0..2*Pi]);
> f2 := t-> cos(t)/ln(t) ; g2 := t-> sin(t)/ln(t) ;
> plot([f2(t), g2(t), t=3..20]);
> plot({[f1(t), g1(t), t=0..2*Pi], [f2(t), g2(t), t=3..20]});
```

Try different combinations of `f1(t)`, `f2(t)`, `g1(t)`, and `g2(t)` over different intervals of `t`. You can also create animations to visualize (and understand) the effects that different scaling factors have in a given curve; try, for example:

```
> animate([2*cos(3*t), b*sin(2*t), t=0..2*Pi, numpoints=500], b=2..10);
```

and

[1] This worksheet has been prepared with the help of online materials available from Maple's Application Center.

```
> animate([2*cos(3*t), (6 + 4*sin(b))*sin(2*t), t=0..2*Pi], b=0..2*Pi);
```

Play with the controls in the plot window and see if there are any differences between the two animations.

Families of Polar Curves

To plot polar curves, we use the command `polarplot`. Try some cardioids,

$$r = a \pm b\sin\theta \quad \text{or} \quad r = a \pm b\cos\theta$$

for different values of a and b (e.g., see what happens when $|a| = |b|$, when $|a|\ |b|$, and when $|a| < |b|$). You can see this all at once by typing:

```
> display( polarplot( 8 + 8*cos(theta) , theta = 0..2*Pi, scaling = constrained,
 color = green, thickness = 3), polarplot({8 + a*cos(theta) $ a = 9..15},
 theta = 0..2*Pi, color = blue), polarplot({ 8 + a*cos(theta) $ a = 1..7},
 theta = 0..2*Pi, color = red));
```

You should also become familiar with *multi-petaled roses*,

$$r = a\sin b\theta \quad \text{or} \quad r = a\cos b\theta$$

Try different values of a and b:

```
> polarplot({sin(3*theta), cos(3*theta)}, theta = 0..2*Pi, scaling = constrained);
> polarplot({sin(6*theta), cos(6*theta)}, theta = 0..2*Pi, scaling = constrained);
```

Is there any difference between those with b even and those with b odd? What does the value of a determine? Try this:

```
> polarplot({a*cos(6*theta) $ a = 1..12}, theta = 0..2*Pi, scaling = constrained);
```

You can also try some *hybrids* of these curves:

```
> polarplot( { 6 + a*cos(6*theta) $ a = 1..11}, theta = 0..2*Pi,
scaling = constrained);
> polarplot( {12 + a*sin(7*theta) $ a = 1..12}, theta = 0..2*Pi,
scaling = constrained);
```

or some animations:

```
> animate(cos(t*theta), theta=0..2*Pi,t=1..20, coords=polar, color=black,
thickness=2, numpoints=1000, scaling=constrained, frames=20);
> animate(-cos(t*theta), theta=0..2*Pi, t=1..20, coords=polar, color=aquamarine,
thickness=2, numpoints=200, scaling=constrained, frames=100, axes=none);
```

Arc Length of Parametric Curves

Here, we are going to approximate the arclength of a parametric curve by approximating the curve with linear pieces and adding the lengths of those pieces.

We first define the curve with the commands:

```
> f := t-> t^2 ; g := t-> t^3 - 3*t ;
```

You may want to plot it to see what it looks like.

Type the following commands to create a Maple *procedure* that approximates a given function `f` by `n` linear pieces over the interval `[a,b]`:

```
> pl := proc(f,a,b,n)
>         local u,v,k;
>         k := 1 + floor(n*(x-a)/(b-a));
>         u := a + (k-1)*(b-a)/n;
>         v := a + k*(b-a)/n;
>         unapply(f(u) + (f(v) - f(u))/(v-u)*(x-u), x);
>       end proc:
```

The following two commands approximate the functions `f` and `g` using six linear pieces over the interval `[-2,2]`:

```
> plf := pl(f,-2,2,6):
> plg := pl(g,-2,2,6):
```

This will allow us to plot the parametric curve and its linear approximation:

```
> plot({[f(t),g(t),t=-2..2], [plf(t),plg(t),t=-2..2]}, thickness=2);
```

We then approximate the arclength of the curve as:

$$L \approx \sum_{i=1}^{n} \sqrt{(f(t_{i+1}) - f(t_i))^2 + (g(t_{i+1}) - g(t_i))^2}.$$

That we define as another Maple procedure:

```
> approxlen := proc(f,g,a,b,n)
                 local l;
                 l := (b-a)/n;
                 sum( sqrt( (f(a+j*l) - f(a+(j-1)*l))^2
                   + (g(a+j*l) - g(a+(j-1)*l))^2 ), j=1..n);
               end proc:
```

Calculate different approximations until you find a convergent value:

```
> evalf(approxlen(f,g,-2,2,5));
> evalf(approxlen(f,g,-2,2,10));
> evalf(approxlen(f,g,-2,2,20));
> evalf(approxlen(f,g,-2,2,50));
> evalf(approxlen(f,g,-2,2,100));
```

Finally, we calculate the exact arclength, given by:

$$L = \int_a^b \sqrt{\left(\frac{df}{dt}\right)^2 + \left(\frac{dg}{dt}\right)^2}\,dt,$$

that we implement by typing:

```
> arclength := proc(f,g,a,b)
                Int(sqrt(diff(f(t),t)^2 + diff(g(t),t)^2), t=a..b);
              end proc:
```

followed by:

```
> evalf(arclength(f,g,-2,2));
```

This is it! Save this hand out as it may be useful for the computer project that will be assigned with homework #2.