Parametric

October 28, 2019

1 Homework 7 - Q4

We're going to plot some serious Parametric Equations now…

Get ready to behold the beauty of some math (Don't cringe!)

I've used Wikipedia extensively for this one, especially for the Lissajous Figures, Hypotrochoid and Epicycloid.

Among imports, we're mostly going to need `matplotlib` for plotting, `numpy` to generate float ranges, using the `arange()` function I learnt last time, and the regular `math` package to use `cos` and `sin` functions and `pi`'s value…

[1]:
```
import matplotlib.pyplot as plt
import numpy as np
import math
```

1.1 Parabola

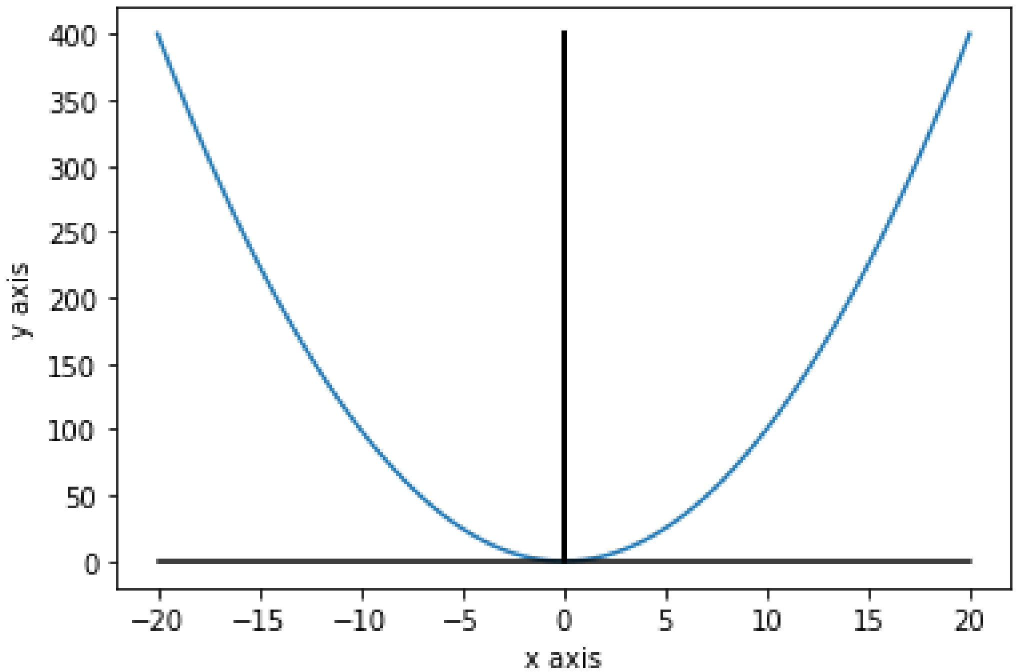
This one's really simple, all we're going to do is produce a set of values of x, and then create our y using the cool `map` function and the amazing `lambda` operator available in Python.

[2]:
```
x = np.arange(-20,20.5,0.5)
y = list(map(lambda t:t*t, x))
```

Next, we're going to plot the Parabola. Just to make it look better, I'm also plotting the values of x and y while keeping the other one fixed as zero, just to make it look like an x-axis and a y-axis.

[3]:
```
z = plt.plot(x,y,label='Parabola, Parametrized in terms of x')
xaxis = plt.plot(x,len(x)*[0], color='black')
yaxis = plt.plot(len(y)*[0],y,color='black')
plt.xlabel('x axis')
plt.ylabel('y axis')
```

[3]:
```
Text(0, 0.5, 'y axis')
```



1.2 Circle

This one is where we're first going to use the cos() and sin() functions.

Now, we're going to create a list of values of θ from 0 to 2π, using the `arange` function again.

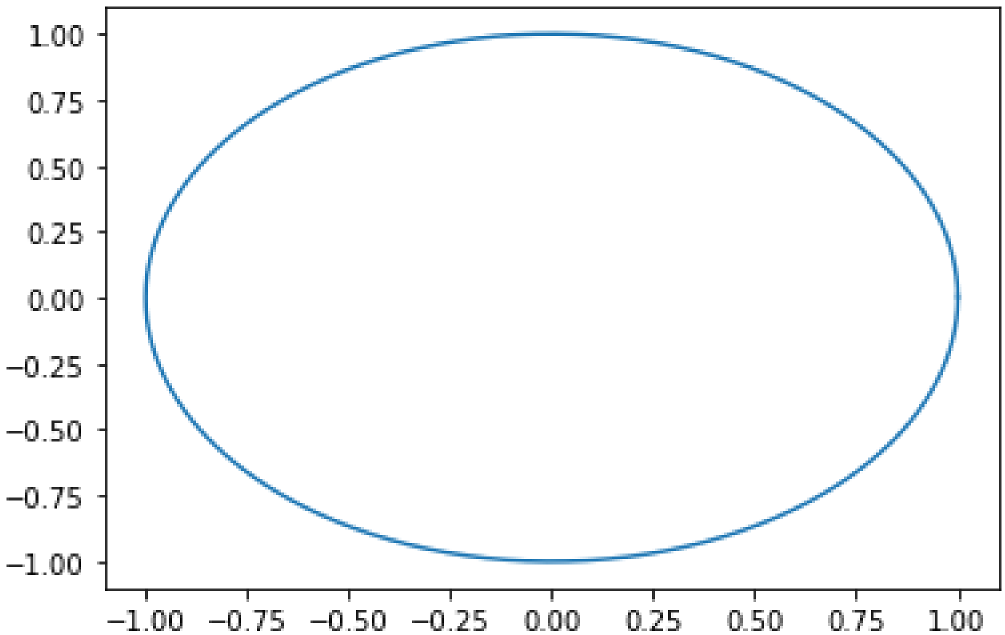
[4]:
```
t = np.arange(0,2*math.pi+0.01, 0.01)
```

We need to use the `plt.axis('equal')` command to make a circle look like an actual circle, as `matplotlib` likes to squeeze and stretch our data when we don't use it.

See the output before and after:

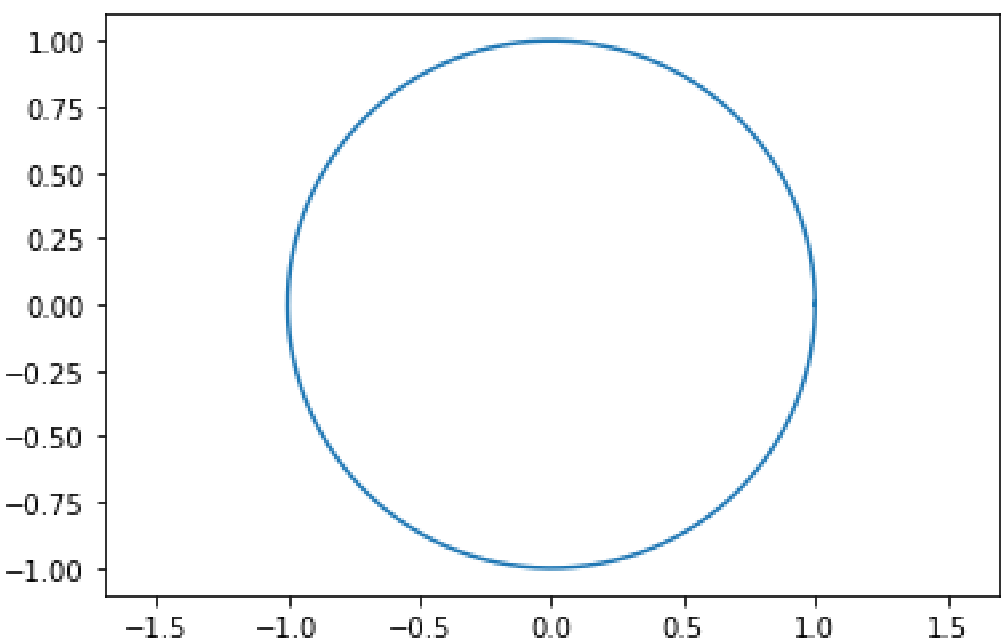
[5]:
```
plt.plot(list(map(lambda x:math.cos(x),t)),list(map(lambda x:math.sin(x),t)))
print("Before:")
```

```
Before:
```



[6]:
```
plt.axis('equal')
plt.plot(list(map(lambda x:math.cos(x),t)),list(map(lambda x:math.sin(x),t)))
print("After:")
```

After:

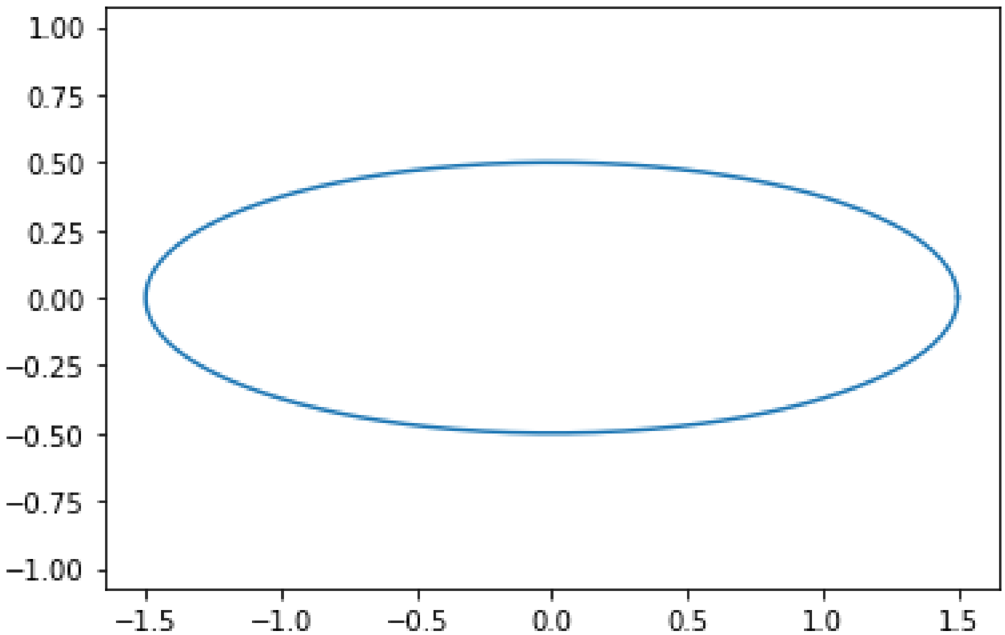


1.3 Ellipse

Pretty much the same thing as a circle… Honestly, I just copied the code, and scaled x and y :)

[7]:
```
plt.axis('equal')
plt.plot(list(map(lambda x:1.5*math.cos(x),t)),list(map(lambda x:0.5*math.
 ↪sin(x),t)))
```

[7]:
```
[<matplotlib.lines.Line2D at 0x7f0d4b5267f0>]
```



1.4 Lissajous Curves:

This is where the challenge slightly increases. At least in terms of understanding…

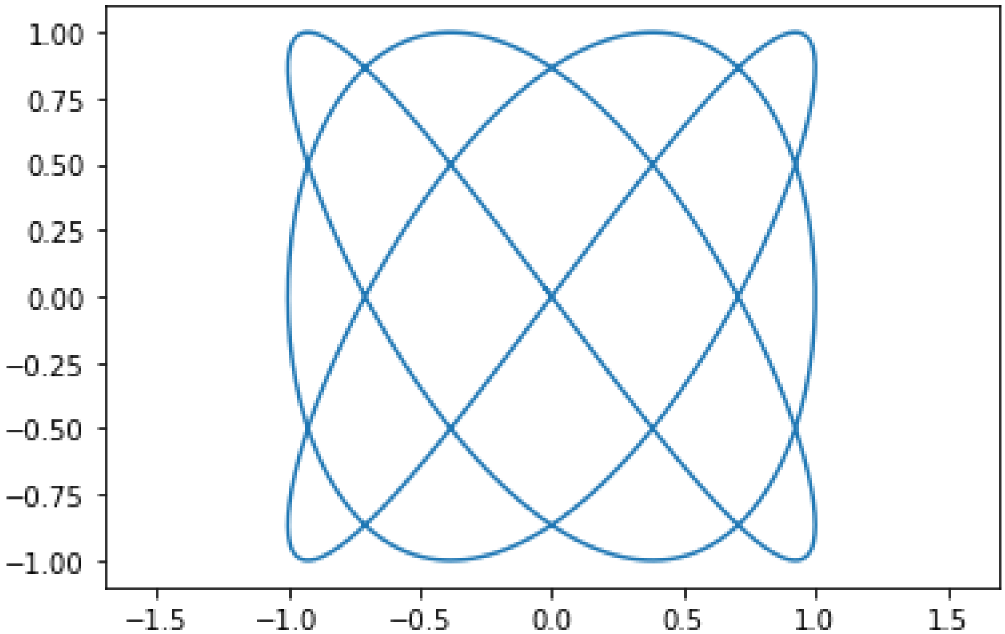
A Lissajous Curve is a curve that is of the following form:

$$x = A\cos(at + \delta), y = B\sin(bt)$$

Now, in my case, I took $A = 1, B = 1, \delta = \frac{\pi}{2}, a = 3$ and $b = 4$

[8]:
```
plt.axis('equal')
plt.plot(list(map(lambda x:math.cos(3*x+math.pi/2),t)),list(map(lambda x:math.
 ↪sin(4*x),t)))
```

```
[8]: [<matplotlib.lines.Line2D at 0x7f0d4b493898>]
```



1.5 Hypotrochoid

The equation of a Hypotrochoid is as follows:

$$x(t) = (R-r)\cos(t) + d\cos\left(\frac{R-r}{r}t\right), y = (R-r)\sin(t) - d\sin\left(\frac{R-r}{r}t\right)$$

I've chosen $R = 15$, $r = 7$ and $d = 4$.

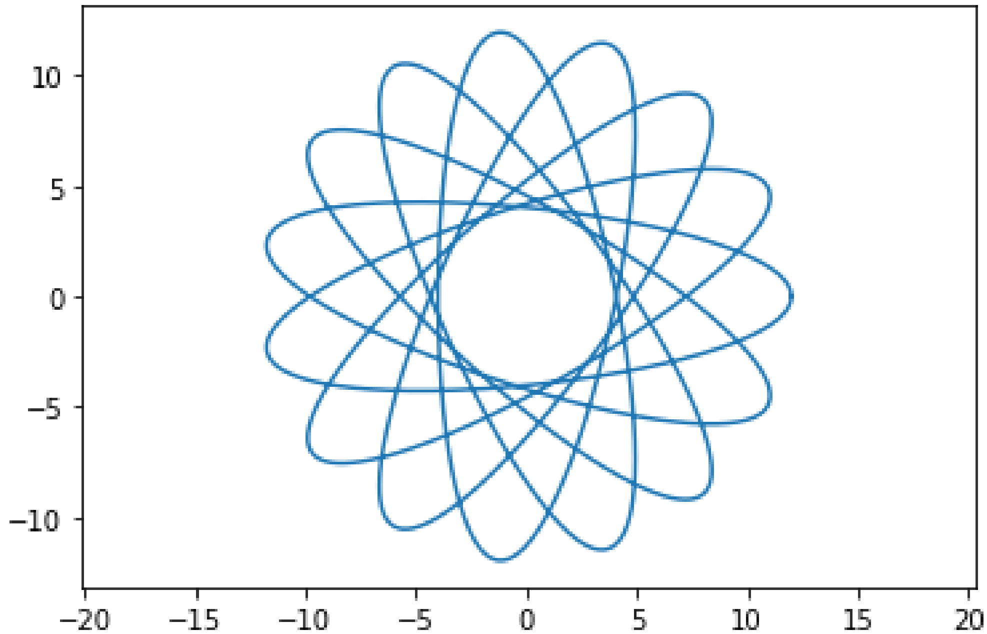
(It's a little better if it can be visualised while moving, for which I recommend: https://www.desmos.com/calculator/3plby3pgqv)

(Note that I've taken t to range from 0 to 30π, since the range of the parameter is given by:

$$[0, 2\pi \times \frac{LCM(r,R)}{R}] = [0, 14\pi]$$

```
[9]: plt.axis('equal')
R = 15
r = 7
d = 4
u = np.arange(0,14*math.pi+0.01,0.01)
plt.plot(list(map(lambda x: (R-r)*math.cos(x)+d*math.cos((R-r)/r *␣
 ↪x),u)),list(map(lambda x: (R-r)*math.sin(x)-d*math.sin((R-r)/r * x),u)))
```

[9]: [<matplotlib.lines.Line2D at 0x7f0d4b47fb70>]



1.6 Epicycloid

The equation of an Epicycloid is as follows:

$$x(t) = (R+r)\cos(t) - r\cos\left(\frac{R+r}{r}t\right), y = (R+r)\sin(t) - r\sin\left(\frac{R+r}{r}t\right)$$

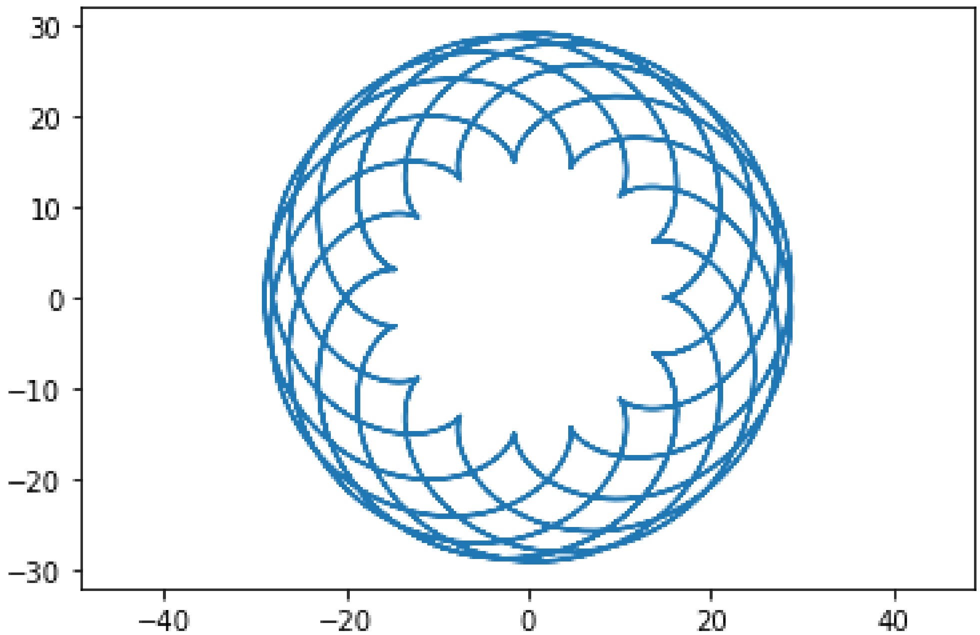
I've chosen $R = 15$ and $r = 7$.

(Note that I've taken t to range from 0 to 30π, since the range of the parameter is given by:

$$[0, 2\pi \times \frac{LCM(r,R)}{r}] = [0, 30\pi]$$

[10]:
```
plt.axis('equal')
R = 15
r = 7
u = np.arange(0,30*math.pi+0.01,0.01)
plt.plot(list(map(lambda x: (R+r)*math.cos(x)-r*math.cos((R+r)/r *␣
 ↪x),u)),list(map(lambda x: (R+r)*math.sin(x)-r*math.sin((R+r)/r * x),u)))
```

[10]: [<matplotlib.lines.Line2D at 0x7f0d4b408da0>]


30
20
10
0
−10
−20
−30
−40
−20
0
20
40