Envelope Finder

October 29, 2019

1 Homework 7 - Q5

Now, we're going to try to apply the finite difference method and plot some stuff to show that the envelope of tangents drawn at each point is going to give us the original function's outline.

This time round, we're going to use only `matplotlib`, as the maths is simple enough to be done without the use of any other packages.

[1]:
```
import matplotlib.pyplot as plt
```

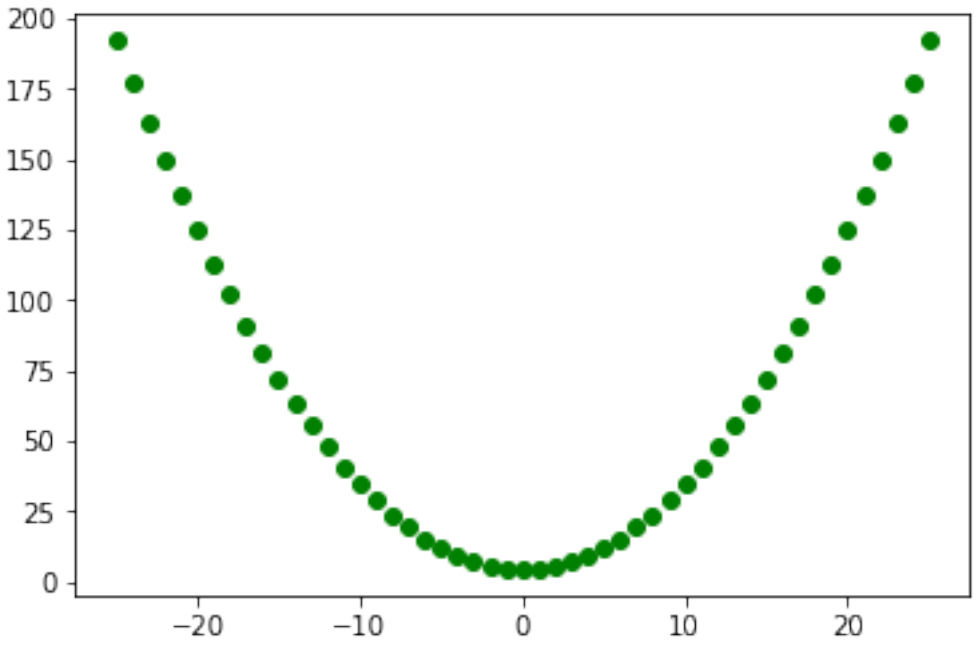
Generating our list of points to be from $x \in (-25, 25)$ to the corresponding y's such that: $y = 0.3x^2 + 4.5$

[2]:
```
pts=[list(range(-25,26)),list(map(lambda t:0.3*t*t+4.5,list(range(-25,26))))]
```

Now, plotting the x and y points as green filled circles

[3]:
```
plt.plot(pts[0],pts[1],'go')
```

[3]:
```
[<matplotlib.lines.Line2D at 0x7f16e59d8780>]
```



1.1 Using the Finite Difference Method

In principle:

$$f'(x) = \lim_{\Delta x \to 0} \frac{f(x + \Delta x) - f(x - \Delta x)}{2\Delta x}$$

Hence, we can say that, approximately:

$$f'(x) = \frac{f(x + 1) - f(x - 1)}{2}$$

Now, what we're doing is we're evaluating the value of f at two points, the one ahead and the one behind the point at which we want to calculate the derivative. Now, we'll find these values and store them in a list of values (basically, this becomes our m array).

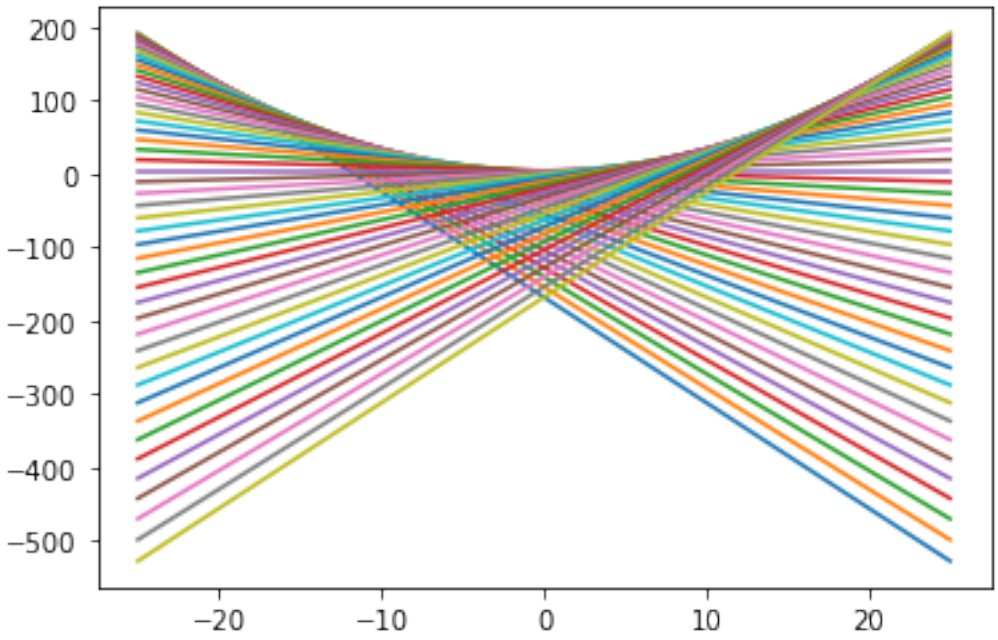
Next, we use some simple coordinate geometry to find the value of c for a given m, using the concept of a straight line. This is done taking the fact that $y = mx + c$, and hence $c = y - mx$.

Thus, at each point, we can evaluate the c_i value by taking the value of $y_i - m_i x_i$

[4]:
```
mc = [[],[]]
for i in range(1,50):
    m = ((pts[1][i+1]-pts[1][i-1])/(2))
    mc[0].append(m)
    mc[1].append(pts[1][i]-m*pts[0][i])
```

Now, plotting all the lines we've just created over the same x-array, again, using the `map` and `lambda` functions we have in Python.

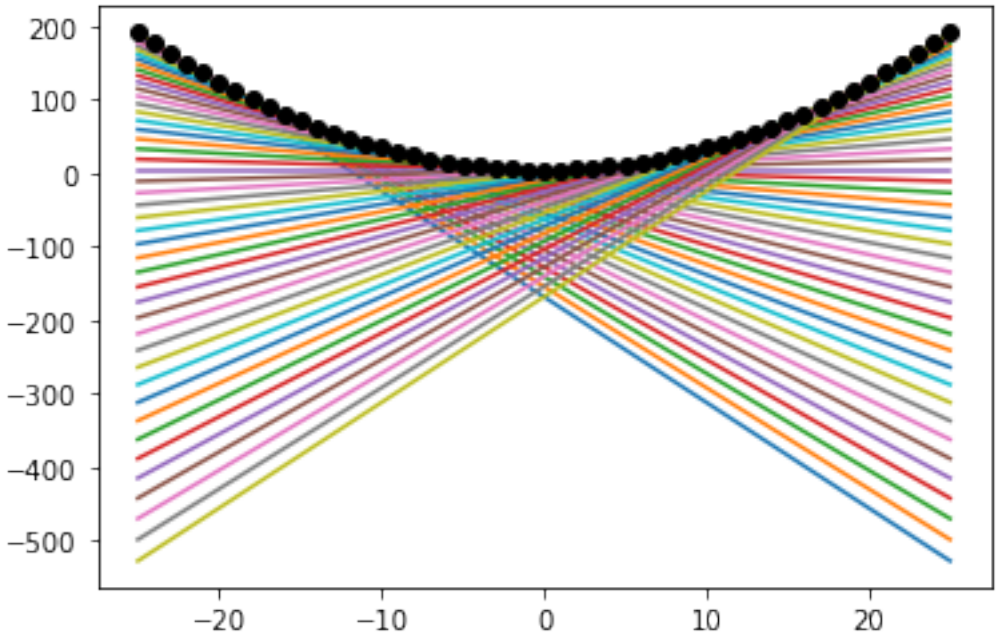
[5]:
```
for i in range(len(mc[0])):
    plt.plot(pts[0],list(map(lambda t: mc[0][i]*t+mc[1][i],pts[0])))
```



Now, superposing the above plot with the original plot of x and y to show that it's actually the envelope of the function we have chosen.

[6]:
```
for i in range(len(mc[0])):
    plt.plot(pts[0],list(map(lambda t: mc[0][i]*t+mc[1][i],pts[0])))
plt.plot(pts[0],pts[1],'o',color='black')
```

[6]:
```
[<matplotlib.lines.Line2D at 0x7f16e5844a20>]
```



Just showing the points (x, y) taken:

[7]:
```
for i in range(len(pts[0])):
    print(pts[0][i],",",pts[1][i])
```

```
-25 , 192.0
-24 , 177.29999999999998
-23 , 163.2
-22 , 149.7
-21 , 136.79999999999998
-20 , 124.5
-19 , 112.8
-18 , 101.69999999999999
-17 , 91.19999999999999
-16 , 81.3
-15 , 72.0
-14 , 63.300000000000004
-13 , 55.199999999999996
-12 , 47.699999999999996
-11 , 40.8
-10 , 34.5
-9 , 28.799999999999997
-8 , 23.7
-7 , 19.200000000000003
-6 , 15.299999999999999
-5 , 12.0
```

```
-4 , 9.3
-3 , 7.199999999999999
-2 , 5.7
-1 , 4.8
0 , 4.5
1 , 4.8
2 , 5.7
3 , 7.199999999999999
4 , 9.3
5 , 12.0
6 , 15.299999999999999
7 , 19.200000000000003
8 , 23.7
9 , 28.799999999999997
10 , 34.5
11 , 40.8
12 , 47.699999999999996
13 , 55.199999999999996
14 , 63.300000000000004
15 , 72.0
16 , 81.3
17 , 91.19999999999999
18 , 101.69999999999999
19 , 112.8
20 , 124.5
21 , 136.79999999999998
22 , 149.7
23 , 163.2
24 , 177.29999999999998
25 , 192.0
```

Now, showing the values of (m, c) for each point (x, y) (except the first and last ones, as their slope can't be defined without other information).

[8]:
```
for i in range(len(mc[0])):
    print(mc[0][i],",",mc[1][i])
```

```
-14.400000000000006 , -168.30000000000015
-13.799999999999997 , -154.19999999999993
-13.200000000000003 , -140.7000000000001
-12.599999999999994 , -127.79999999999993
-11.999999999999993 , -115.49999999999986
-11.400000000000006 , -103.80000000000011
-10.800000000000004 , -92.7000000000001
-10.199999999999996 , -82.19999999999993
-9.599999999999994 , -72.29999999999991
-8.999999999999996 , -62.99999999999994
-8.400000000000002 , -54.30000000000002
```

-7.800000000000004 , -46.20000000000007
-7.199999999999999 , -38.699999999999996
-6.599999999999998 , -31.799999999999983
-6.0 , -25.5
-5.4 , -19.800000000000004
-4.799999999999997 , -14.699999999999978
-4.2 , -10.2
-3.6000000000000014 , -6.30000000000001
-2.999999999999999 , -2.9999999999999964
-2.4000000000000004 , -0.3000000000000007
-1.8000000000000003 , 1.799999999999999
-1.1999999999999997 , 3.3000000000000007
-0.6000000000000001 , 4.199999999999999
0.0 , 4.5
0.6000000000000001 , 4.199999999999999
1.1999999999999997 , 3.3000000000000007
1.8000000000000003 , 1.799999999999999
2.4000000000000004 , -0.3000000000000007
2.999999999999999 , -2.9999999999999964
3.6000000000000014 , -6.30000000000001
4.2 , -10.2
4.799999999999997 , -14.699999999999978
5.4 , -19.800000000000004
6.0 , -25.5
6.599999999999998 , -31.799999999999983
7.199999999999999 , -38.699999999999996
7.800000000000004 , -46.20000000000007
8.400000000000002 , -54.30000000000002
8.999999999999996 , -62.99999999999994
9.599999999999994 , -72.29999999999991
10.199999999999996 , -82.19999999999993
10.800000000000004 , -92.7000000000001
11.400000000000006 , -103.80000000000011
11.999999999999993 , -115.49999999999986
12.599999999999994 , -127.79999999999993
13.200000000000003 , -140.7000000000001
13.799999999999997 , -154.19999999999993
14.400000000000006 , -168.30000000000015