Assignment02

March 20, 2019

Name : Kang Yeongeun

Student No. : 20151532

Github : https://github.com/yeonun/MLAssignment/Assignment02

1 Import packages numpy for calculating and matplotlib for drawing graph

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

2 Define a differentiable function that maps from real number to real number.

2.1 Define $f(x) = x * cosx$ as func(x)

```
In [2]: def func(x):
            f = np.cos(x)*x
            return f
```

3 Define a domain of the function.

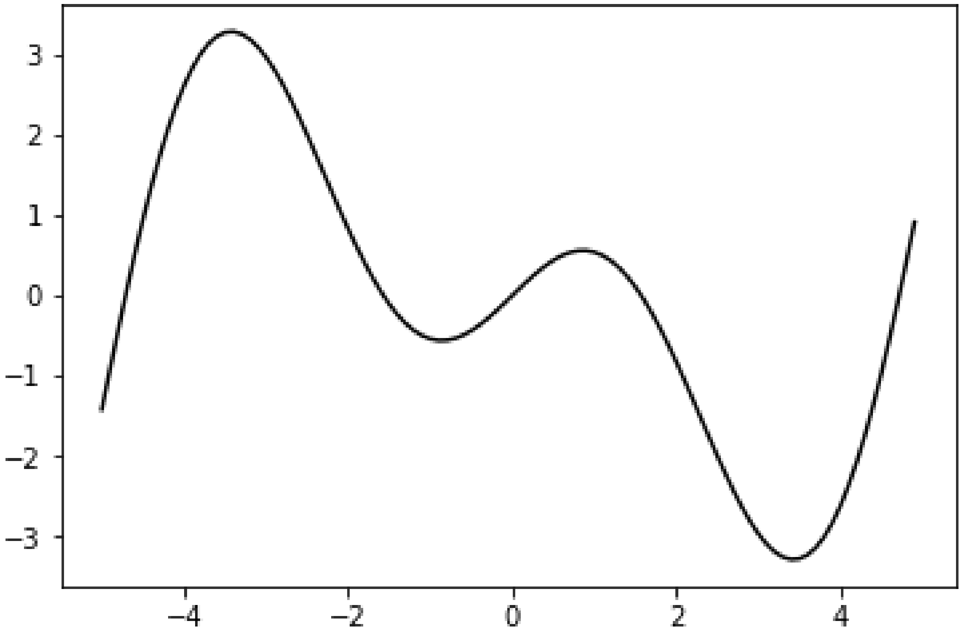
3.1 Domain : $ -5 < x < 5 $

```
In [3]: x = np.arange(-5,5,0.1)
```

4 Plot the function.

```
In [4]: f = func(x)
        plt.figure(1)
        plt.plot(x,f,'k',label="function")
```

```
Out[4]: [<matplotlib.lines.Line2D at 0x18493f332e8>]
```



5 Select a point within the domain

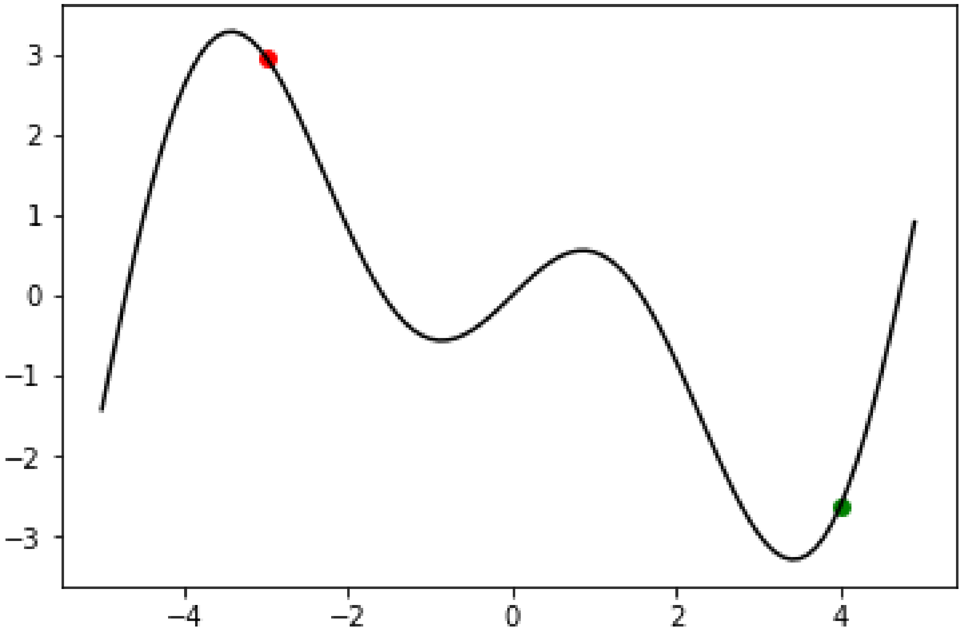
5.1 point x : -3,4

```
In [5]: p1 = -3
        p2 = 4
```

6 Mark the selected point on the function

```
In [6]: def origin_plot():
            y1 = func(p1)
            y2 = func(p2)
            plt.scatter(p1,y1,c='r',s=35)
            plt.scatter(p2,y2,c='g',s=35)
            plt.plot(x,f,'k',label="function")

        origin_plot()
```



7 Define the first-order Taylor approximation at the selected point

7.1 Define $f'(x) = -x * sinx + cosx$ as derivate function d_func(x)

```
In [7]: def d_func(x):
            df = - x*np.sin(x) +np.cos(x)
            return df
```

7.2 Define Tylor Approximation $f(a) + f'(a)(x - a)$

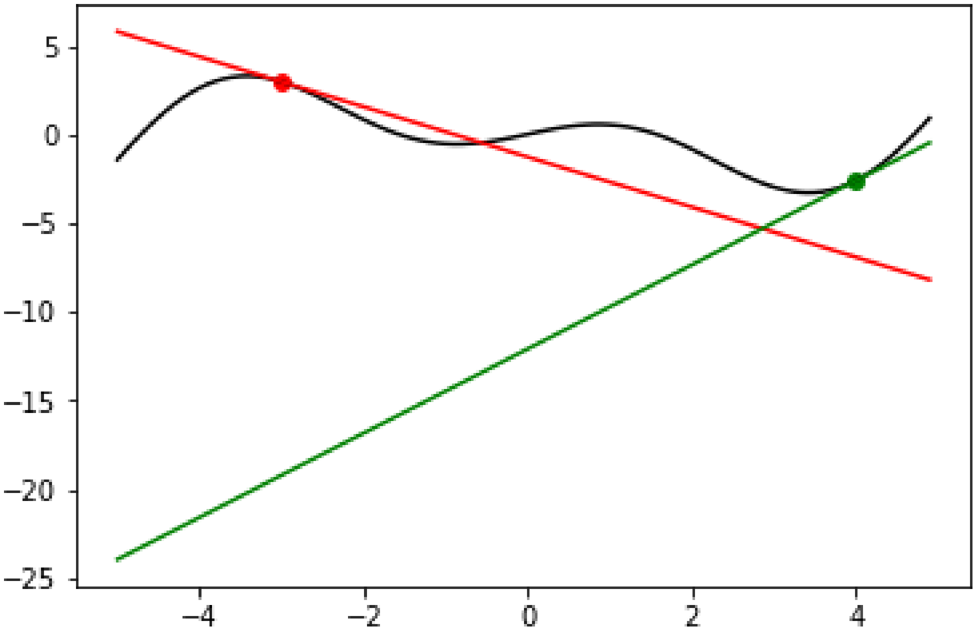
```
In [8]: def tylor(a,x):
            result = func(a) + d_func(a)*(x-a)
            return result
```

8 Plot the Taylor approximation with the same domain of the original function.

```
In [9]: origin_plot()

        df1 = tylor(p1,x)
        df2 = tylor(p2,x)
        plt.plot(x, df1, 'r')
        plt.plot(x, df2, 'g')
```

```
Out[9]: [<matplotlib.lines.Line2D at 0x18493d59048>]
```


5
0
−5
−10
−15
−20
−25
−4
−2
0
2
4