assignment03_2

March 26, 2021

1 Taylor approximation

1.1 import library

[113]:
```
%matplotlib inline
import numpy as np
import matplotlib.pyplot as plt
```

1.2 define my function $f(x)$

[114]:
```
def myfunction(x):

    y = 4 * x**3 + 7* x**2 + 3*x + 1
    return y
```

```
git commit -a -m "define my function"
git push origin master
```

1.3 define derivative of my function $f'(x)$

[115]:
```
def derivative_myfunction(x):

    h = np.exp(-10)
    y_prime = ( myfunction(x + h) - myfunction(x) ) / h

    return y_prime
```

[116]:
```
derivative_myfunction(1.0)
```

[116]:
```
29.000862606924855
```

```
git commit -a -m "define derivative of my function"
git push origin master
```

1.4 define 1st order Taylor approxation of my function $\hat{f}(x) = f(a) + f'(a)(x - a)$

[117]:
```
def taylor(x, a):

    y_approximate = myfunction(a) + derivative_myfunction(a) * (x - a)

    return y_approximate
```

```
git commit -a -m "define Taylor approximation"
git push origin master
```

1.5 define functions for the visualization

[118]:
```
x = np.linspace(-10 , 10, num = 50)
y = myfunction(x)

a = 0.1
b = myfunction(a)

t = taylor(x, a)

def plot_myfunction(x, y):

    plt.plot( x, y, 'b')
    plt.xlim([-10 , 10])
    plt.ylim([-10 , 10])

    plt.show()

def plot_myfunction_and_taylor(x, y, t, a, b):

    plt.plot( x, y, 'b')
    plt.plot( x, t, 'r')
    plt.plot( a, myfunction(a), 'go')
    plt.xlim([-10 , 10])
    plt.ylim([ -10, 10])

    plt.show()
```

```
git commit -a -m "define functions for the visualization"
git push origin master
```

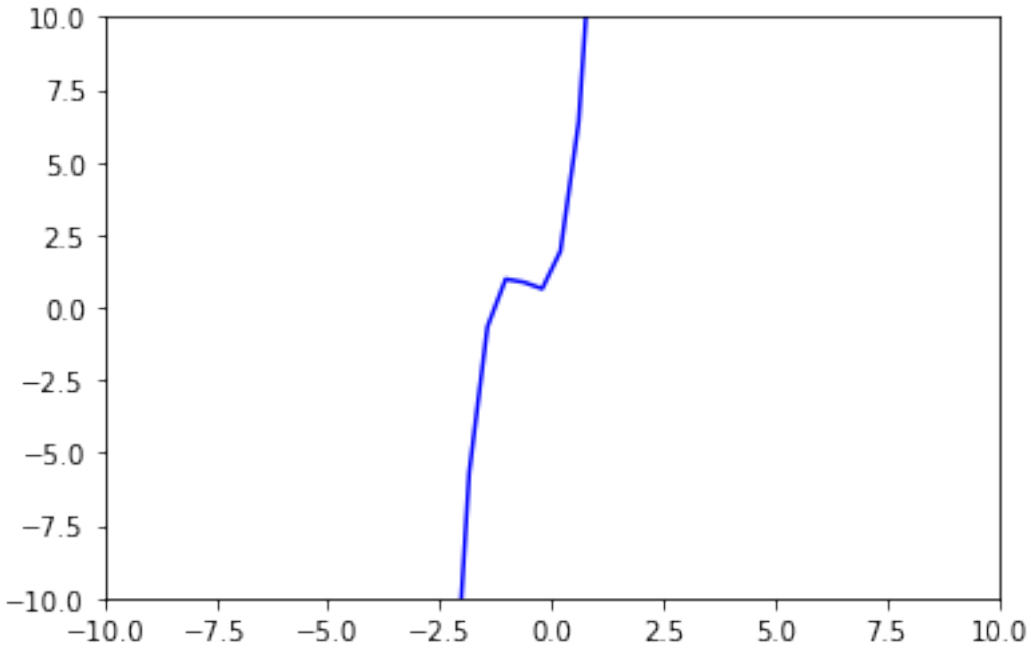
2

3 # results

4

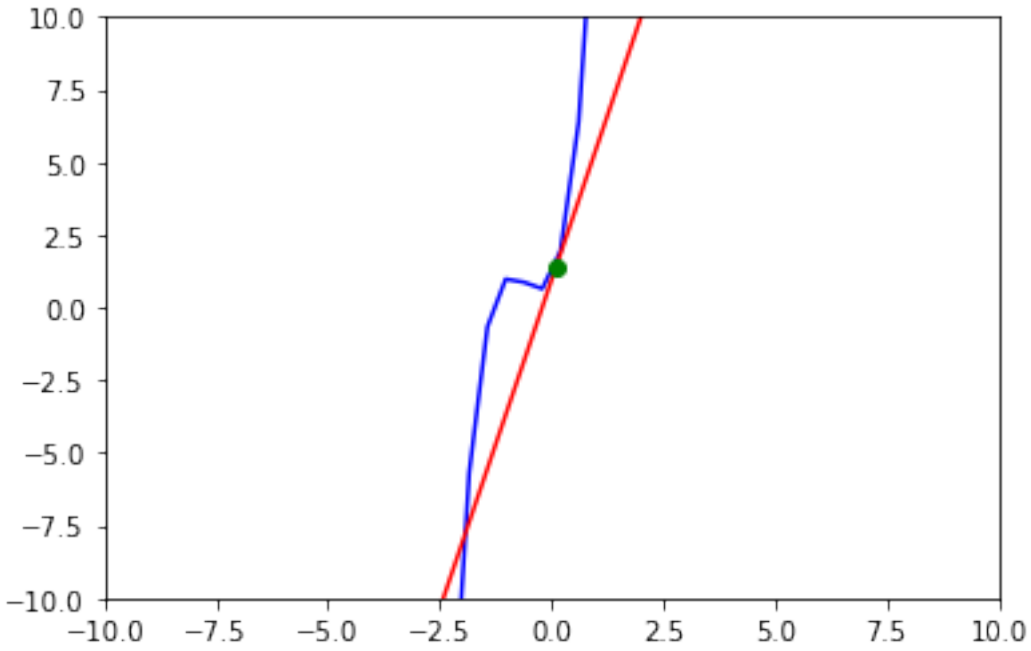
5 # 01. plot my function $f(x)$

[119]:
```
plot_myfunction(x, y)
```



6 # 02. plot my function $f(x)$ & Taylor approxation $\hat{f}(x)$

[120]:
```
plot_myfunction_and_taylor(x, y, t, a, b)
```


10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−10.0
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
10.0

[ ]: