202001555 지은미

In [2]:
```
from IPython.core.interactiveshell import InteractiveShell
InteractiveShell.ast_node_interactivity="all"

from collections import Counter

def squared_distance(v,w):
    return sum_of_squares(vector_subtract(v,w))

def distance(v,w):
    return math.sqrt(squred_distance(v,w))

def vector_subtract(v,w):
    """subtract two vectors componentwise"""
    return [v_i+w_i for v_i,w_i in zip(v,w)]

def scalar_multiply(v,w):
    return [c*v_i for v_i in v]

def vector_add(v,w):
    """adds two vectors componentwise"""
    return [v_i+w_i for v_i,w_i in zip(v,w)]

def vector_sum(vectors):
    return reduce(vector_add, vectors)

def vector_mean(vectors):
    """compute the vector whose i-th element is the mean of the mean of the i-th eleme
    n = len(vectors)
    return scalar_multiply(1/n, vector_sum(vectors))
```

In [3]:
```
from functools import reduce
import math, random

import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline

def sum_of_squares(v):
    """computes the sum of squared elements in v"""
    return sum(v_i ** 2 for v_i in v)

vector = [i for i in range(10)]
sum_of_squares(vector)

np.sum(np.square(vector))
```

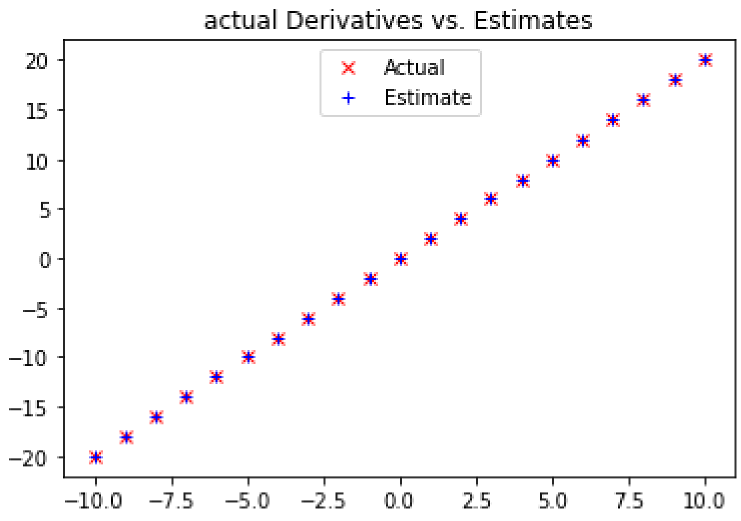
Out[3]:
```
285
```

Out[3]:
```
285
```

In [4]:
```
def difference_quotient(f, x, h):
    return (f(x + h) - f(x)) / h

def square(x: float) -> float:
    return x * x

def derivative(x: float) -> float:
    return 2 * x

xs = range(-10,11)
actuals = [derivative(x) for x in xs]
```

```
estimates = [difference_quotient(square, x, h=0.0001) for x in xs]

# 두 계산식의 결괏값이 거의 비슷함을 보여 주기 위한 그래프
import matplotlib.pyplot as plt
plt.title("actual Derivatives vs. Estimates")
plt.plot(xs, actuals, 'rx', label='Actual')
plt.plot(xs, estimates, 'b+', label='Estimate')
plt.legend(loc=9)
plt.show()
```

```
Out[4]: Text(0.5, 1.0, 'actual Derivatives vs. Estimates')

Out[4]: [<matplotlib.lines.Line2D at 0x1e24a702e80>]

Out[4]: [<matplotlib.lines.Line2D at 0x1e24a970220>]

Out[4]: <matplotlib.legend.Legend at 0x1e24a970400>
```



202001555 지은미