Gradient Descent programming

Gradient Descent

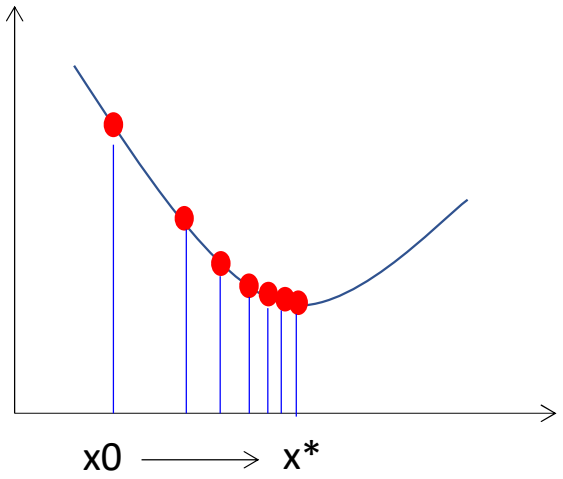
- Optimization solver

1. Initialize x_0

2. $x_{k+1} = x_k - \alpha \frac{df}{dx}$

3. Repeat

Step size *Direction*



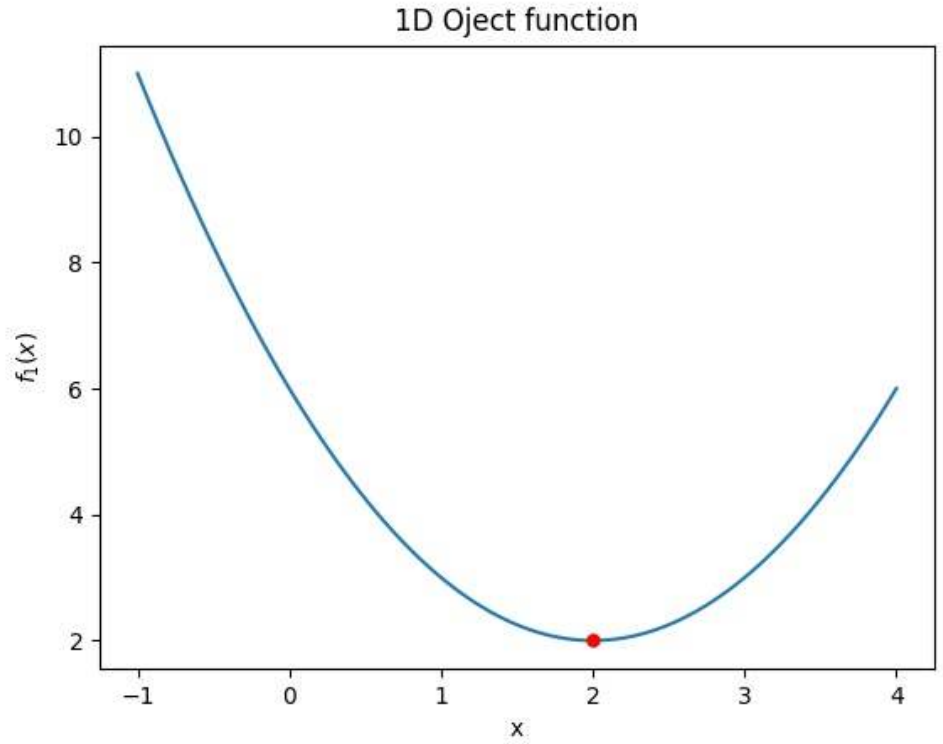
1D Object function

```python
import numpy as np
import matplotlib.pyplot as plt

def f1(x):
    return (x-2)**2 + 2

xx = np.linspace(-1, 4, 100)

plt.plot(xx, f1(xx))
plt.plot(2, 2, 'ro', markersize = 5)
plt.xlabel('x')
plt.ylabel('$f_1(x)$')
plt.title('1D Oject function')
plt.show()
```



Gradient Descent

```python
import numpy as np
import matplotlib.pyplot as plt

def f1(x):
    return (x-2)**2 + 2

def f1d(x):
    #f1'(x) : derevartive function
    return 2 * (x - 2.0)

xx = np.linspace(-1, 4, 100)
plt.plot(xx, f1(xx), 'k-')

# step size
mu = 0.3
```

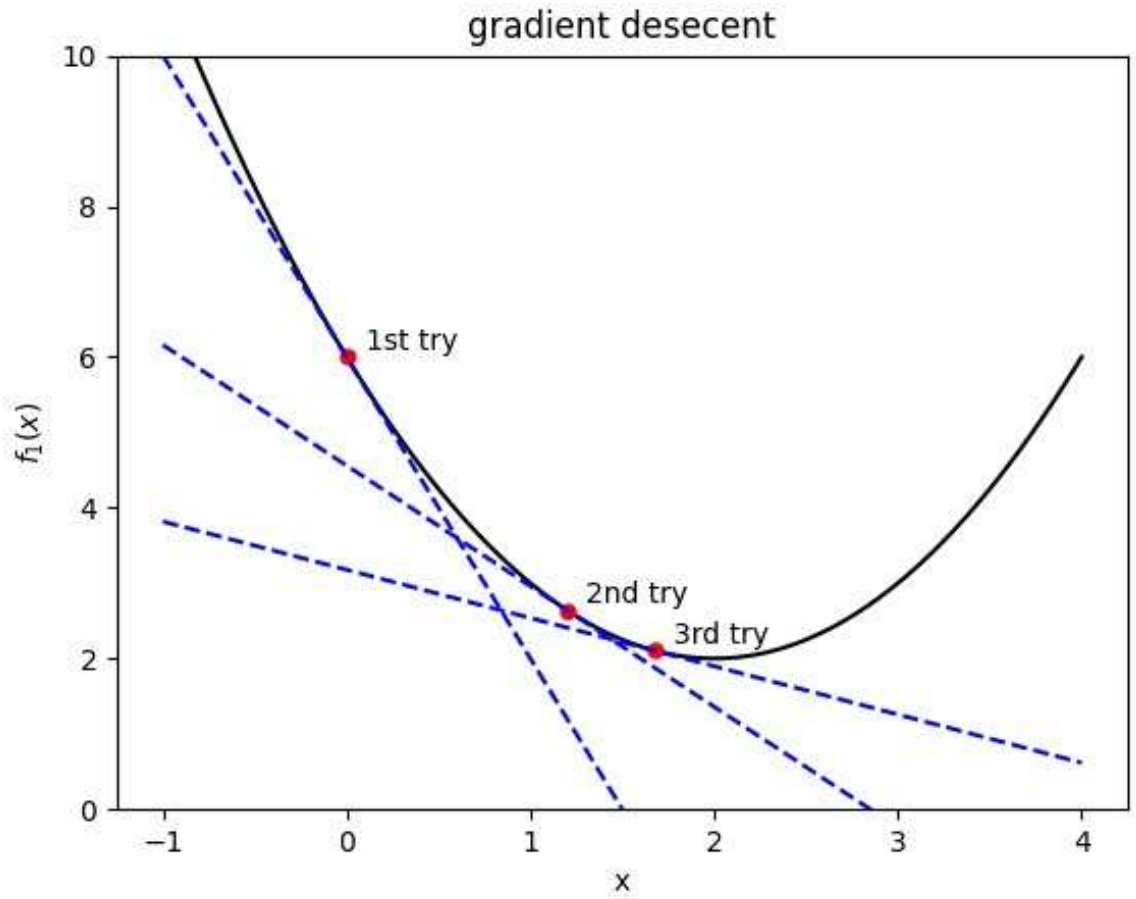
```python
x = 0
plt.plot(x, f1(x), 'go', markersize=5)
plt.text(x + 0.1, f1(x)+0.1, '1st try')
plt.plot(xx, f1d(x) * (xx - x) + f1(x), 'b--')
print('1st try: x_2 = {:.2f}, g_2 = {:.2f}'.format(x, f1d(x)))

x = x - mu*f1d(x)
plt.plot(x, f1(x), 'go', markersize=5)
plt.text(x + 0.1, f1(x)+0.1, '2nd try')
plt.plot(xx, f1d(x) * (xx - x) + f1(x), 'b--')
print('2nd try: x_2 = {:.2f}, g_2 = {:.2f}'.format(x, f1d(x)))

x = x - mu*f1d(x)
plt.plot(x, f1(x), 'go', markersize=5)
plt.text(x + 0.1, f1(x)+0.1, '3rd try')
plt.plot(xx, f1d(x) * (xx - x) + f1(x), 'b--')
print('3rd try: x_2 = {:.2f}, g_2 = {:.2f}'.format(x, f1d(x)))

plt.xlabel('x')
plt.ylabel('$f_1(x)$')
plt.title('gradient desecent')
plt.ylim(0, 10)
plt.show()
```

Gradient Descent



E N D