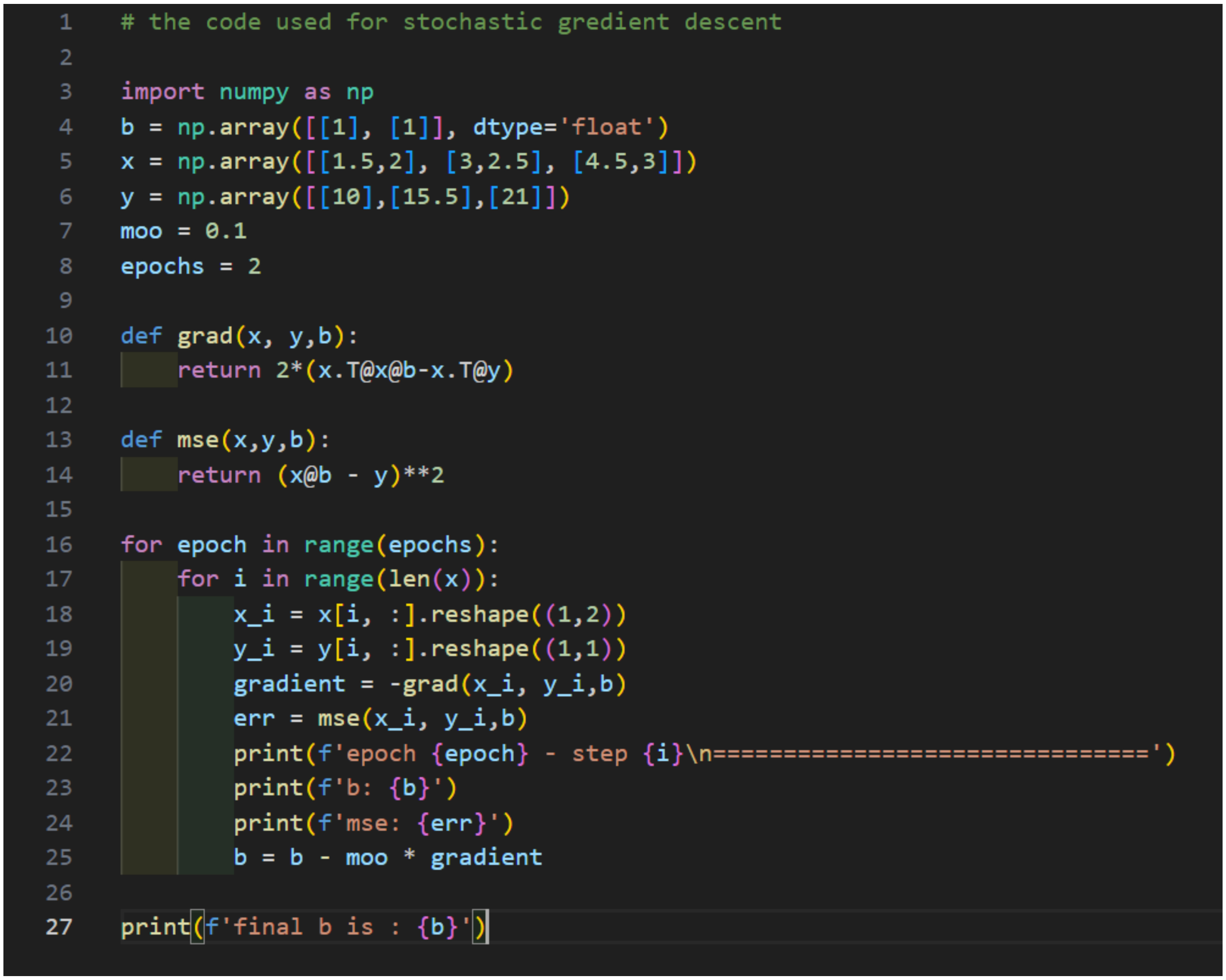
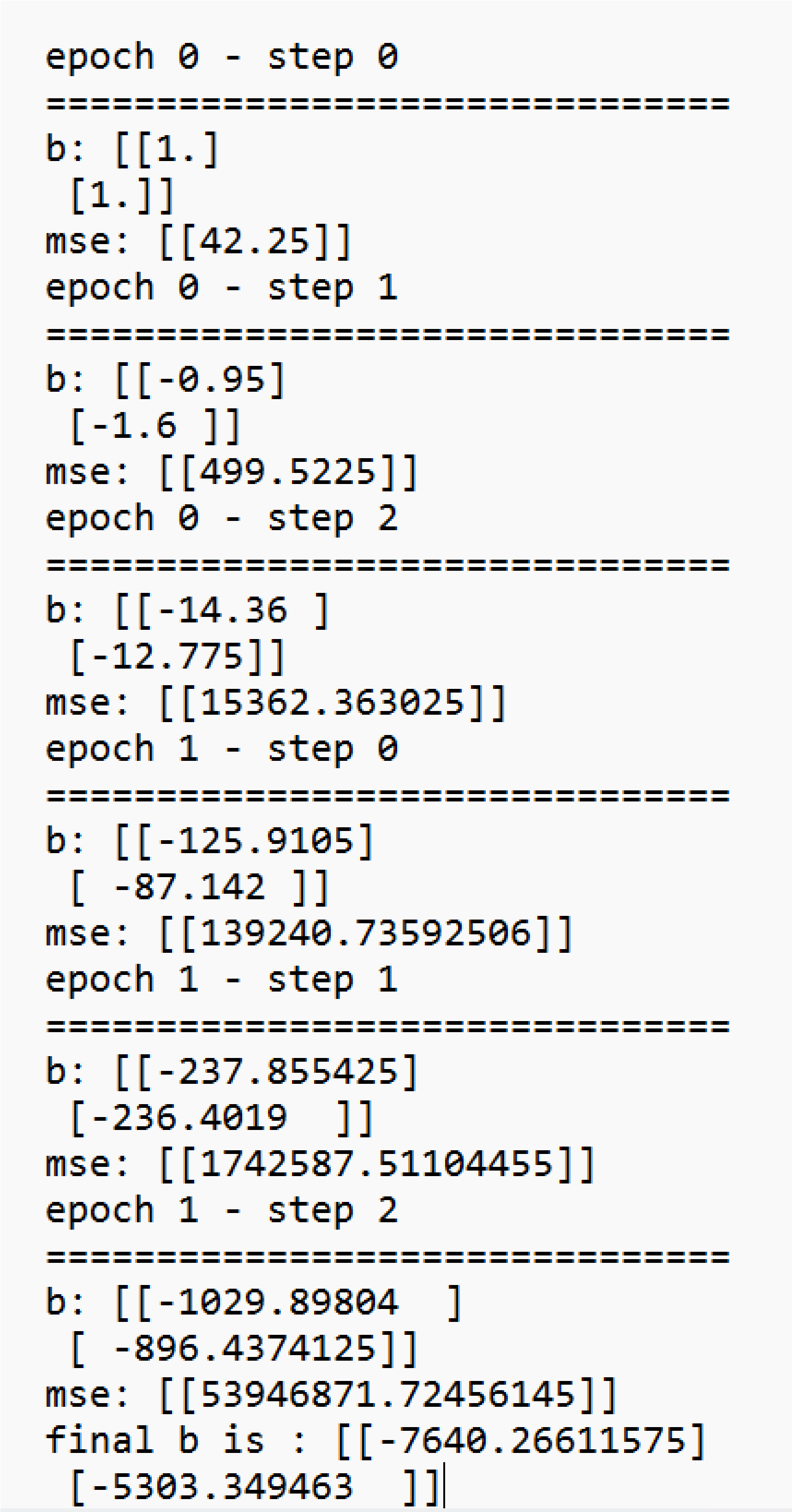
The code and the results (final and intermediate) for question 2, part b)

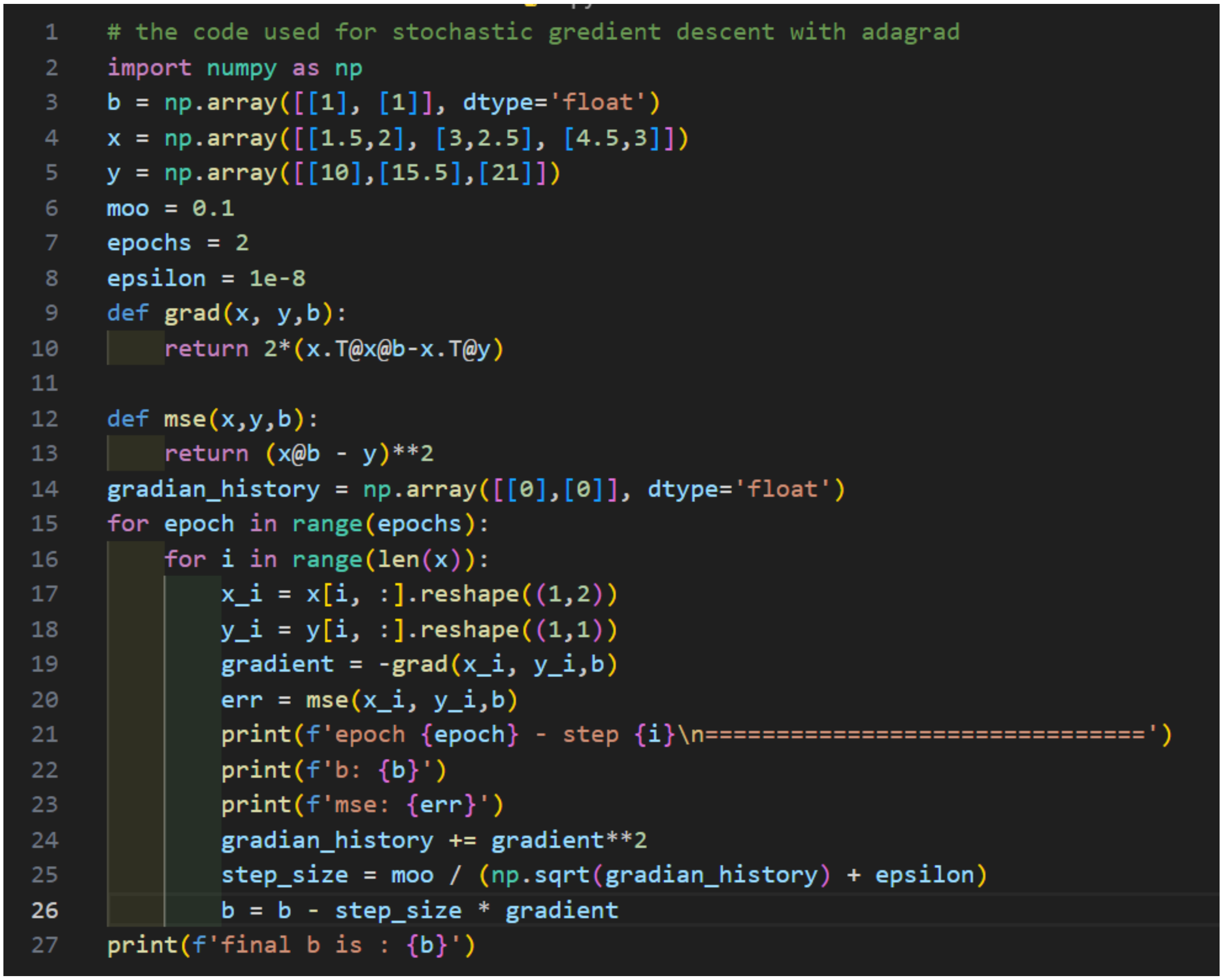


```python
# the code used for stochastic gredient descent

import numpy as np
b = np.array([[1], [1]], dtype='float')
x = np.array([[1.5,2], [3,2.5], [4.5,3]])
y = np.array([[10],[15.5],[21]])
moo = 0.1
epochs = 2

def grad(x, y,b):
    return 2*(x.T@x@b-x.T@y)

def mse(x,y,b):
    return (x@b - y)**2

for epoch in range(epochs):
    for i in range(len(x)):
        x_i = x[i, :].reshape((1,2))
        y_i = y[i, :].reshape((1,1))
        gradient = -grad(x_i, y_i,b)
        err = mse(x_i, y_i,b)
        print(f'epoch {epoch} - step {i}\n===============================')
        print(f'b: {b}')
        print(f'mse: {err}')
        b = b - moo * gradient

print(f'final b is : {b}')
```

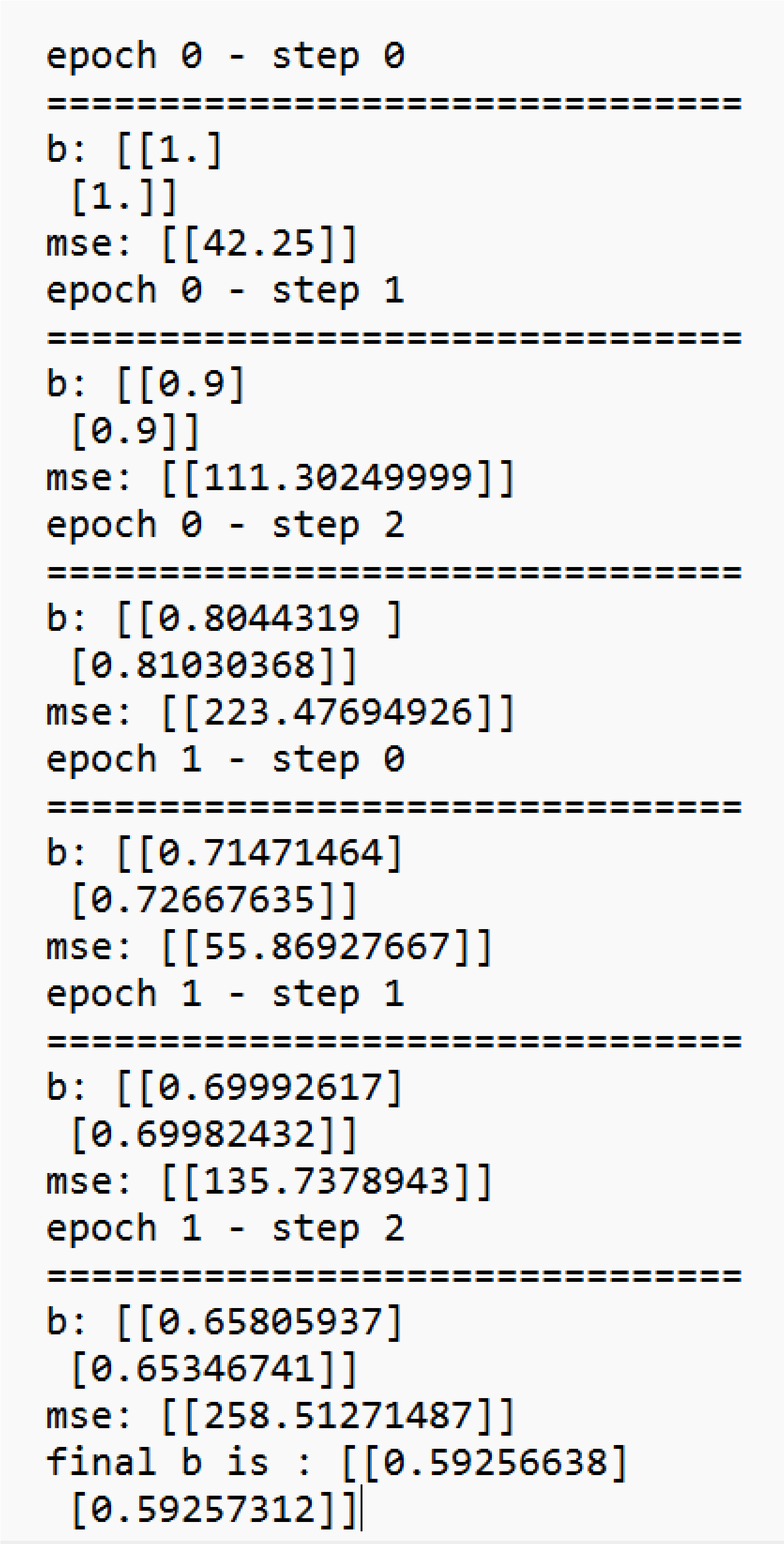


```
epoch 0 - step 0
===============================
b: [[1.]
 [1.]]
mse: [[42.25]]
epoch 0 - step 1
===============================
b: [[-0.95]
 [-1.6 ]]
mse: [[499.5225]]
epoch 0 - step 2
===============================
b: [[-14.36 ]
 [-12.775]]
mse: [[15362.363025]]
epoch 1 - step 0
===============================
b: [[-125.9105]
 [ -87.142 ]]
mse: [[139240.73592506]]
epoch 1 - step 1
===============================
b: [[-237.855425]
 [-236.4019  ]]
mse: [[1742587.51104455]]
epoch 1 - step 2
===============================
b: [[-1029.89804   ]
 [ -896.4374125]]
mse: [[53946871.72456145]]
final b is : [[-7640.26611575]
 [-5303.349463  ]]
```

The code and the results (final and intermediate) for question 2, part c)



```python
# the code used for stochastic gredient descent with adagrad
import numpy as np
b = np.array([[1], [1]], dtype='float')
x = np.array([[1.5,2], [3,2.5], [4.5,3]])
y = np.array([[10],[15.5],[21]])
moo = 0.1
epochs = 2
epsilon = 1e-8
def grad(x, y,b):
    return 2*(x.T@x@b-x.T@y)

def mse(x,y,b):
    return (x@b - y)**2
gradian_history = np.array([[0],[0]], dtype='float')
for epoch in range(epochs):
    for i in range(len(x)):
        x_i = x[i, :].reshape((1,2))
        y_i = y[i, :].reshape((1,1))
        gradient = -grad(x_i, y_i,b)
        err = mse(x_i, y_i,b)
        print(f'epoch {epoch} - step {i}\n===============================')
        print(f'b: {b}')
        print(f'mse: {err}')
        gradian_history += gradient**2
        step_size = moo / (np.sqrt(gradian_history) + epsilon)
        b = b - step_size * gradient
print(f'final b is : {b}')
```



```
epoch 0 - step 0
===============================
b: [[1.]
 [1.]]
mse: [[42.25]]
epoch 0 - step 1
===============================
b: [[0.9]
 [0.9]]
mse: [[111.30249999]]
epoch 0 - step 2
===============================
b: [[0.8044319 ]
 [0.81030368]]
mse: [[223.47694926]]
epoch 1 - step 0
===============================
b: [[0.71471464]
 [0.72667635]]
mse: [[55.86927667]]
epoch 1 - step 1
===============================
b: [[0.69992617]
 [0.69982432]]
mse: [[135.7378943]]
epoch 1 - step 2
===============================
b: [[0.65805937]
 [0.65346741]]
mse: [[258.51271487]]
final b is : [[0.59256638]
 [0.59257312]]
```