# ENG TECH 3PR3

Procedural and Object-Oriented Programming Concepts

Assignment 4

Submitted by: Qin Yang 400420584

Date: Feb 24th , 2022

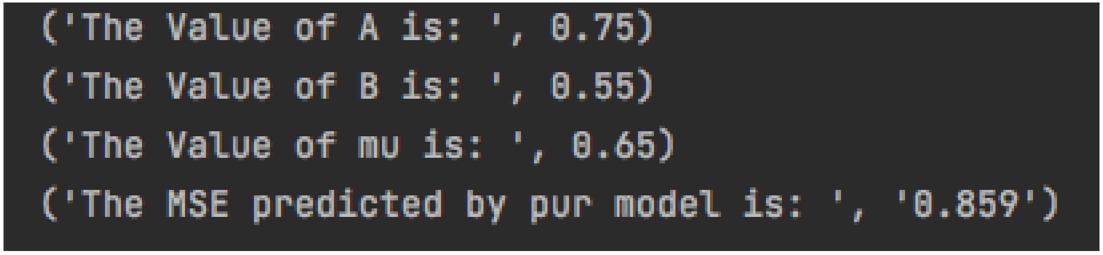
# Part. A Modelling Login Attempts

## 1. Code

```
import math
import matplotlib
matplotlib.use('TkAgg')
import matplotlib.pyplot as plt
import numpy as np


def get_inputs():  # getting file and insert value to an array
    file01 = open("/Users/qinyang/Desktop/A04_sfwr_data_01.txt", "r")
    s_file = []
    s_file.extend(file01.readlines())
    s_file = map(float, s_file)
    return s_file


def setparameters():  # set up parameters
    for a in np.arange(0, 2.01):
        a = format(a, ".2f")
    for b in np.arange(0, 2.01):
        b = format(b, ".2f")
    for mu in np.arange(0, 2.01):
        mu = format(mu, ".2f")

    return mu, a, b


def getfit():  # conduct brute force search to minimize mse to less than 1.0
    t_total = 120
    mse_trend = []
    mse_better = 1000
    s_file = get_inputs()
    i = 0  # define how many iterations we compared mse
    for a in np.arange(0, 2.01, 0.05):
        for b in np.arange(0, 2.01, 0.05):
            for mu in np.arange(0, 2.01, 0.05):
                if mse_better < 1:
                    break
                else:
                    mse_result = evaluatemode(a, b, mu, t_total, s_file)
                    if mse_result < mse_better:
                        i = i + 1
                        mse_better = mse_result
                        parameters_mse_min = [a, b, mu]
                        mse_trend.append(mse_better)

    return mse_better, parameters_mse_min, i, mse_trend


def evaluatemode(a, b, mu, t_total, s_file):  # calculate mse
    n = t_total
    s_calculated = []
    for t in range(1, 121):
        s = (-1) * (a * math.sin(2 * math.pi * t / t_total) + mu) * math.e ** (b * 2 * math.pi * t /
t_total)
        s_calculated.append(s)
    s_subtract = np.subtract(s_calculated, s_file)
    s_square = np.square(s_subtract)
    s_sum = np.sum(s_square)
    mse = s_sum / n

    return mse

s_file = get_inputs()
setparameters()
mse_better, parameters_mse_min, i, mse_trend = getfit()
s_calculated = []
a = parameters_mse_min[0]
b = parameters_mse_min[1]
mu = parameters_mse_min[2]
T = 120
for t in range(1, 121):
    s = (-1) * (a * math.sin(2 * math.pi * t / T) + mu) * math.e ** (b * 2 * math.pi * t / T)
    print(s)
    s_calculated.append(s)

print('The Value of A is: ', parameters_mse_min[0])
print('The Value of B is: ', parameters_mse_min[1])
print('The Value of mu is: ', parameters_mse_min[2])
print('The MSE predicted by pur model is: ', format(mse_better, ".3f"))

t_x = np.arange(1, 121, 1)
plt.scatter(t_x, s_file)
plt.plot(t_x, s_calculated)
plt.xlabel('Time [s]')
plt.ylabel('No.of attempts')
plt.show()

i_x = np.arange(0, i, 1)
plt.plot(i_x, mse_trend)
plt.xlabel('iterations')
plt.ylabel('MSE')
plt.show()
```

## 2. Test Result

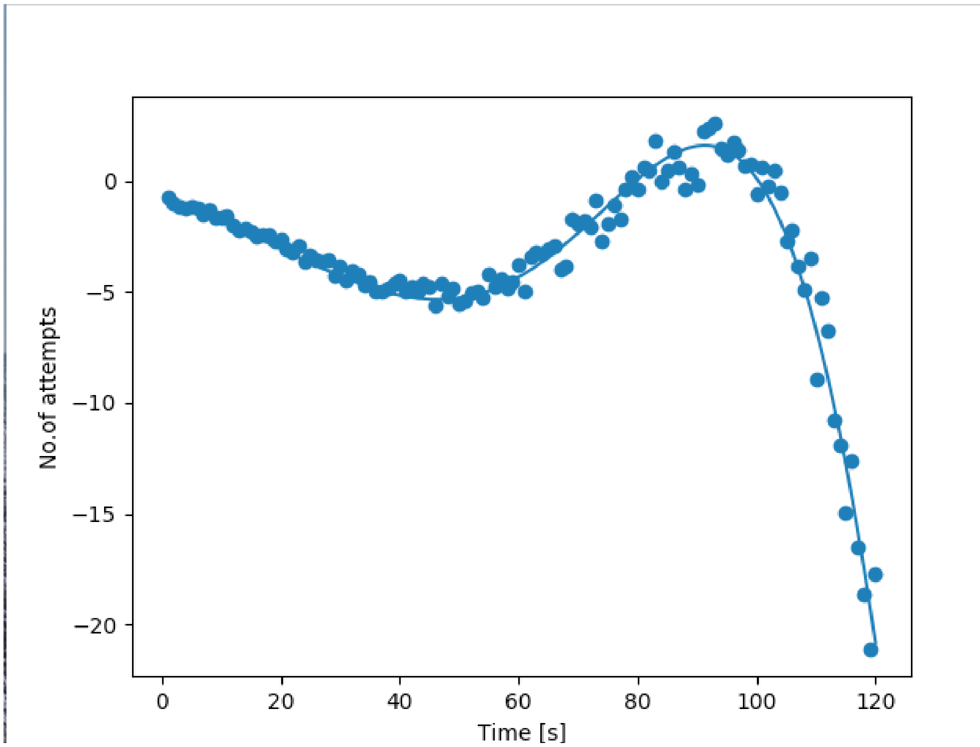
(1) Screenshot of print output



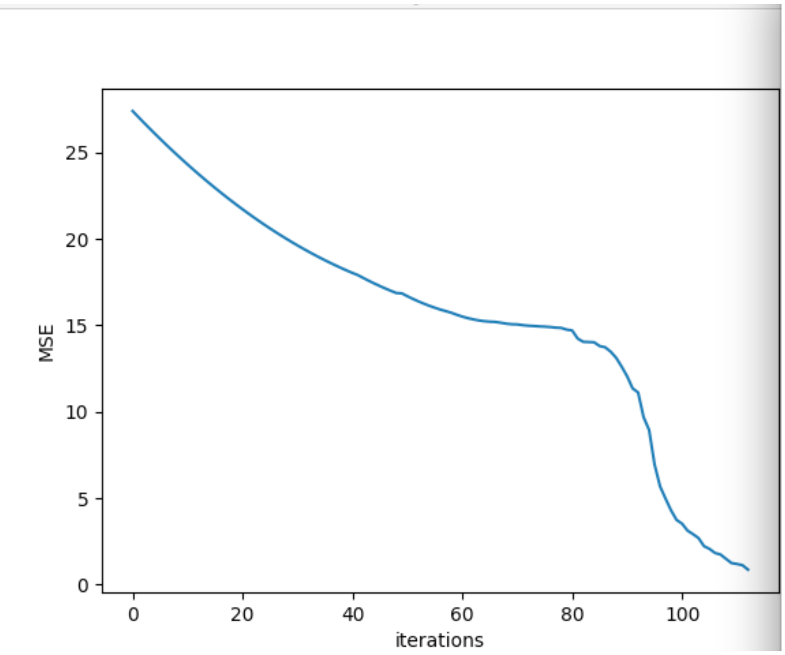
```
('The Value of A is: ', 0.75)
('The Value of B is: ', 0.55)
('The Value of mu is: ', 0.65)
('The MSE predicted by pur model is: ', '0.859')
```

(2) Screenshot of plot output

No. Attempts



MSE Trend:



## Part. B Some Variation is observed

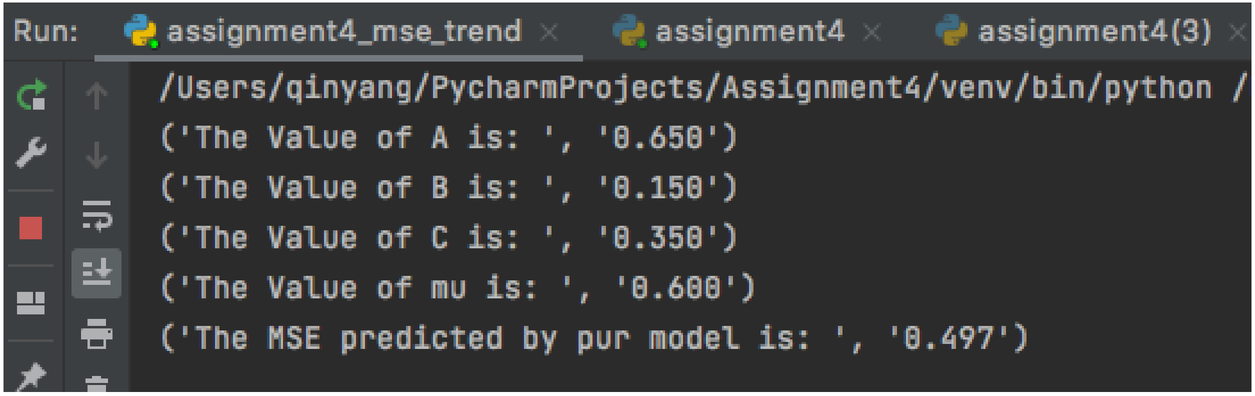
### 1. Code

```
import math
import matplotlib
matplotlib.use('TkAgg')
import matplotlib.pyplot as plt
import numpy as np


def get_inputs():  # getting file and insert value to an array
    file01 = open("/Users/qinyang/Desktop/A04_sfwr_data_03.txt", "r")
    s_file = []
    s_file.extend(file01.readlines())
    s_file = map(float, s_file)
    return s_file


def setparameters():  # set up parameters
    for a in np.arange(0, 2.01):
        a = format(a, ".2f")
    for b in np.arange(0, 2.01):
        b = format(b, ".2f")
    for c in np.arange(0.01, 2.01):
        c = format(c, ".2f")
    for mu in np.arange(0, 2.01):
        mu = format(mu, ".2f")
    return mu, a, b, c

def getfit():  # conduct brute force search to minimize mse to less than 1.0
    t_total = 120
    mse_trend = []
    mse_better = 1000
    s_file = get_inputs()
    i = 0  # define how many iterations we compared mse
    for a in np.arange(0, 2.01, 0.05):
        for b in np.arange(0, 2.01, 0.05):
            for mu in np.arange(0, 2.01, 0.05):
                for c in np.arange(0.05, 2.01, 0.05):
                    if mse_better < 0.5:
                        break
                    else:
                        mse_result = evaluatemode(a, b, c, mu, t_total, s_file)
                        if mse_result < mse_better:
                            i = i + 1
                            mse_better = mse_result
                            parameters_mse_min = [a, b, c, mu]
                            mse_trend.append(mse_better)

    return mse_better, parameters_mse_min, i, mse_trend


def evaluatemode(a, b, c, mu, t_total, s_file):  # calculate mse
    n = t_total
    s_calculated = []
    for t in range(1, 121):
        s = (-1) * (a * math.sin(2 * math.pi * t / t_total) + mu) * math.e ** (b * 2 * math.pi * t / (t_total * c))
        s_calculated.append(s)
    s_subtract = np.subtract(s_calculated, s_file)
    s_square = np.square(s_subtract)
    s_sum = np.sum(s_square)
    mse = s_sum / n

    return mse

s_file = get_inputs()
setparameters()
mse_better, parameters_mse_min, i, mse_trend = getfit()
s_calculated = []
a = parameters_mse_min[0]
b = parameters_mse_min[1]
c = parameters_mse_min[2]
mu = parameters_mse_min[3]
T = 120
for t in range(1, 121):
    s = (-1) * (a * math.sin(2 * math.pi * t / T) + mu) * math.e ** (b * 2 * math.pi * t / T)
    s_calculated.append(s)

print('The Value of A is: ', format(parameters_mse_min[0], ".3f"))
print('The Value of B is: ', format(parameters_mse_min[1], ".3f"))
print('The Value of C is: ', format(parameters_mse_min[2], ".3f"))
print('The Value of mu is: ', format(parameters_mse_min[3], ".3f"))
print('The MSE predicted by pur model is: ', format(mse_better, ".3f"))

t_x = np.arange(1, 121, 1)
plt.scatter(t_x, s_file)
plt.plot(t_x, s_calculated)
plt.xlabel('Time [s]')
plt.ylabel('No.of attempts')
plt.show()
```

```
i_x = np.arange(0, i, 1)
plt.plot(i_x, mse_trend)
plt.xlabel('iterations')
plt.ylabel('MSE')
plt.show()
```

## 2. Test Result

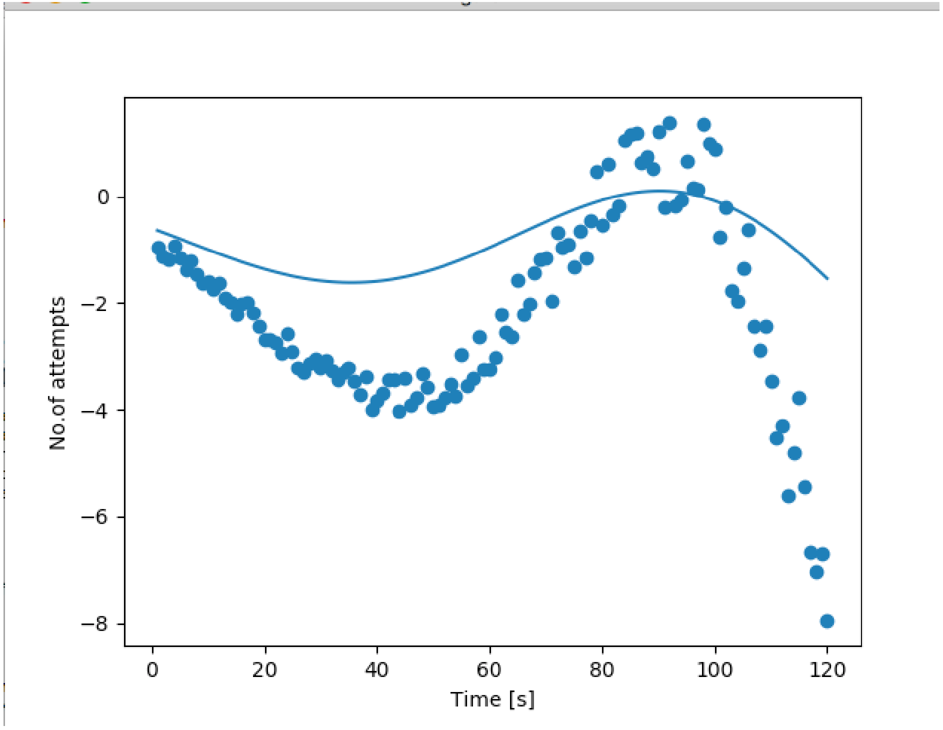
(1) Screenshot of print output



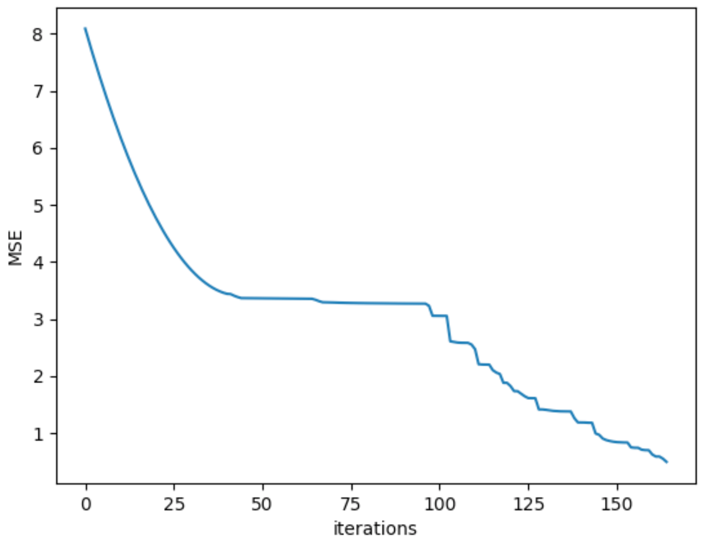
```
/Users/qinyang/PycharmProjects/Assignment4/venv/bin/python /
('The Value of A is: ', '0.650')
('The Value of B is: ', '0.150')
('The Value of C is: ', '0.350')
('The Value of mu is: ', '0.600')
('The MSE predicted by pur model is: ', '0.497')
```

(2) Screenshot of plot output

No. Attempts



MSE Trend:



## Part. C Towards a generic model

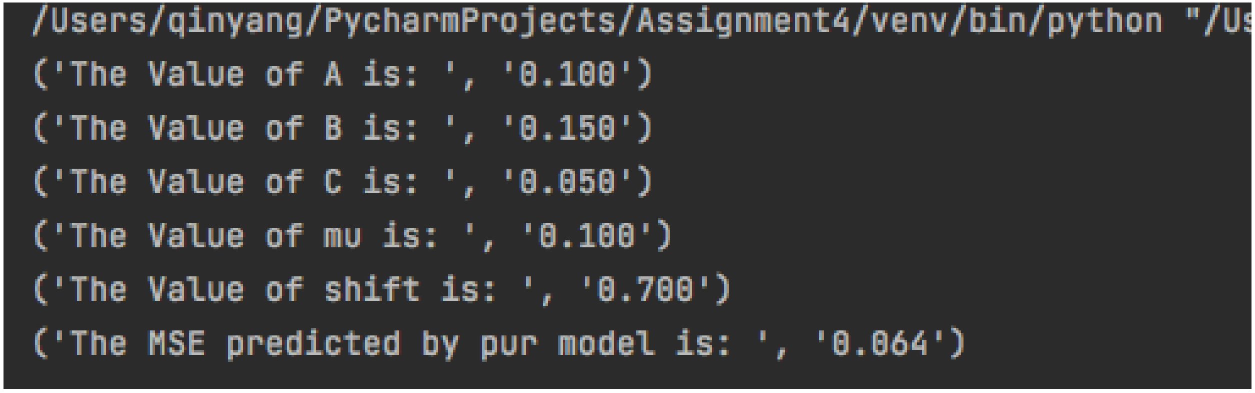
### 1. Code

```
import math
import matplotlib
matplotlib.use('TkAgg')
import matplotlib.pyplot as plt
import numpy as np


def get_inputs():  # getting file and insert value to an array
   file01 = open("/Users/qinyang/Desktop/A04_sfwr_data_05.txt", "r")
   s_file = []
   s_file.extend(file01.readlines())
   s_file = map(float, s_file)
   return s_file


def setparameters():  # set up parameters
   for a in np.arange(0, 2.01):
      a = format(a, ".2f")
```

```
    for b in np.arange(0, 2.01):
        b = format(b, ".2f")
    for c in np.arange(0.01, 2.01):
        c = format(c, ".2f")
    for mu in np.arange(0, 2.01):
        mu = format(mu, ".2f")
    for shift in np.arange(0, 1.51):
        shift = format(shift, ".2f")
    return mu, a, b, c, shift

def getfit():  # conduct brute force search to minimize mse to less than 1.0
    t_total = 120
    mse_trend = []
    mse_better = 1000
    s_file = get_inputs()
    i = 0  # define how many iterations we compared mse
    for a in np.arange(0, 2.01, 0.05):
        for b in np.arange(0, 2.01, 0.05):
            for mu in np.arange(0, 2.01, 0.05):
                for c in np.arange(0.05, 2.01, 0.05):
                    for shift in np.arange(0.05, 1.51, 0.05):
                        print(a, b, c, mu, mse_better)
                        if mse_better < 0.1:
                            break
                        else:
                            mse_result = evaluatemode(a, b, c, mu, t_total, s_file, shift)
                            if mse_result < mse_better:
                                i = i + 1
                                mse_better = mse_result
                                parameters_mse_min = [a, b, c, mu, shift]
                                mse_trend.append(mse_better)

    return mse_better, parameters_mse_min, i, mse_trend


def evaluatemode(a, b, c, mu, t_total, s_file, shift):  # calculate mse
    n = t_total
    s_calculated = []
    for t in range(1, 121):
        s1 = (-1) * ((a * math.sin(2 * math.pi * t + shift) / t_total) + mu)
        s2 = math.e ** (b * ((2 * math.pi * t / t_total) + shift / c))
        s = s1 * s2
        s_calculated.append(s)
    s_subtract = np.subtract(s_calculated, s_file)
    s_square = np.square(s_subtract)
    s_sum = np.sum(s_square)
    mse = s_sum / n

    return mse

s_file = get_inputs()
setparameters()
mse_better, parameters_mse_min, i, mse_trend = getfit()
s_calculated = []
a = parameters_mse_min[0]
b = parameters_mse_min[1]
c = parameters_mse_min[2]
mu = parameters_mse_min[3]
shift = parameters_mse_min[4]
T = 120
for t in range(1, 121):
    s1 = (-1) * ((a * math.sin(2 * math.pi * t + shift) / T ) + mu)
    s2 = math.e ** (b * ((2 * math.pi * t / T) + shift / c))
    s = s1 * s2
    s_calculated.append(s)
print('The Value of A is: ', format(parameters_mse_min[0], ".3f"))
print('The Value of B is: ', format(parameters_mse_min[1], ".3f"))
print('The Value of C is: ', format(parameters_mse_min[2], ".3f"))
print('The Value of mu is: ', format(parameters_mse_min[3], ".3f"))
print('The Value of shift is: ', parameters_mse_min[4])
print('The MSE predicted by pur model is: ', format(mse_better, ".3f"))

t_x = np.arange(1, 121, 1)
plt.scatter(t_x, s_file)
plt.plot(t_x, s_calculated)
plt.xlabel('Time [s]')
plt.ylabel('No.of attempts')
plt.show()

i_x = np.arange(0, i, 1)
plt.plot(i_x, mse_trend)
plt.xlabel('iterations')
plt.ylabel('MSE')
plt.show()
```

## 2. Test Result

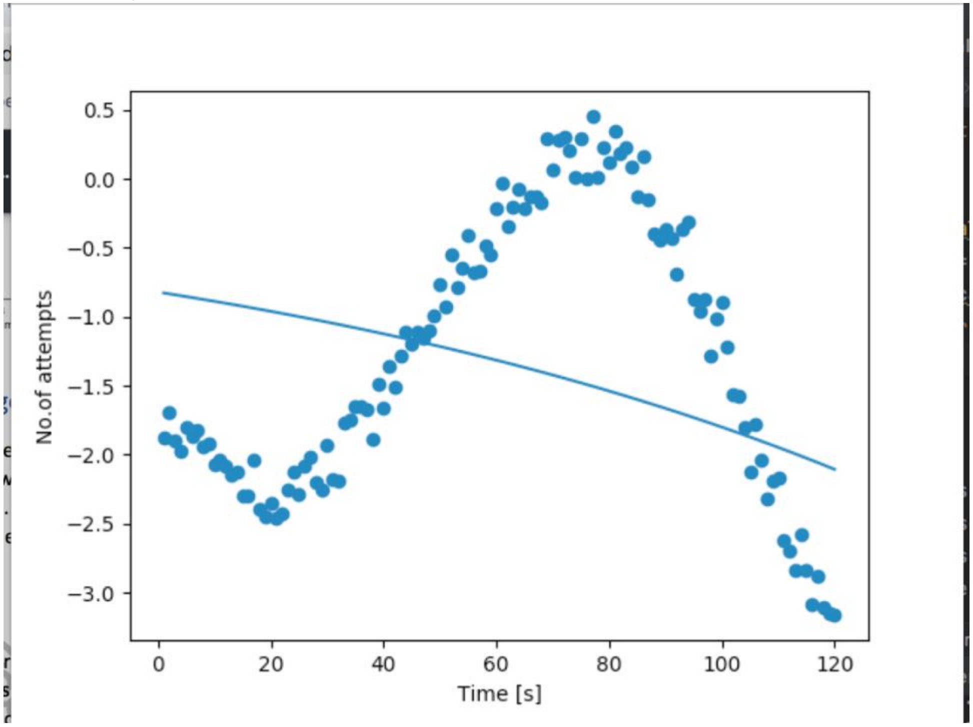
(1) Screenshot of print output



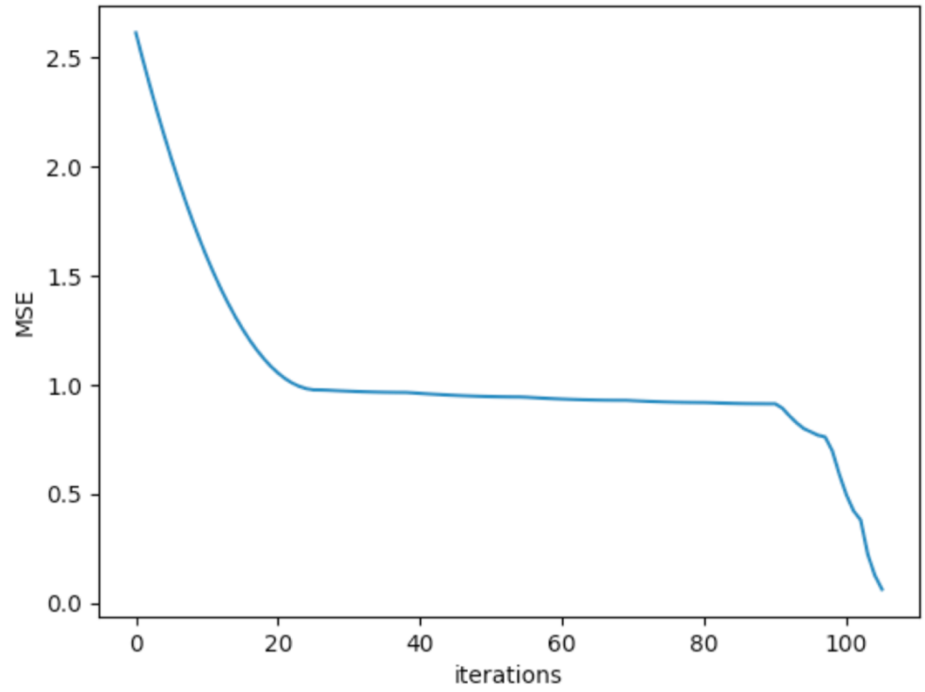
```
/Users/qinyang/PycharmProjects/Assignment4/venv/bin/python "/Us
('The Value of A is: ', '0.100')
('The Value of B is: ', '0.150')
('The Value of C is: ', '0.050')
('The Value of mu is: ', '0.100')
('The Value of shift is: ', '0.700')
('The MSE predicted by pur model is: ', '0.064')
```

(2) Screenshot of plot output

No Attempts:



MSE:


MSE
iterations