Student ID: 06996263 Student name: Fang Zhu

Q1:

Code:

import numpy as np

import matplotlib

import matplotlib.pyplot as plt

%matplotlib inline

import math

import textwrap

import numpy as np

import sklearn

from sklearn.model\_selection import KFold

upper = 1

lower = 0

Num = 100\*1

gVar = 0.1

split\_num = 4

scalar=1

step = (upper-lower)/(Num)

x = np.arange(lower+step/2,upper+step/2,step)

ntr = np.random.normal(0,gVar,Num-25\*scalar)

nte = np.random.normal(0,gVar,Num-75\*scalar)

m\_num = 10

kf = KFold(split\_num)

kf.get\_n\_splits(x)

print(kf)

E\_train=np.zeros(m\_num)

E\_test=np.zeros(m\_num)

M1=np.zeros(m\_num)

E=np.zeros(m\_num)

for train\_index, test\_index in kf.split(x):

x\_train, x\_test = x[train\_index], x[test\_index]

#print(x\_train)

#print(x\_test)

for M in range(0,m\_num):

X\_train = np.array([x\_train\*\*m for m in range(M+1)]).T

w = np.linalg.inv(X\_train.T@X\_train)@X\_train.T@ntr

X\_tr=np.array([x\_train\*\*m for m in range(M+1)]).T

X\_te=np.array([x\_test\*\*m for m in range(M+1)]).T

y\_test=X\_te@w

y\_train=X\_tr@w

y\_trn=np.sin(2\*math.pi\*x\_train)+ntr

y\_ten=np.sin(2\*math.pi\*x\_test)+nte

y1=np.sin(2\*math.pi\*x\_test)

y2=np.sin(2\*math.pi\*x\_train)

e\_1=(y\_train-y\_trn)\*\*2

E\_1=(sum(e\_1)/50)\*\*0.5

E\_train[M]=E\_1

e\_2=(y\_test-y\_ten)\*\*2

E\_2=(sum(e\_2)/50)\*\*0.5

E\_test[M]=E\_2

M1[M]=M

#print(E\_train)

p1 = plt.plot(M1,E\_test,'r')

p2 = plt.plot(M1,E\_train,'b')

plt.scatter(M1,E\_test,facecolors='none',edgecolors='r')

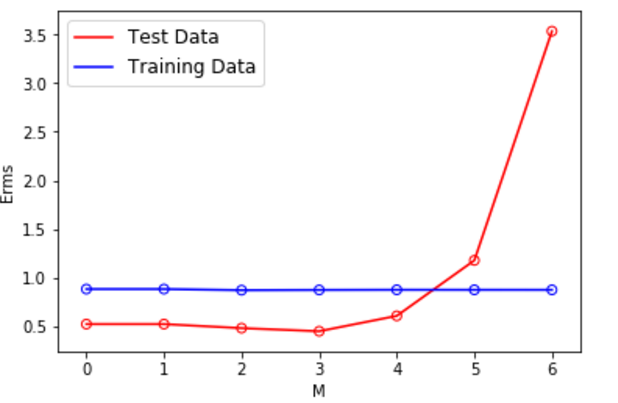
plt.scatter(M1,E\_train,facecolors='none',edgecolors='b')

plt.ylabel('Erms') #label x and y axes

plt.xlabel('M')

plt.legend((p1[0],p2[0]),('Test Data', 'Training Data'), fontsize=12)

Picture: (m=3 according to the picture)



Discussion:

In this example I can take the more training data and different test validation data set to form different shape, and comparing different types of line chart, we can find a proper number.

Q2:

1. XY size: M×N
2. YX size: not defined
3. YXT size: N×M
4. aX size: not defined
5. aTX size: 1×N
6. aXT size: not defined
7. aTb size: not defined
8. bTb size: is constant
9. bbT size: N×N
10. sX+Y size: not defined

Q3:

Suppose the size of X is a\*b, and because of definition of XH, the size of XH is b\*a.

In order to make the computation meaningful, the size of v should be a\*1 and the size of u should be b\*1.

Then:

XHXv = λv : a\*b @ b\*a @ a\*1 = a\*1

XHXu = λu : b\*a @ a\*b @ a\*1 = b\*1

I found here “ b\*a @ a\*1 = b\*1 ”, so I try to define “ XHv=n ”.

Afterwards:

XHX (XHv) = XH λv = λ (XHv)

XHXn = λn

Therefore, they have the same eigenvalues but different eigenvectors.

Q4:

1. = 3xT + 3xT + 4yT = 6xT + 4yT
2. = 6

Q5:

1. =- 10(QX)T - 10XTQ + 4yT=-10 (XTQT + XTQ) + 4yT =

-10XT (QT + Q) + 4yT

1. = -10 (QT + Q)T = -10QT - 10Q