MATH 304 - Numerical Analysis and Optimization Project ---Least Squares Regression

<Xi Chen>

<xc166@duke.edu>

0. Abstract

In this project, we will try to use least square regression to find out the curve that fit the data.

$$f(x) = a_n x^n + a_{n-1}x^{n-1} + \cdots + a_2 x^2 + a_1 x + a_0$$

1. Introduction

Least squares method is a mathematical optimization technique. It seeks the best functional match of data by minimizing the sum of squares of errors. The unknown data can be easily obtained by using the least square method, and the sum of squares of the errors between the obtained data and the actual data is minimized. The least squares method can also be used for curve fitting. Other optimization problems can also be expressed by least square method by minimizing energy or maximizing entropy.

2. Methodology

(1) How you formulate the model fitting problem by the least squares regression:

The theoretical background of the least square method is solving a set of overdetermined linear equations. The solution we want to find is the model coefficients. The model is:

$$f(x) = a_n x^n + a_{n-1}x^{n-1} + \cdots\ + a_2 x^2 + a_1 x^1 + a_0$$

We need to find out the best set of a_n to fit the curve. The process is basically solving the overdetermined function:

$$Ax = B$$

We need to consider to what degree the function would fit the data best. Since we can choose the degree whatever we want within the range of 1 to 9, we can test which degree of the polynomial is the most suitable for the data. Theoretically, as the degree increases, the equation can better fit the data. However, increasing the degree without testing could cause the problem of overfitting which means the model is too specific to the training data and lose its ability to fit the test data. In this case, the model coefficients could be very large, and the figure became twisted.

$$\left[\begin{array}{ccc} & A & \\ \hline \sqrt{\lambda} & & \\ & \sqrt{\lambda} & \\ & & \sqrt{\lambda} \end{array}\right] \cdot \alpha = \left[\begin{array}{c} B \\ \hline 0 \\ 0 \\ 0 \end{array}\right]$$

Figure 1. Ax=B with regularization

Regularization uses the parameter λ to avoid this problem. The weight of λ determines how much want the coefficients vector α to be penalized, which will be tested and determined later by experiments of k-fold cross validation.

(2) Role of regularization

Regularization is used to minimize the overfitting influence which is caused by too inappropriate coefficient for the least square regression. Sometimes, the model is trained too well to fit some train data. In this case, this model will lose some capacity to be generalized to deal with other data like the test data. A common feature of overfitting is that there are many obvious distortions and the coefficients are very large in the fitting curve. To solve this problem, we introduce the concept of regularization which means adding a penalty for this kind of situation to make the coefficient not outstanding in the training. The general format for the regularization is:

$$||A \cdot \alpha - B||_2^2 + \lambda \cdot ||\alpha||_2^2$$

(3) How you implement the Least Squares Regression algorithm

The coefficient for the fitting curve is generalized by solving the $Ax = B$ function with the help of backlash. A matrix contains the coefficient for different basis functions and the regularization parameter λ. B matrix has the values for these functions. In this case, B matrix should be the set of y value for the data points.

We set up one array to store the x value for the data which is called a. Based on this array, we create a new matrix b:

$$b = [a^1 \quad \cdots \quad a^{degree}]$$

Then we will splice a column matrix which stand for the coefficient for a^0 that is all 1 with b adduction from the left.

$$b = [ones \quad b]$$

Then we create a new matrix $c\ degree\ \times degree$ whose the main diagonal is $\sqrt{\lambda}$:

$$c = \begin{bmatrix} \sqrt{\lambda} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \sqrt{\lambda} \end{bmatrix}$$

Matrix A is the combination of Matrix c and Matrix b from vertical direction:

$$A = \begin{bmatrix} b \\ c \end{bmatrix}$$

$$A = \begin{bmatrix} a^0 & a^1 & a^2 & \cdots & a^{degree} \\ \sqrt{\lambda} & 0 & \cdots & & 0 \\ 0 & \sqrt{\lambda} & \cdots & & 0 \\ 0 & 0 & \sqrt{\lambda} & \cdots & \\ & \cdots & & & \end{bmatrix}$$

Picture 1. Composition of A matrix

Meanwhile, we need to deal with the y values of the data points as well. We set up one array to store the y value for the data which is called y. To carry out the matrix calculation, y should be an $\#\ of\ data\ points\ \times degree$ matrix. So we need to provide enough zeros for it:

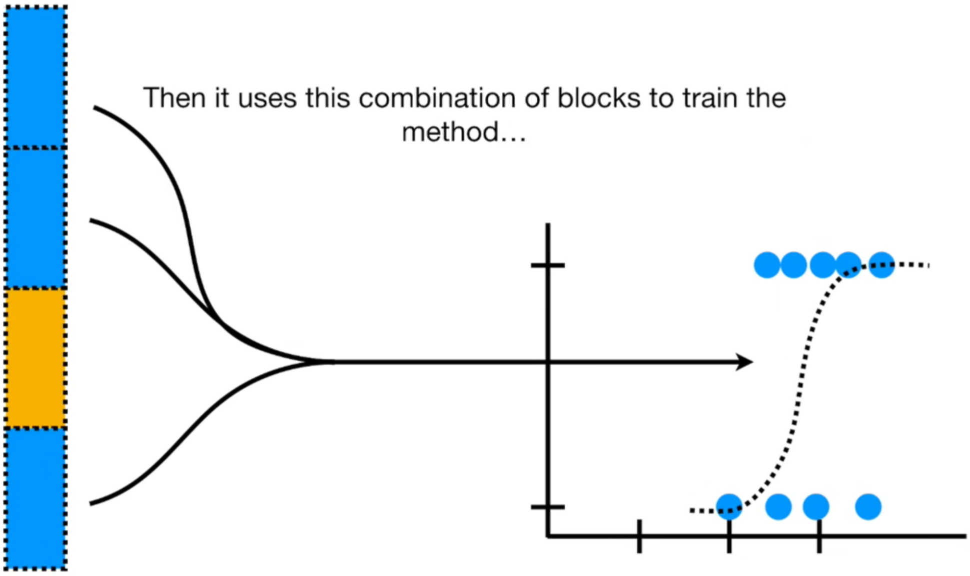
$$B = \begin{matrix} y \\ \vdots \\ 0 \end{matrix}$$

So by calculation:

$$X = A \backslash B$$

We can find out the solution for this least square problem.

Cross-validation method is used to find out the best λ for the regularization. We first divide the LargeData into five parts evenly and choose one part as the Test Data. The rest four part are used as Training Data. The error is calculated and recorded after the training and testing. Then, another part is chosen to be the test data, and the rest are for training. In the end, every part t has been used as the test data. Then we take the average value for each Lambda which should have five different Test Error for each.


Then it uses this combination of blocks to train the method…

Picture 2. Example of 4-fold cross validation

3. Experimental Results

Task1

a.

Model	N=1	N=2	N=3	N=4	N=5	N=6	N=7	N=8	N=9
Training Error	0.855951	0.853836	0.27023	0.243957	0.168454	0.153101	0.110869	0.074539	1.18E-20
Test Error	0.156486	0.146281	0.148816	0.161313	0.25901	0.302324	0.398575	0.413903	4.979396

Table 1. Small Data Error without regularization

b.

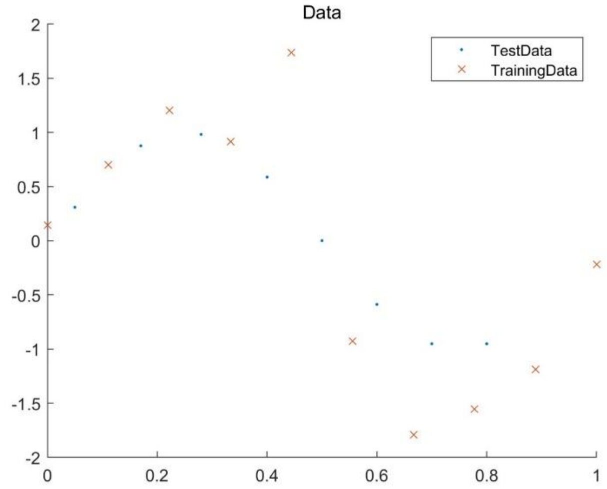


Figure 2. Training Data and Test Data for Task1

c.

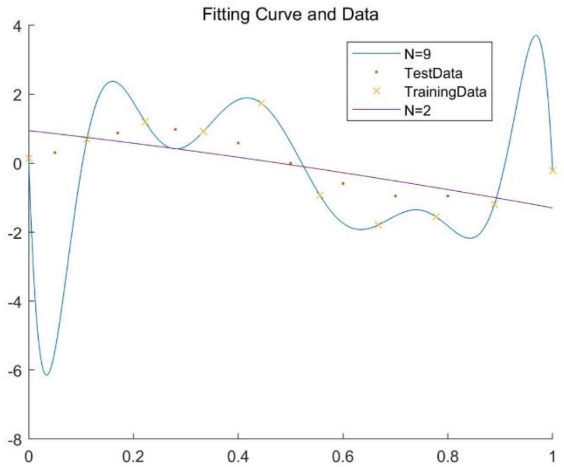


Figure 3. Data and Fitting Curve with smallest error

d. From a., we can see that when $n = 2$, we have the smallest Test Error:

$$a_0 = 0.9464$$
$$a_1 = -1.7276$$
$$a_2 = -0.5127$$

When $n = 9$, we have the smallest Training Error:

$$a_0 = 0.0000$$
$$a_1 = -0.0004$$
$$a_2 = 0.0104$$
$$a_3 = -0.0935$$
$$a_4 = 0.4348$$
$$a_5 = -1.1689$$
$$a_6 = 1.8835$$
$$a_7 = -1.7944$$
$$a_8 = 0.9318$$
$$a_9 = -0.2032$$

Task2

a.

Model	N=1	N=2	N=3	N=4	N=5	N=6	N=7	N=8	N=9
Training Error	0.386532	0.386018	0.215725	0.215499	0.209562	0.206802	0.205401	0.202191	0.20051
Test Error	0.181946	0.178101	0.004612	0.004618	0.000477	0.002946	0.00507	0.00912	0.007493

Table 2. Large Data Error without regularization

b.

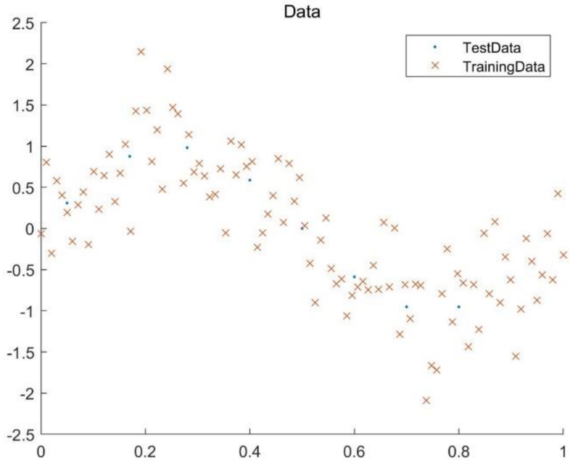


Figure 4.Training Data and Test Data for Task2

c.

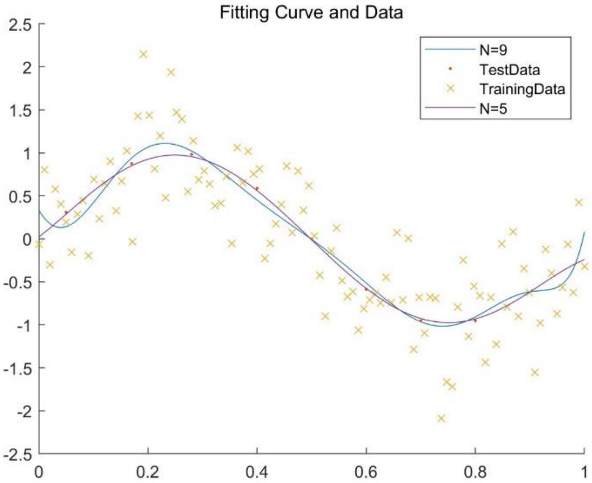


Figure 5. Large Data and Fitting Curve with smallest error

d. From a., we can see that when $n = 5$, we have the smallest Test Error:

$$a_0 = 0.0212$$
$$a_1 = 4.8355$$
$$a_2 = 14.6874$$
$$a_3 = -108.8702$$
$$a_4 = 150.4993$$
$$a_5 = -61.4130$$

When $n = 9$, we have the smallest Train Error:

$$a_0 = 0.0000$$
$$a_1 = -0.0011$$
$$a_2 = 0.0142$$
$$a_3 = 0.0022$$
$$a_4 = -0.4917$$
$$a_5 = 2.2966$$
$$a_6 = -4.9279$$
$$a_7 = 5.6457$$
$$a_8 = -3.3434$$
$$a_9 = 0.8053$$

Task3

a.

Model	10^-6	10^-3	1	10^3	10^6
Training Error	0.156036	0.229233	0.839931	1.370526	1.376684

Test Error	0.280026	0.139054	0.192366	0.538351	0.541034

Table 3. Regularization Error

b.

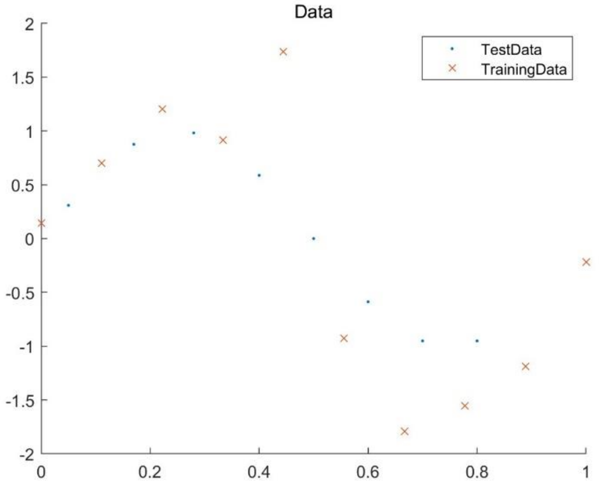


Figure 6. Training Data and Test Data for Task3

c.

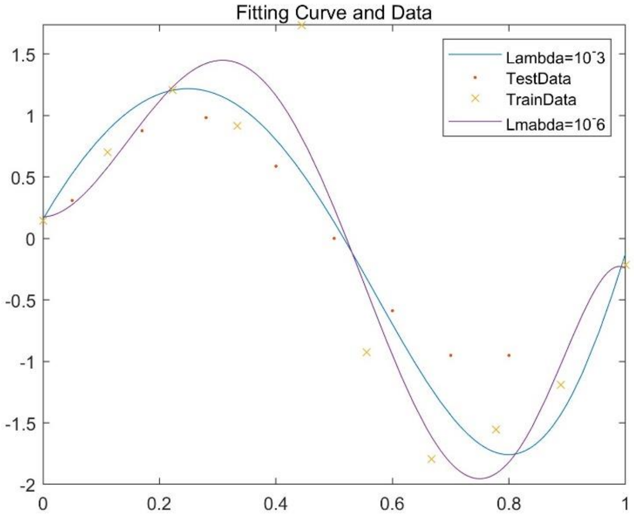


Figure 7. Fitting Curve and Data

d. when $\lambda = 10^{-3}$, the coefficients are:

$$a_0 = 0.1566$$

$$a_1 = 8.0897$$

$$a_2 = -13.1951$$

$$a_3 = -9.4143$$

$$a_4 = 0.4092$$

$$a_5 = 6.9911$$

$$a_6 = 8.4356$$

$$a_7 = 5.6119$$

$$a_8 = -0.0217$$

$$a_9 = -7.1864$$

when $\lambda = 10^{-6}$, the coefficients are:

$$a_0 = 0.1754$$

$$a_1 = 0.0994$$

$$a_2 = 42.5976$$

$$a_3 = -93.6615$$

$$a_4 = -41.5986$$

$$a_5 = 92.0527$$
$$a_6 = 84.5541$$
$$a_7 = -29.4705$$
$$a_8 = -92.7029$$
$$a_9 = 37.7115$$

Task4

a.

Weight	10^-6	10^-3	10^-0	10^3	10^6
average validation error	0.061266	0.021855	0.064627	0.179386	0.136178

Table 4. Cross Validation

c.

$$Test\ Error = 0.27119115012813369557673835053065$$

Coefficient:

$$a_0 = -0.0185$$
$$a_1 = 7.8914$$
$$a_2 = -15.8554$$
$$a_3 = -4.7887$$
$$a_4 = 5.6510$$
$$a_5 = 7.6883$$
$$a_6 = 4.8984$$
$$a_7 = 0.8601$$
$$a_8 = -2.4331$$
$$a_9 = -4.1180$$

Best regularization weight:

$$\lambda = 10^{-3}$$

4. Discussion (2 marks)

From Task 1 and Task 2, we can see that training errors decrease with the increase of degree: This is because with the increase of degree, the curve has better flexibility and can fit the data more appropriately. But this trend did not continue in Test Error: In Task 1, we can see the increasing tendency of the Test Error with the increase of degree. However, in Task 2, the trend becomes downward.

Although it is difficult to draw a conclusion about the Test Error from the perspective of degree. However, by comparing Task 1 and Task 2 horizontally, we can clearly find out that the Test Error in Task 2 are significantly smaller than that in Task 1: This indicates that the accuracy of the model obtained by training with a large amount of data will be significantly higher than that of the model trained with a small amount of data.

We can see that in Task 1, when n=9, the Training Error is obviously smaller than any other Training Error. This is because when the degree is 9, theoretically ten points can be perfectly fitted (SmallData has ten data points and we have ten coefficient when n=9). Therefore, the theoretical value of training error should be 0. But perhaps because of the programming issue, MATLAB still presents a value that is not 0 (although this value is very close to 0).

References

Qiufeng, Wang., 2022. MATH 304 - Numerical Analysis and Optimization Project

Starmer, Josh., 2018 from

https://www.youtube.com/watch?v=fSytzGwwBVw&ab_channel=StatQuestwithJo

shStarmer