Reconstructing Signals via Least Squares Error Modelling

Jonquil Isys R. Mallorca

March 24, 2020

1 Introduction

The Least Squares Method (LSM) is a standard procedure in regression analysis to approximate a solution for a set of coordinates. It involves minimising the residual sum of squares (RSS) of every coordinate to find the best fitting line.

We will use LSM with the data given to train our program and observe the results to determine if it approximates the solution for any file of data points with minimal RSS.

2 Explanation of the program

The program reads in a number of data segments from a file. It finds the function type of each data segment in a file by employing the LSM and retrieving the function type with the least Sum of Squares Error (SSE). The SSE of each line segment is then summed up to form the RSS.

We will now review regression techniques used for polynomial and sinusoidal regressions.

2.1 Linear & Polynomial Regression

Polynomial, and thus linear, regressions can be acquired through the general LSM in matrix form. A degree 3 (cubic) polynomial has been chosen for the polynomial function (we discuss why in subsection 3.2).

We input the vector column of y-coordinates y from a file and a matrix X where each row represents the variables of the terms in the equation of a polynomial.

$$y_{(N\times 1)} = \begin{bmatrix} y_1 \\ \vdots \\ y_N \end{bmatrix} \quad X_{(N\times(p+1))} = \begin{bmatrix} 1 & x_1^1 & \cdots & x_1^p \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_N^1 & \cdots & x_N^p \end{bmatrix}$$

where

$$p = \begin{cases} 1, & \text{if polynomial is linear} \\ 3, & \text{otherwise} \end{cases}$$

Using the input, we compute the coefficient vector $A = (X^tX)^{-1}X^ty$, and regression line $\hat{y}$, where $\hat{y}_i = a_0 + a_1x_i + a_2x_i^2 + \cdots + a_px_i^p$ and each a_i are components of A.

2.2 Sinusoidal Regression

Through an observation of select data points in certain files, namely basic_5.csv, adv_3.csv and noise_3.csv, the unknown function was deduced to be sinusoidal in nature.

Using similar steps in subsection 2.1, We input the y-coordinates y from a file and a matrix X where each row represents the variables of the terms in the equation of a polynomial.

$$y_{(N\times 1)} = \begin{bmatrix} y_1 \\ \vdots \\ y_N \end{bmatrix} \quad X_{(N\times 2)} = \begin{bmatrix} 1 & sin(x_1) \\ \vdots & \vdots \\ 1 & sin(x_N) \end{bmatrix}$$

Using the input, we compute the coefficient vector $A = (X^tX)^{-1}X^ty$, and regression line $\hat{y}$, where $\hat{y}_i = a_0 + a_1 sin(x_i)$ and each a_i are components of A.

3 Results

3.1 Results from chosen regression functions

As demonstrated in Table 1, the RSS in the basic files are extremely small and the sequence of functions used for each file is expected when comparing the line and points plotted by eye. This indicates that the lines plotted are accurate without overfitting.

However in non-basic files, we see higher values of RSS and more disagreeable sequences of function types of each line segment compared to their respective set of data points.

Table 1: RSS and sequence of function types for the line segments of each file

File	**RSS**	**Regression functions used**
basic_1	1.69×10^{-28}	Linear
basic_2	6.21×10^{-27}	Linear, Linear
basic_3	1.44×10^{-18}	Cubic
basic_4	1.39×10^{-12}	Linear, Cubic
basic_5	1.05×10^{-25}	Sine
adv_1	198.23	Cubic, Cubic, Cubic
adv_2	3.65	Sine, Cubic, Sine
adv_3	979.59	Cubic, Cubic, Sine, Cubic, Sine, Cubic
noise_1	10.99	Cubic
noise_2	797.92	Cubic, Cubic
noise_3	477.70	Cubic, Cubic, Sine

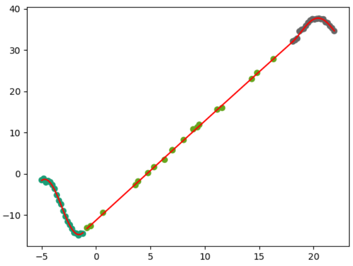

Figure 1: Graph of adv_1.csv

As shown on Figure 1, the first and last line segment function types are agreeable; however, the second line segment is arguably expected to be linear. While the SSE of the cubic function is lower than its linear counterpart, and the line plotted almost follows a linear function, we risk overfitting the data.

This behaviour occurs in all non-basic files, typically in data segments that visibly look linear in nature.

3.2 Comparison to other possible function types

The reasons why a cubic polynomial was preferred compared to other potential polynomials are displayed in Table 2. When $p = 2$, most files have either the same or higher RSS than when $p = 3$, indicating that choosing $p = 2$ would lead to underfitting and unnecessary inaccuracy. In comparison, when $p = 4$, certain files have lower RSS than when $p = 2$; however, other files have gained a large increase in RSS, most notably adv_3.csv.

Table 2: Comparison of RSS when degree of polynomial $p = 2, 3, 4$

File	**RSS** $(p = 2)$	**RSS** $(p = 3)$	**RSS** $(p = 4)$
basic_1	1.69×10^{-28}	1.69×10^{-28}	1.69×10^{-28}
basic_2	6.21×10^{-27}	6.21×10^{-27}	6.21×10^{-27}
basic_3	15.74	1.44×10^{-18}	2.60×10^{-12}
basic_4	0.01	1.39×10^{-12}	4.79×10^{-5}
basic_5	1.05×10^{-25}	1.05×10^{-25}	1.05×10^{-25}
adv_1	218.60	198.23	243.21
adv_2	3.65	3.65	3.40
adv_3	986.58	979.59	91 487.03
noise_1	11.85	10.99	10.54
noise_2	809.24	797.92	727.20
noise_3	482.21	477.70	440.87

In the last data segment of Figure 2, the program chose to use a linear function type to fit the data as it had the lowest SSE when $p = 4$. It is clear that using $p = 4$ or higher values of p would only lead to experiencing undesired consequences caused by overfitting and a risk of choosing highly inappropriate functions to fit the data.

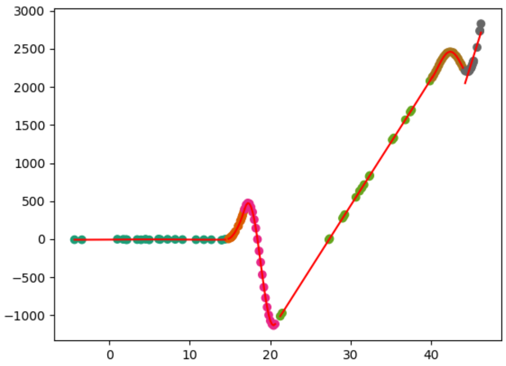

Figure 2: Graph of adv_3.csv when $p = 4$

4 Conclusion

Using linear, cubic and sinusoidal functions as the three primary ways of discerning the regression line of every data segment is sufficient in generalising the data with minimal risk of overfitting/underfitting, with respect to the training data given.

As a future improvement, we could use regularisation with cross-validation to further the accuracy of our program.