Symbols, Patterns, Signals Coursework 1: An Unknown Signal Report

Lik Wong

20th May, 2020

1. **Introduction**

For this task, the Least Square Method (LSM) is utilised to approximate a line given a sequence of coordinates. The objective of the task is to minimise the Residual Sum of Squares (RSS) or the sum of squared errors in order to provide the most accurate solution.

The program implemented reads from a file with a set of points and uses LSM once for each function type, namely linear, polynomial and sine per segment (20 points). Afterwards, it decides which type of function each line segment will be by selecting the one with the least RSS. This is done on training data initially to work out function types and improve on accuracies, which we then move on to more advanced and broader data sets.

1. **Method**

The general equation for LSM in its matrix form is as follows with the input coordinates’ respective x and y components:

Then we calculate the coefficient vector using:

and regression line where each is an element of .

From the above, n is the number of points and is the power of the regression line. Hence, this implies that the equation works for any polynomial function by changing the value of . In this case, powers for linear and cubic are chosen for our solution ().

Nonetheless, given that the unknown function is not polynomial, the previous equation won’t satisfy it. Through using a combination of trial-and-error and observation on the files with an unusually large RSS (basic\_5.csv, adv\_3.csv and noise\_3.csv), the unknown function was determined to be sinusoidal. We have to, therefore, alter the general equation:

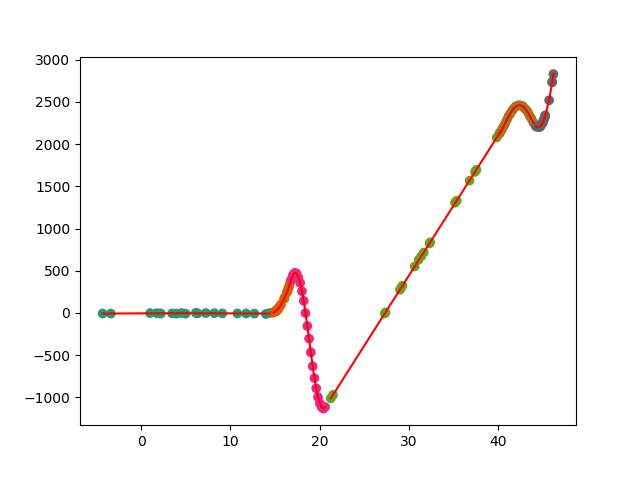
Then we calculate the coefficient vector using:

and regression line where each is an element of .

1. **Analysis**

The result from all the datasets with their respective function types are listed in the table below:

|  |  |  |
| --- | --- | --- |
| File Name | RSS (to 3 d.p.) | Function Type for Each Segment (in order) |
| basic\_1 |  | linear |
| basic\_2 |  | linear, linear |
| basic\_3 |  | cubic |
| basic\_4 |  | linear, cubic |
| basic\_5 |  | sine |
| adv\_1 |  | cubic, cubic, cubic |
| adv\_2 |  | sine, cubic, sine |
| adv\_3 |  | cubic, cubic, sine, cubic, sine, cubic |
| noise\_1 |  | cubic |
| noise\_2 |  | cubic, cubic |
| noise\_3 |  | cubic, cubic, sine |

 There are a few notable features shown on this table. Firstly, it is apparent that all the basic files have a measly RSS. This indicates that the functions chosen and implementation are definitely correct. Secondly, certain segments that are cubic in the advanced and noisy data sets can be seen by eye that it should be linear. Illustrated in graph plot of adv\_3.csv below, the first and fourth segment are cubic when it is visibly linear. With this in mind, we could potentially be overfitting for these data sets.

Plot of adv\_3.csv

Furthermore, one question we can ask ourselves is does increasing or decreasing the power of the polynomial function improve the results? More specifically, how did we deduce that the cubic function provides us with the most accurate model? To answer this, observe the outcome where we test different values of on files that has cubic segments:

1000 = lowest RSS

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| File Name | RSS (3 d.p.) | | | |
| Quadratic () | Cubic () | Quartic () | Quintic () |
| basic\_3 |  |  |  |  |
| basic\_4 |  |  |  |  |
| adv\_1 |  |  |  |  |
| adv\_2 |  |  |  |  |
| adv\_3 |  |  |  |  |
| noise\_1 |  |  |  |  |
| noise\_2 |  |  |  |  |
| noise\_3 |  |  |  |  |

Generally, it is clear that cubic performs the best consistently out of all the other types. Take quadratic for example, the only merit for it is in adv\_3.csv where it has the lowest RSS across the board but does poorly on every other file, a case of underfitting. Quintic suffers from a similar problem to quadratic, with noise\_1.csv bearing the best result and everything else below par. To the extent where the RSS in adv\_3 is almost 98 times larger than that of cubic. This is due to the fact that certain segments are using the linear model instead. Finally, for quartic, on the surface, it has the same number of lowest RSS as cubic in the table. Nevertheless, it shares the same undesirable trait with quintic where it uses a linear model for its segments in adv\_3.csv, resulting in an almost identical RSS. Notably, we can see a pattern here where higher values of won’t necessarily enhance our results and may lead to overfitting.

1. **Future Improvements**

As for improvements for future implementations, we could add a check for overfitting by using the type with the lowest RSS only if its value is lower than the next lowest RSS type by an arbitrary percentage.

Moreover, in order to enhance our accuracy, we could regularise our model by using a k-fold cross validation.

1. **Conclusion**

In conclusion, in terms of generalisation of new data, the 3 chosen function types, linear, cubic and sine, produce the most accurate output. Between changing the power of the polynomial function or switching to a completely different type for the unknown function (to exponential, cosine or tangent), there is no better alternative to the situation as it would only lead to a worse fit. Though we do have to acknowledge the fact that we may be already overfitting in some of the files and there are room for improvements for our model.