1. The problem

In this coursework we were provided with files that have points of x,y coordinates which form an unknown signal. Each of these unknown signals consists of a number of different line segments each one represented by up to 20 points of this signal. Each line segment is either a linear, a polynomial or an unknown function.
The order of the polynomial and the kind of the unknown function is decided by the student after observing the example signals provided.
The determination of the function type for each line segment will be done using least squares regression.
Also, in this implementation, to prevent overfitting cross validation was used but only to decide between linear and polynomial function types.
Finally, we are asked to estimate a model given the data and output the resulting total error.

2. Polynomial Order & unknown function type selection

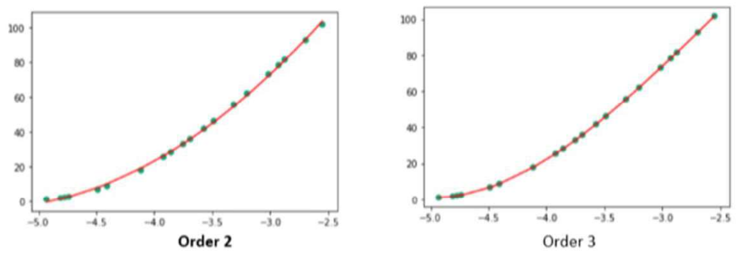
To decide the order of the polynomial function, the example unknown signals were used to determine the most optimal order.
I took every segment of all the example signals and fitted to them linear, polynomial and unknown functions to see how optimally they were fitting.

The issue here is that for the polynomial functions is that as the order increases the error between the fitted line and the signal decreases. But as the order increases the complexity of the function increases as well, so we have to find a nice balance between the error and the complexity.

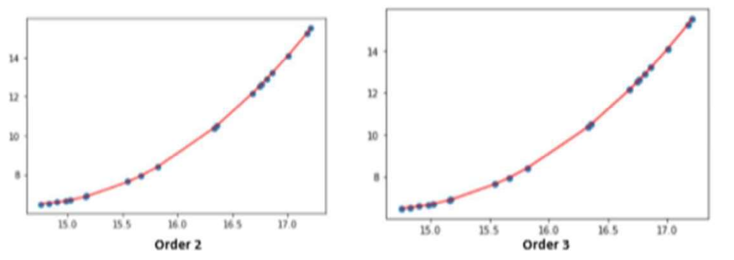
- Polynomial Order selection -> 2nd order

For the polynomial I decided that I'm going to use order 2 since from the available data if the data fit a polynomial usually order 2 and 3 have virtually the same result but of course order 3 has more parameters. Therefore, I chose order 2.
Examples:

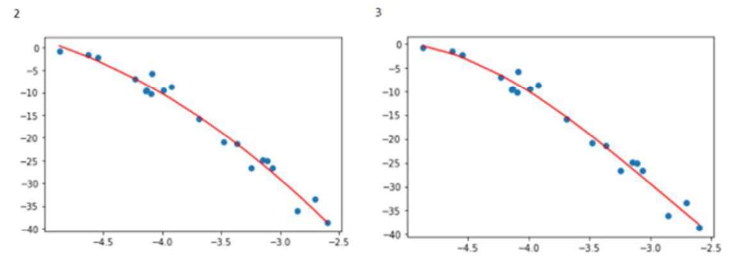
Basic_3

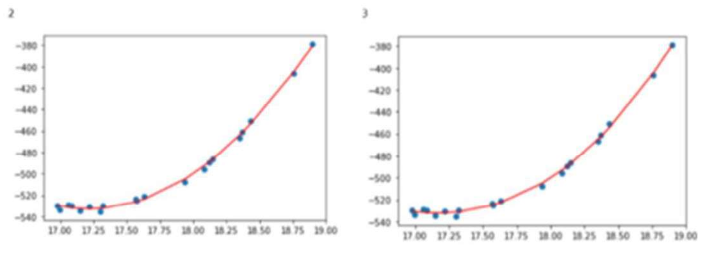


Basic_4



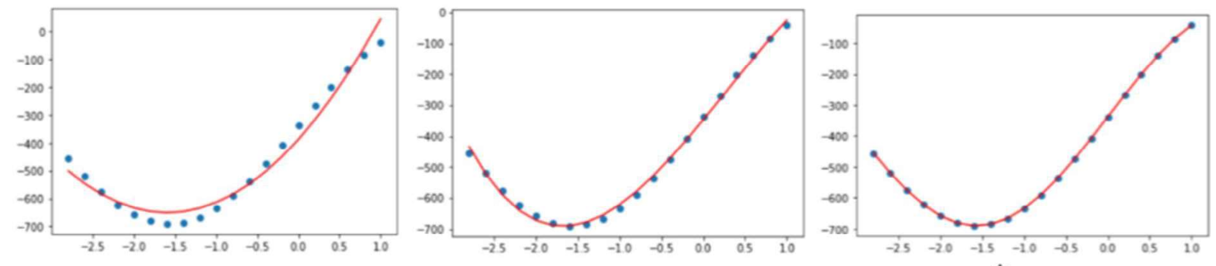
Adv_1





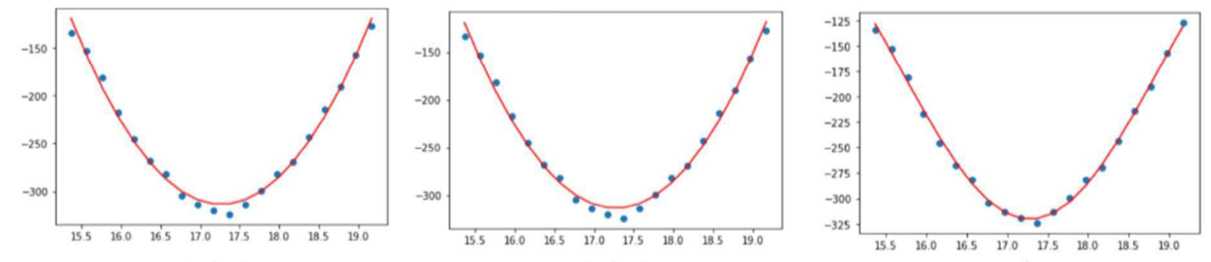
- Unknown function selection -> sinusoidal

To determine the unknown function I observed the example signals and tried a few different types e.g. the exponential and the sinusoidal.
I noticed that in the segments that the 2nd order polynomial and the 3rd order polynomial didn't quite fit the signal nicely, the sinusoidal always fit right.
For example:



Order 2 Order 3 sin

Noise_3



Order 2 Order 3 sin

Therefore, by observing the example signals, it was determined that if a linear function didn't quite fit the segment, a 2nd order polynomial had to be checked and if a polynomial didn't fit then a sinusoidal function would fit.

3. Least Squares Regression

The program uses the least squares method to fit functions through the data points.
It is used for 3 different types of functions:
a) linear,
b) polynomial (2nd order), and
c) unknown (sinusoidal)

The line that has to be fitted through the given data points is **linear**: $\boldsymbol{f(x) = a + bx}$

The parameters are a and b and the residual error is: $R(a, b) = \sum_i (y_i - (a + bx_i))^2$

The residual error between the data and the fitted line in matrix form is: $R(\vec{w}) = \|\vec{y} - X\vec{w}\|^2$

$$\text{where: } \vec{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix}, X = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_N \end{bmatrix}, \vec{w} = \begin{bmatrix} a \\ b \end{bmatrix}$$

To find the least squares in matrix form we need to find the weights (parameters).
For the best line the residual error has to be minimized:

$$R(\overrightarrow{w_{LS}}) = \|\vec{y} - X\,\overrightarrow{w_{LS}}\|^2$$

$$\|\vec{y} - X\,\overrightarrow{w_{LS}}\|^2 = 0$$

$$\vec{y} - X\,\overrightarrow{w_{LS}} = 0$$

$$X\,\overrightarrow{w_{LS}} = \vec{y}$$

$$X^T X\,\overrightarrow{w_{LS}} = X^T\vec{y}$$

$$\overrightarrow{w_{LS}} = (X^T X)^{-1}\,X^T\,\vec{y}$$

For the fitting of the other type of functions, the only differences are their X matrix and their weight vectors with the parameters.

2nd order polynomial: $\boldsymbol{f(x) = a + bx + cx^2}$

Residual error: $R(a,b) = \sum_i (y_i - (a + bx_i + cx_i{}^2))^2$

Residual error in matrix form: $R(\vec{w}) = \|\vec{y} - X\vec{w}\|^2$

where: $\vec{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix}, X = \begin{bmatrix} 1 & x_1 & x_1{}^2 \\ 1 & x_2 & x_2{}^2 \\ \vdots & \vdots & \vdots \\ 1 & x_N & x_N{}^2 \end{bmatrix}, \vec{w} = \begin{bmatrix} a \\ b \\ c \end{bmatrix}$

Sinusoidal: $\boldsymbol{f(x) = a + b\sin(x)}$

Residual error: $R(a,b) = \sum_i (y_i - (a + b\sin(x_i)))^2$

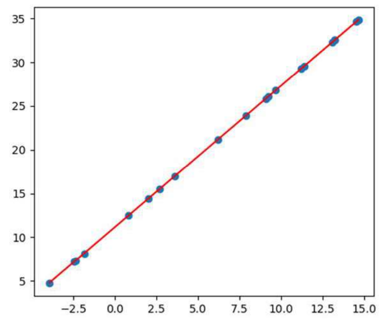
Residual error in matrix form: $R(\vec{w}) = \|\vec{y} - X\vec{w}\|^2$

where: $\vec{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix}, X = \begin{bmatrix} 1 & \sin(x_1) \\ 1 & \sin(x_2) \\ \vdots & \vdots \\ 1 & \sin(x_N) \end{bmatrix}, \vec{w} = \begin{bmatrix} a \\ b \end{bmatrix}$

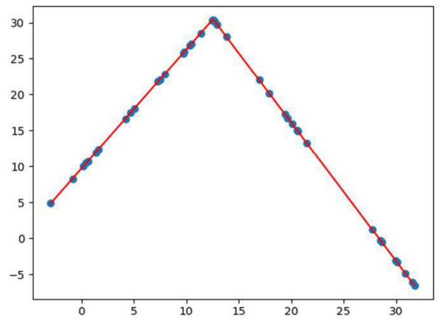
4. Results

Figures demonstrating the results for the training data provided
In blue there are the example points of the unknown signal, in red it is the fitted function and below the plot there is the least squared error.

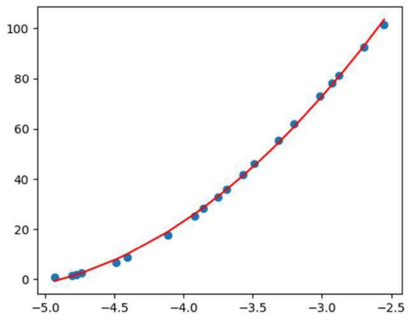
Basic:



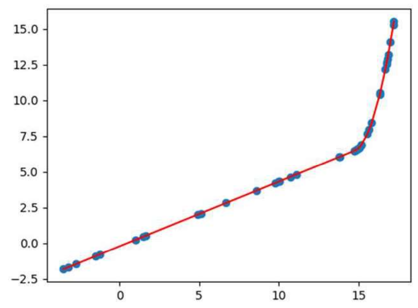
```
1.868811484468577e-27
```



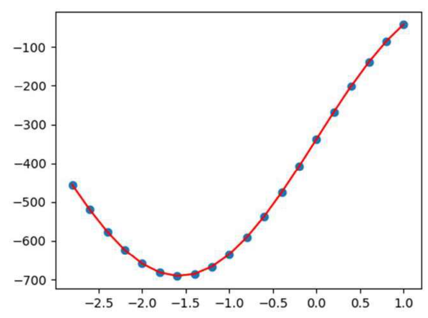
```
2.505816665434544e-26
```



```
15.743318683748587
```

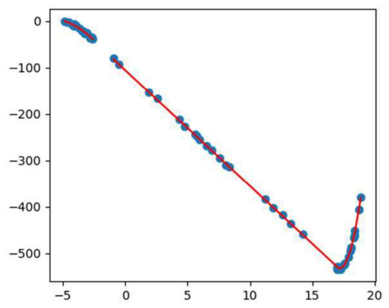


```
0.0072686690455590905
```

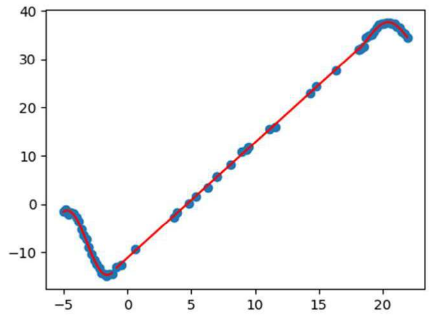


```
1.2682359001057163e-25
```

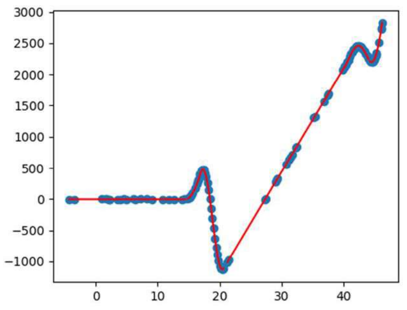
Adv:



```
218.59789456027073
```

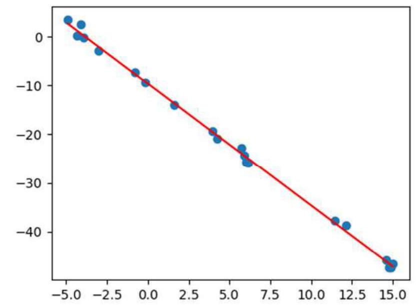


```
3.685132050447588
```

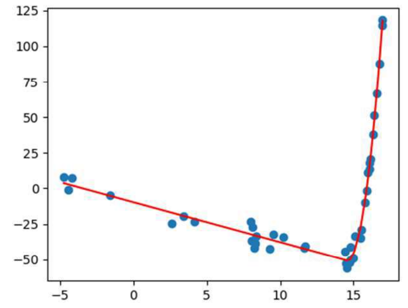


```
1014.409779492245
```

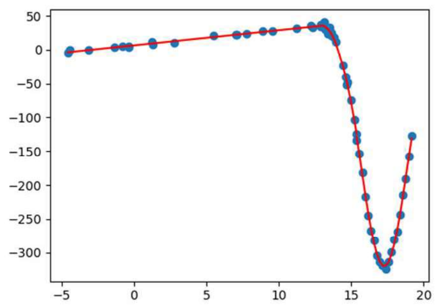
Noise:



```
12.207460140137092
```



```
850.9013645733851
```



```
483.0638008489906
```

5. Discussion

Cross validation is applied to resolve overfitting only to decide between the linear and 2nd order polynomial function fitting. This is because for example a lot of segments have to be fitted with a linear model, but since the 2nd order polynomial has more parameters the program chooses always the 2nd order polynomial, since it has a smaller error.
But that’s not correct since it overfits.

Process of finding the best order (linear vs 2nd order polynomial) using cross validation:

1. Divide data points to training and validation set (the ratio was decided after trial and error) This separation of points was done with the linspace command, so that the points are linearly spaced along the signal.
2. Model with training set for linear function
3. Use the fitted parameters and find the error in the validation set
4. Model with training set for 2nd order polynomial function
5. Use the fitted parameters and find the error in the validation set
6. Compare the errors and choose the type of function with the smallest one

At the end, the program for each segment compares the error between the linear and polynomial using cross validation and the sinusoidal error and chooses the one with the smallest error.