TU ILMENAU

Using the Data Quality Assessment GUI

MATLAB Programme

Yuri A.W. Shardt

Table of Contents

List of Figures

List of Tables

No table of figures entries found.

Chapter 1: Introduction to Data Quality Assessment

Data quality assessment is an automated procedure that allows large data sets to be assessed for their value in system identification. As data sets become larger and more complicated, it is difficult to manually extract those regions that can be used for system identification. Furthermore, there may be cases where it is desired to automatically create models from a given data set. In all such cases, it is necessary to assess the quality of the data set before using it and extract only those regions that are informative enough for the required model.

Theoretically, data quality assessment is based on the invertibility and numerical conditioning of the Fisher information matrix, $\mathscr{F}$. Basically, if the Fisher information matrix is invertible and well-conditioned, then the data set should provide a good estimate for the parameters. The invertibility and conditioning of the matrix can be assessed using the eigenvalues of the matrix, that is, the data quality assessment index is given as

$$\eta_{data} = \frac{\max\left(\left|\lambda\left(\mathscr{F}\right)\right|\right)}{\min\left(\left|\lambda\left(\mathscr{F}\right)\right|\right)} < \varepsilon \tag{1}$$

where λ is the eigenvalue of $\mathscr{F}$ and ε is a threshold. A data set is said to be **informative enough** with respect to the given model structure if $\eta_{data} < \varepsilon$, that is, the Fisher information matrix is sufficiently well-conditioned for the taking of an inverse. Furthermore, this implies that variances of the parameters will be reasonable, and hence, the results will be significant. In practice, the threshold of 10^4 works well. However, the threshold can be changed based on the desired accuracy of the model, the noise properties, and the model structure.

Figure 1 shows the general steps involved in data quality assessment. The detailed steps are (Peretzki, Isaksson, Bittencourt, & Forsman, 2011; Bittencourt, Isaksson, Peretzki, & Forsmann, 2015; Shardt, Yang, Brooks, & Torgashov, 2020):

1) **Preprocessing**: In this step, the data set is loaded and analysed. Often, the data set will be mean scaled and centred. As well, known thresholds and limits will be considered at this point.
2) **Mode changes**: It may be known how the operating modes of the process change, for example, the controller modes could change based on the relationships between r_t and u_t.

As well, it may be the case that the specific mode will determine how much data is required to obtain good parameter estimates.

3) **Partitioning**: For each identified mode, perform the following steps:
    a. **Initialisation**: If the length of the unanalysed data for the given mode is greater than the minimum required length r, set the model counter to the current data point, $k_{init} = k$ and then set $k = k + r$. Otherwise, go to the next identified mode.
    b. **Preprocessing**: For certain types of processes, it may be necessary to perform additional manipulations, for example, for an integrating process, it is necessary to integrate the input.
    c. **Computation**: Compute the required values. In most cases, this will include the variances of the signals and the condition number of the information matrix.
    d. **Comparison**: Compare the variances, the condition number of the regressor matrix, and the significance of the parameters against the thresholds.
        i. **Failure**: If any of the thresholds fail to be met go to the next data point, that is, $k = k + 1$, and go to Step 3.a.
        ii. **Success**: Otherwise, set $k = k + 1$, and go to Step 3.b. The "good" data region is then $[k_{init}, k]$.
        iii. **Termination**: The procedure stops once k equals N, the total number of data points in the given mode. Repeat Step 3 for any remaining modes.
4) **Simplification**: To improve performance and obtain longer data sets, adjacent regions can be compared and determined if they could be considered to come from a single model.

Practically, a recursive method is used to compute the required variances, that is,

$$\begin{aligned} m_{y_t} &= \lambda_{m_y} y_t + \left(1 - \lambda_{m_y}\right) m_{y_{t-1}} \\ \sigma^2_{y_t} &= \frac{2 - \lambda_{m_y}}{2}\left(\lambda_{\sigma_y}\left(y_t - m_y\right)\right)^2 + \left(1 - \lambda_{\sigma_y}\right)\sigma^2_{y_{t-1}} \end{aligned} \tag{2}$$

where λ is the forgetting factor and σ^2 is the variance of the given signal. The two forgetting factors, λ_{m_y} and λ_{σ_y}, need to be tuned.

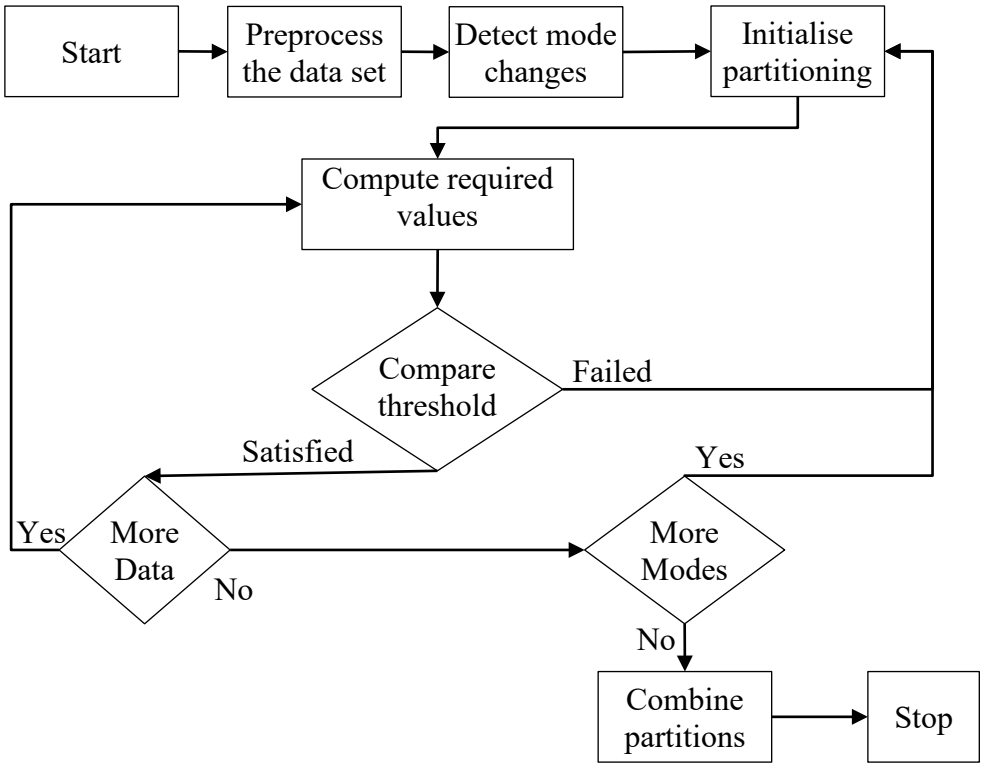

Figure 1: Data Quality Assessment Framework

Finally, consideration needs to be given to setting all the parameters and thresholds required for the data quality method. Since, in general, Laguerre models are used, the Laguerre model parameters α and N_g need to be set. From (Peretzki, 2010), we have that

$$N_g \geq -\frac{\theta \log(\alpha)}{2\tau_s} + 1 \tag{3}$$

where θ is the continuous time delay and τ_s is the sampling time. Previous investigations have shown that α should be set between 0.80 and 0.95 (Shardt & Shah, 2014). The forgetting factors in Equation (2) can be set to a value of 0.99 (Shardt & Shah, 2014).

Setting the thresholds can be a tricky matter since the properties of the signals in the specific case will need to be considered. In general, with routine operating data, the variance thresholds will need to be set lower, as the amount of variance in the data can be extremely small.

Chapter 2: Installing the GUI

There are two different versions of the GUI that can be downloaded. The first version is the raw MATLAB code stored on the GitHub server, while the second version is a MATLAB package that installs the GUI directly into MATLAB. Once installed, it can be accessed as any other MATLAB GUI.

Section 2.1: GitHub Version

The GitHub version can be downloaded from [**uk: Insert Link**]. The files should be stored in a single folder. Clicking on the file `DQGUI.mlapp`, while load it into MATLAB from which changes can then be made to the code. If you wish to participate in the development of the programme, please contact the author and he can arrange the appropriate access.

Section 2.2: MATLAB-Package Version

The MATLAB-package version can be downloaded from [**uk: Insert Link**] as a single file called `Data Quality Assessment.mlappinstall`. In order to install it in MATLAB, perform the following steps:

1) Download the file to your computer.
2) Double-click on the file. It will open in MATLAB. You should see the window shown in Figure 2.
3) Click on `Install`. This will add the programme to your `APPS` tab in MATLAB. This is shown in Figure 3.
4) You can open the GUI by clicking on the icon.

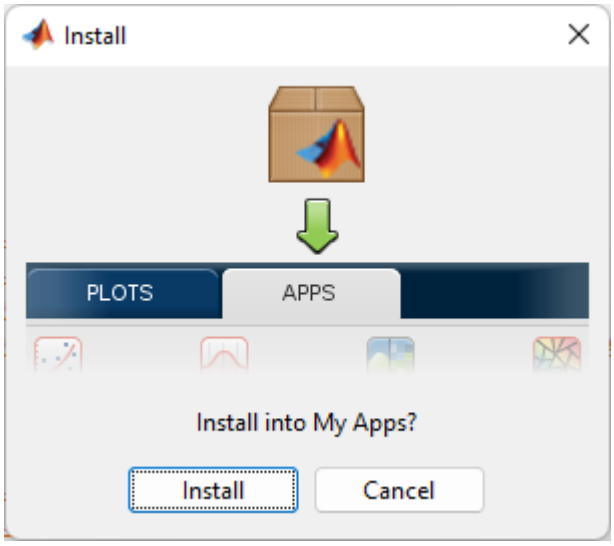

Figure 2: Installing the GUI in MATLAB

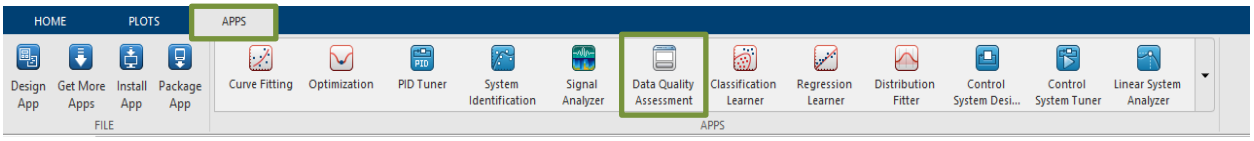

Figure 3: Finding the Installed GUI in the APPS Tab of MATLAB (The exact ordering and available GUIs depends on the toolboxes installed in your version of MATLAB.)

Section 2.3: Requirements for Running the GUI

For the GitHub version, it should run on any version of MATLAB after 2019a. Minor tweaks may need to be made for the older versions. The MATLAB-package version will run on MATLAB 2020b or newer.

The file format required by the programme is a common data storage format. The first column is always the time formatted as YYYY-MM-DD HH:MIN:SS using the 24-hour clock. The first row contains the tag names, while the second row contains the tag description. The third row contains the units for the particular variable. Only the first row must contain values. The other two can be empty. The data for the variables starts from the second column and third row. There are as many columns as there are variables. An extract from a properly formatted file is shown in Figure 4.

Tagname	FI60320.P`	FIC60324.	FIC60324.	FIC60324.
Description	Feed rate	Air flow to	Air flow to	Air flow to
UOM	l/s	m3/hr	m3/hr	m3/hr
2015-08-20 13:00	476.6147	47.09	249.31	250
2015-08-20 13:01	480.224	47.1	249.8	250
2015-08-20 13:02	481.8149	47.12	250.39	250
2015-08-20 13:03	477.3504	47.13	250.97	250

Figure 4: File Format for the GUI

Chapter 3: Description of the GUI

The MATLAB GUI consists of four tabs: `DQA`, `Input Settings`, `Setpoint Settings`, and `Parameters`. The `DQA` tab, shown in Figure 5, is the main interface with the programme. Here, the data can be loaded, selected, and analysed. The buttons and the order in which they are to be pressed have been given in Figure 5. Please note that options that cannot be implemented at a given point are greyed out. When loading the data, it should be noted that the file should be comma-delimited (.csv) with a specified format described in **Error! Reference source not found.**Section 2.3. In box 4, the time interval should be entered using standard MATLAB vector notation, that is, [*a*, *b*], where *a* is the initial time and *b* is the final time. The graph `Original Data` does not update until the `Save settings and plot` button is pressed. Once it has been pressed, then it is possible to compute the partitions and save them.

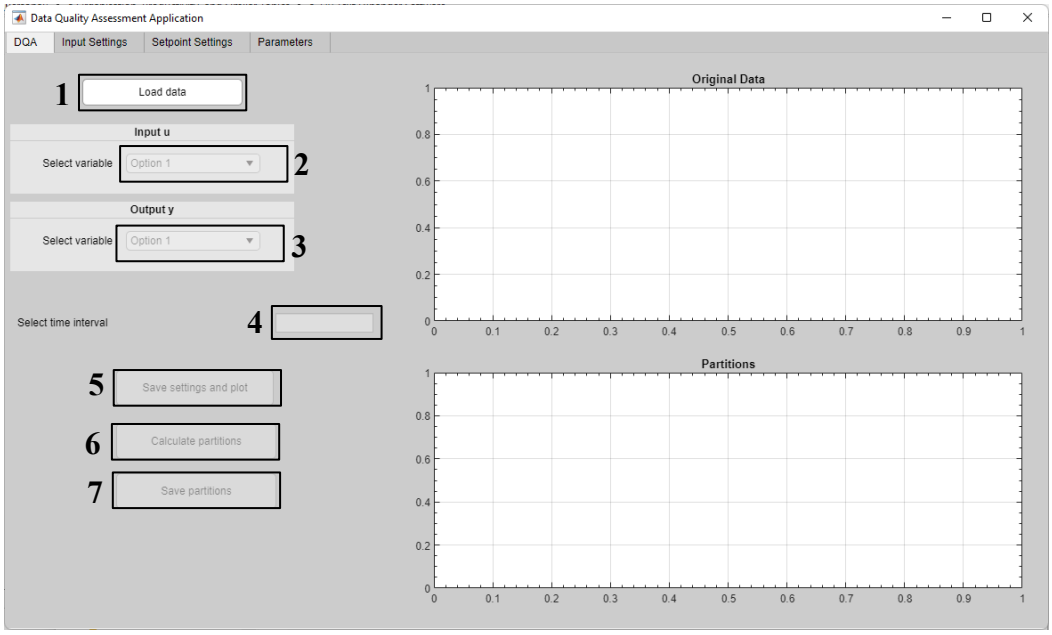

Figure 5: Main GUI Interface

The `Input Settings` tab, shown in Figure 6 (left), allows the user to preprocess the input signal and remove undesirable values and constraints. To select a given option, click the `Click to Select` tickbox and it will be possible to edit the options. Currently, it is possible to select limits on the value and the variance. The variance limits are computed using the *n* last data points, where *n* is the length of the window. Once the constraint parameters have been

selected, it is necessary to press `Implement Constraints` so that the programme can update everything. The graph will be updated based on the selected constraints. The `Setpoint Settings` tab, shown in Figure 6 (right), has the same features but allows setting a setpoint signal and placing constraints on it. In general, the programme assumes that the data quality assessment will be performed using open-loop data. If this is not the case, then we can specify a setpoint signal. It should be noted that the programme assumes that if the setpoint and output are the same, then the process is running in open-loop conditions and it will automatically preprocess the data set using this rule.

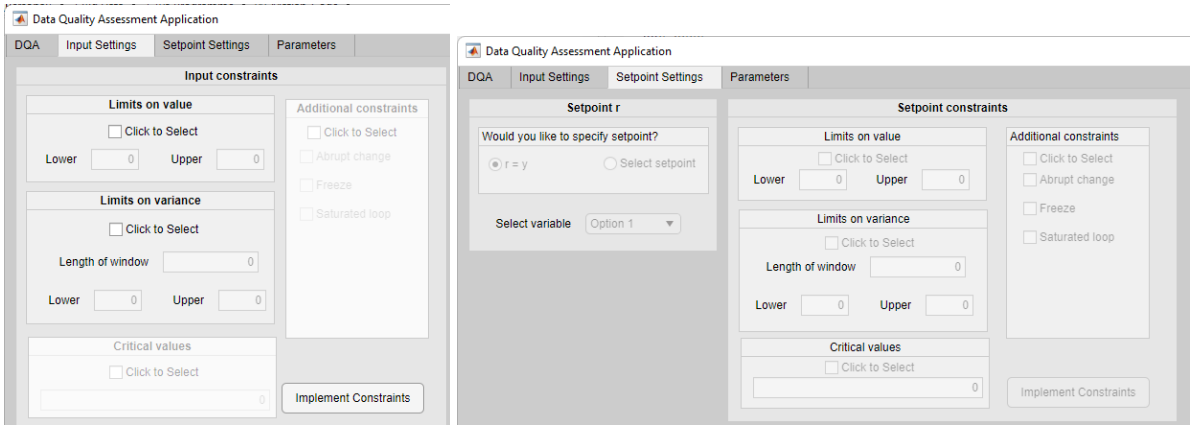

Figure 6: (left) The `Input Settings` Tab and (right) the `Setpoint Settings` Tab

The `Parameters` tab, shown in Figure 7, allows the user to change the default settings for the parameters required by the programme. In general, it is recommended to keep the default values unless the data at hand has some special requirements.

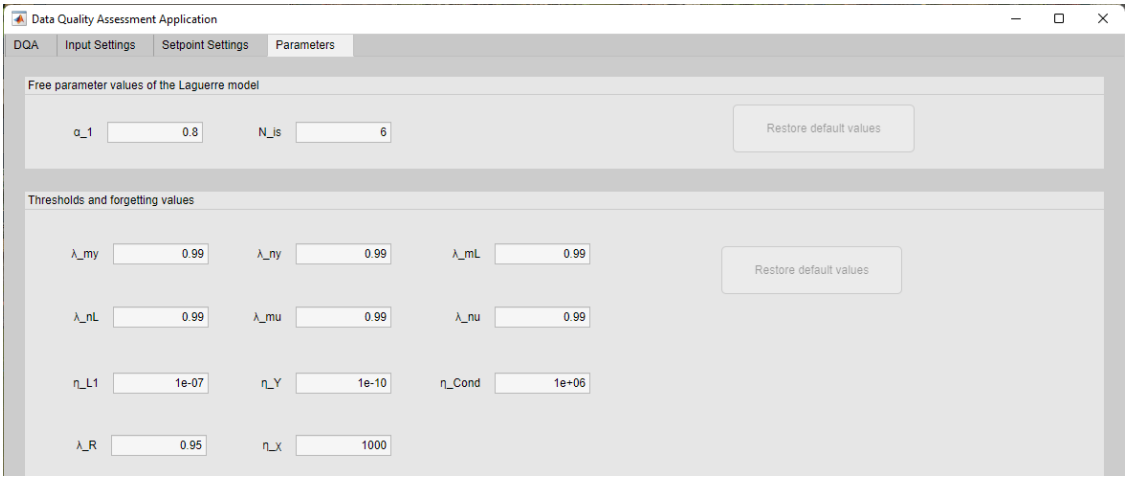

Figure 7: The `Parameters` Tab

Chapter 4: Detailed Example of Using the GUI

In this chapter, a brief example will be presented that shows how to use the data quality assessment GUI. The data set can be downloaded from https://doi.org/10.5281/zenodo.6674007. It will be assumed that you have installed the MATLAB package and are using it for this example. If you wish to use the raw code, then run the `.mlapp` file and in the window that appears click the `Run` button. It is then possible to make changes to the code as the programme is being used.

Once the data set has been download and MATLAB started, perform the following steps:

1) Click on the `Data Quality Assessment` icon in the `APPS` tab in MATLAB.
1) Once the programme starts, click the `Load data` button (box 1 in Figure 5). In the window that appears, select the folder where the `.csv` file has been stored and select the downloaded file. Once the processing window has closed, you will be returned to the main interface.
2) For the input (box 2 in Figure 5), select `CX006A.PV` and for the output (box 3 in Figure 5) `Kero.freeze` from the dropdown menu.
3) Click the `Save settings and plot` (box 5 in Figure 5). The original data plot should resemble that shown in Figure 8. We can see that the input occasionally deviates strongly from the expected value and approaches 0. At the same time, we can see that the output is not available (the space between the data points). Therefore, we will preprocess the input signal.

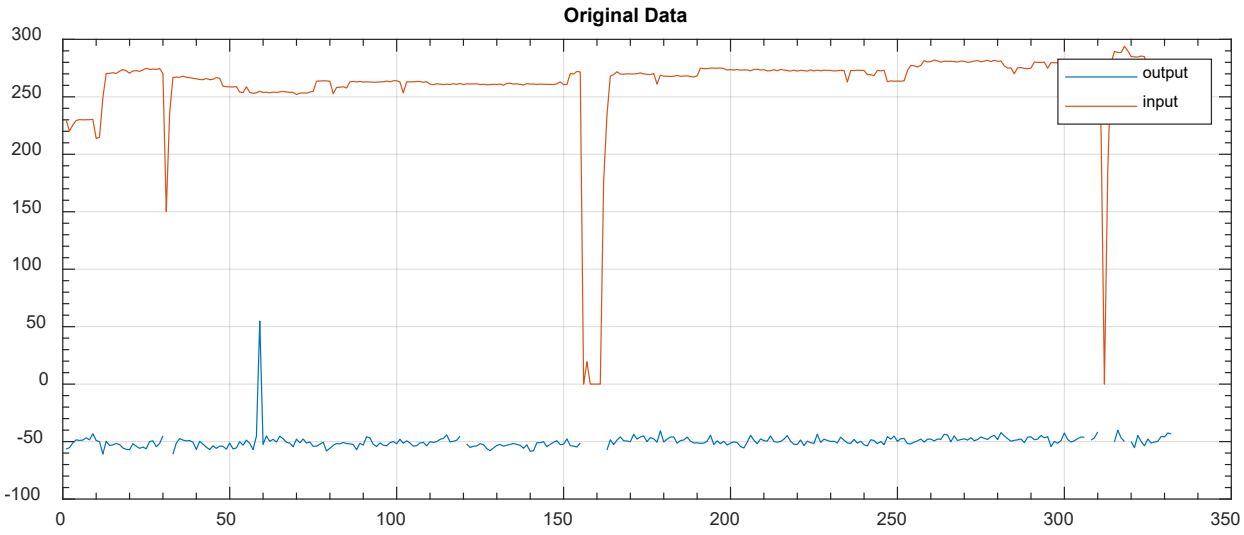

Figure 8: Plot After Loading the Data

4) Go to the `Input Settings` tab. In the `Limit on value` area, click on `Click to select` and enter a value of `200` for `Lower` and `500` for `Upper`. Finally, click on `Implement Constraints`. The steps are shown in Figure 9. Note that both the upper and lower constraints must be set. Leaving the value of zero will set the give bound to zero.

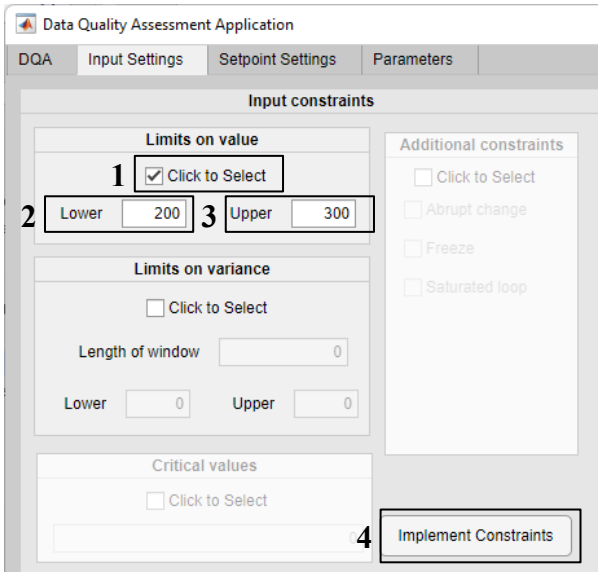

Figure 9: Setting the Constraints on the Input

5) You will be returned to the main tab and the original data plot will have changed to that shown in Figure 10. We can see that the input values have been removed when they are anomalous.

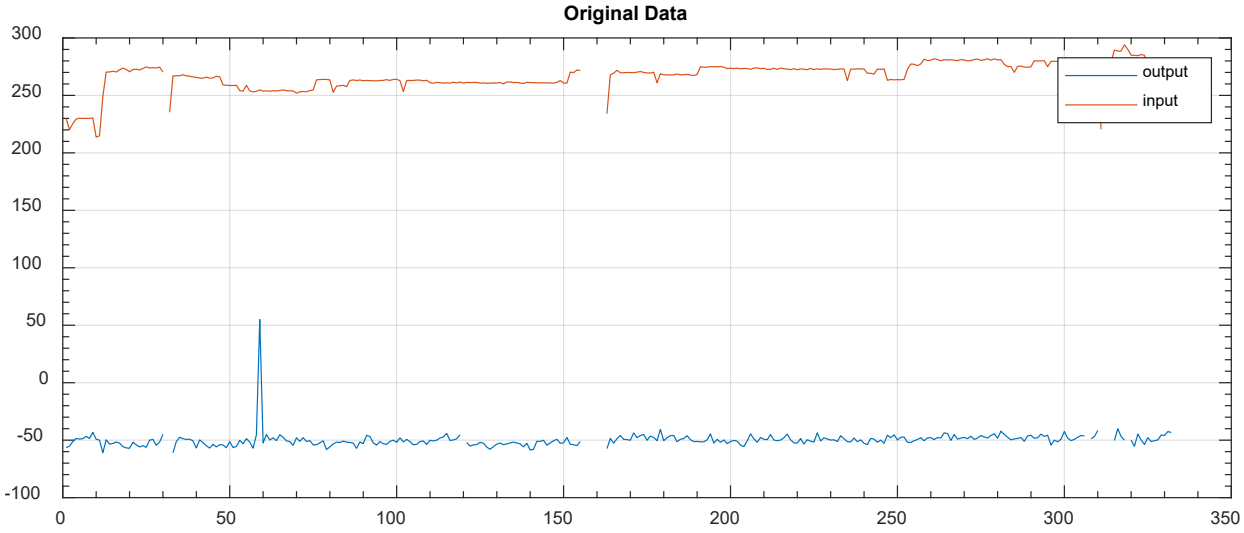

Figure 10: Data Plot after Preprocessing the Input

6) Click on the `Calculate partitions` button (box 6 in Figure 5). After a few seconds, the partitions should be displayed. This will give a partitions graph similar to that shown in Figure 11. We can see that the programme has split the data and removed the regions that are not suitable for modelling. It can be noted that small regions have also been excluded since these may not provide sufficient information for the model.

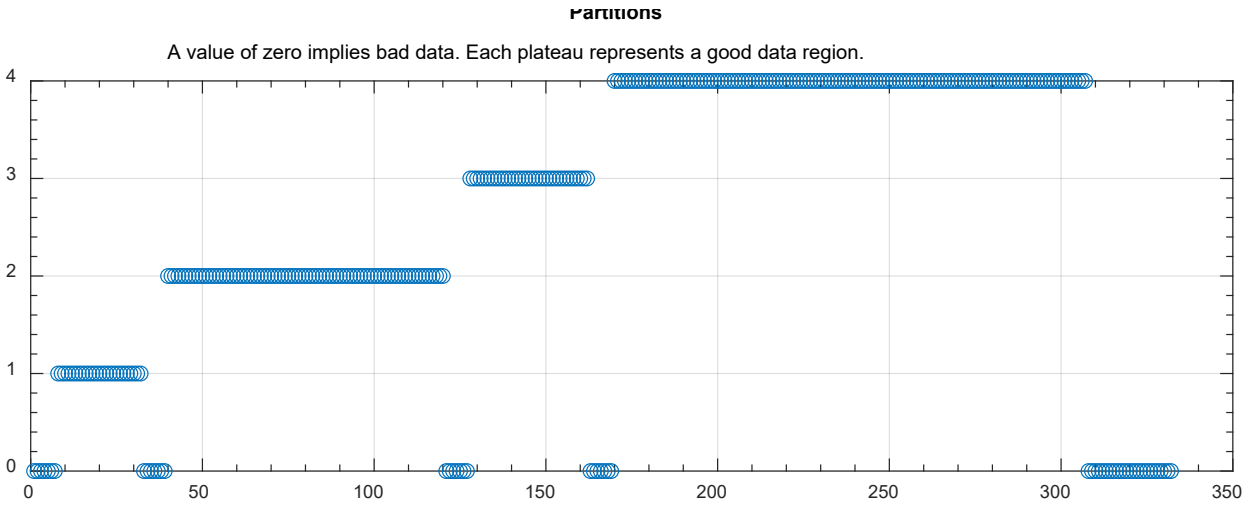

Figure 11: Partitions

References

Bittencourt, A. C., Isaksson, A. J., Peretzki, D., & Forsmann, K. (2015). An Algorithm for Finding Process Identification Intervals from Normal Operating Data. *Processes, 3*(2), 357-383.

Peretzki, D. (2010). *Data mining for process identification (Diploma Thesis).* Cassel, Germany: University of Cassel. Retrieved from http://users.isy.liu.se/rt/andrecb/files/Peretzki_MSc10.pdf

Peretzki, D., Isaksson, A. J., Bittencourt, A. C., & Forsman, K. (2011). Data Mining of Historic Data for Process Identification. *Proceedings of the 2011 AIChE Annual Meeting.* Minneapolis, Minnesota, United States of America.

Shardt, Y. A. (2012). *Data Quality Assessment for Closed-Loop System Identification and Forecasting with Application to Soft Sensors.* Edmonton, Alberta, Canada: University of Alberta. doi:http://hdl.handle.net/10402/era.29018

Shardt, Y. A., & Shah, S. L. (2014). Segmentation Methods for Model Identification from Historical Process Data. *IFAC Proceedings Volumes. 47*, pp. 2836-2841. Cape Town, South Africa: Elsevier.

Shardt, Y. A., Yang, X., Brooks, K., & Torgashov, A. (2020). Data Quality Assessment for System Identification in the Age of Big Data and Industry 4.0. *IFAC World Congress (IFAC Papers Online)* (pp. 104-113). Berlin, Germany: Elsevier.