Identification and Modelling of Nonlinear Processes using Simulink S-Function: A Comprehensive Parametric and Non-Parametric Approach with LS, RLS, and Instrumental Variables Methods

Gašper Leskovec[a]

[a]*Faculty of Electrical Engineering, University of Ljubljana, Tržaška c. 25, 1000 Ljubljana, Slovenia.*

KEYWORDS

Identification
Modelling
Nonlinear process
Simulink S-Function
Parametric modelling
Non-parametric approach
Orthogonal correlation
Least squares (LS)
Recursive least squares (RLS)
Instrumental variables (IV)
FR analysis
Confidence band
TF estimation
Validation

ABSTRACT

This study aims to identify and model a nonlinear process given in the form of a Simulink S-function using parametric and nonparametric methods. The process under investigation can be linearly approximated in the vicinity of the operating point. Initially, the static curve of the process is measured to determine the operating interval for identification. A discrete transfer function model of the process is then identified using the least squares (LS) and recursive least squares (RLS) methods, with standard deviations of the parameters provided. The frequency response of the process is determined through empirical transfer function estimation using Fourier analysis for nonparametric modeling. For validation purposes, orthogonal correlation is applied to the nonparametric model. An additional identification method, the instrumental variables method, is employed to further analyse the process. The models obtained are validated and their errors are estimated. The paper presents the technical details of the solutions, along with discussions and conclusions. The study contributes to the understanding of nonlinear process identification and modeling using both parametric and nonparametric techniques, offering valuable insights for the application of these methods in various fields.

1 Introduction

The objective of this study is to identify a nonlinear process given in the form of a Simulink S-function using various identification techniques. The process is continuous, and its dynamics are analyzed using digital processing methods. The seminar paper is structured to provide a step-by-step approach to achieve this goal. In this study, I identify a continuous-time system represented by a provided Simulink S-function (**mexw64** file). The system is modeled using an S-Function block with a D/A converter (Zero-Order Hold) at the input and an A/D converter (To Workspace) at the output. The sampling time is determined based on the process dynamics and identification method. *Figure 1* illustrates the Simulink scheme with connected blocks for the given system.

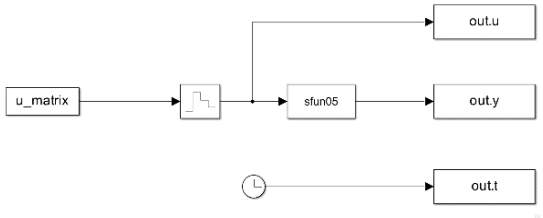

Figure 1 Simulink scheme with blocks connected together: S-Function block (sfun05), Zero-Order Hold block (enable digital processing), To Workspace block (output) and From Workspace block (input) and Clock block.

2 Measuring the Static Curve and Selection of Operating Interval:

The first step in the identification process is to measure the static curve of the nonlinear process. To minimize the effect of noise, the steady state is obtained by long-time averaging. The static curve provides insight into the operating interval where linear approximation is valid.

The code implements a simulation to obtain the static characteristic of a system. It begins by defining the sampling time (Ts) and creating a time vector (t) ranging from 0 to 10 with a step size of Ts. Arrays are then initialized to store the input (U_static) and output (Y_static) values.

Next, I load a Simulink model named **'for_static_characteristic'** and set up a figure for plotting the time series of the system's output. To analyze the system's behavior, I iterate over a range of input values (-5 to 5 with a step size of 0.05). For each input value, I create an input signal (U) as a time vector with a constant value U across all time points. The Simulink model is then simulated with this input signal, and the resulting output (y_out) is plotted against time (t_out) as shown in *figure 2*.

Corresponding Author: Tel: +0-000-000-000; fax: +0-000-000-000
Email Address: leskovecg@gmail.com

To characterize the system's response, I calculate the mean of the output values from time index 500 to the end and store it in the Y_static array. After the loop, I create a new figure to plot the static characteristic curve. This curve represents the relationship between the measured output values (Y_static) and the corresponding input values (U_static). I also include a red dashed line in the plot to illustrate a linear relationship between input and output, determined by the coefficients derived from the points (0.7, 0.048) and (1.3, 0.1564) as shown in *figure 3*.

Overall, this code allows me to analyze the time series behavior and static characteristics of the system under study.

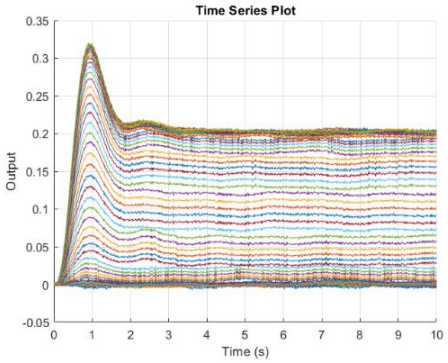

Figure 2 presents the time series plot of the system's output for various input values. The plot demonstrates the dynamic behaviour of the system as the input changes, allowing for analysis of its response over time.

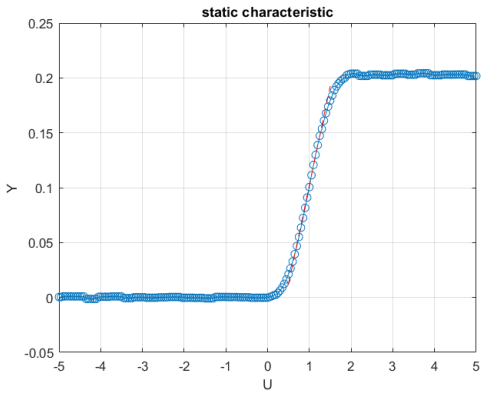

Figure 3 displays the static characteristic curve, showcasing the relationship between the measured output values (Y_static) and the corresponding input values (U_static). The red dashed line represents a linear relationship with coefficients derived from the data points (0.7, 0.048) and (1.3, 0.1564).

For the operating interval, I specifically chose an input range of 0.7 to 1.3 and an output range of 0.048 to 0.1564. This selection was made to ensure that the signal-to-noise ratio is improved, leading to a more accurate linear approximation of the process. These chosen ranges serve as the basis for the subsequent identification process.

3 Pseudo-random binary sequence (PRBS) as a verification signal, creation of a validation chirp signal, and frequency response of the PRBS signal

I created a pseudo-random binary sequence (PRBS) input signal, which will be used for the verification of different models. I began by loading the Simulink model **'for_all'** and defining the PRBS input signal, consisting of three parts: a starting part with constant values, a middle part with PRBS values, and an ending part with constant values again as shown in the *figure 4.*

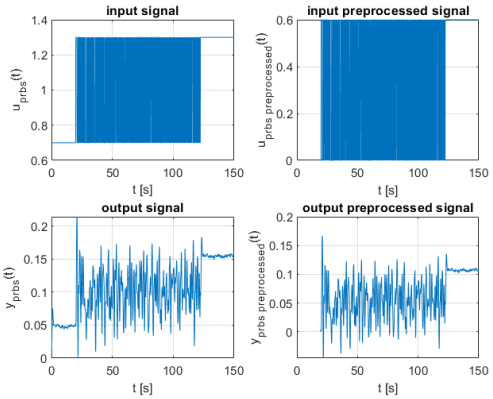

Figure 4 illustrates the input and output signals before and after pre-processing, revealing the changes in the system's behaviour.

3.1 Validation chirp signal

I created a chirp signal, which will be used for the validation of the models. I loaded the Simulink model **'for_all'** and defined the chirp signal parameters, including time parameters, sample interval, signal duration, constant samples, and frequencies. I generated a linear chirp signal with varying amplitudes and combined it with a constant signal.

I then simulated the model and obtained the output signals for the chirp signal. To preprocess the input and output signals, I removed their mean values and cut off an initial portion of the data. This was done to focus on the steady-state response of the system.

I visualized the input and output signals before and after pre-processing as shown in the *figure 5.* The pre-processed signals will be utilized for the validation of different models, providing valuable insights into their performance and characteristics in response to a chirp signal.

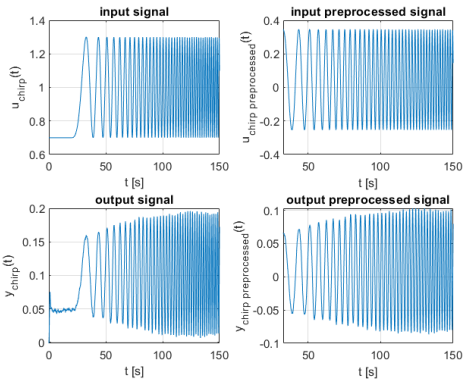

Figure 5 displays the input and output signals before and after pre-processing the chirp signal. These pre-processed signals will be used for validating different models, enabling the assessment of their performance and behaviour in response to a chirp signal.

3.2 Frequency response of the PRBS signal

I then analyse the frequency response of the system using the PRBS input signal.

Using the pre-processed signals, I computed the system's frequency response by performing a Fast Fourier Transform (FFT) on both the input and output signals. I calculated the system's transfer function by dividing the output FFT by the input FFT and converted the amplitude response to decibels (dB). I plotted the amplitude and phase responses of the system using semilogx plots, as shown in *figure 6*. This allowed me to observe the system's behavior across a range of frequencies, providing valuable insights into its performance and characteristics.

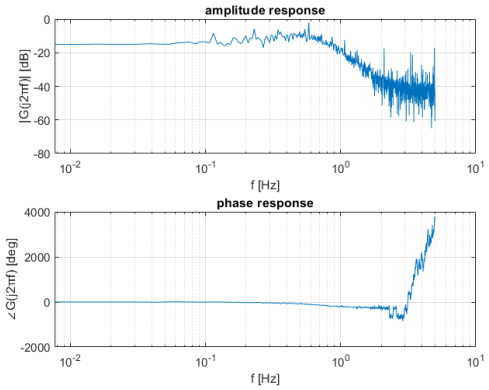

Figure 6 displays the amplitude and phase responses of the system, obtained by calculating the transfer function using Fast Fourier Transform (FFT) on the pre-processed input and output signals. These plots provide valuable insights into the system's performance and behavior across various frequencies.

4 Parametric Model Identification (LS):

I implemented parametric identification using the least squares (LS) method. Code begins by setting up the necessary variables and pre-processed signals for the PRBS verification signal. The input matrix PSI_for is constructed based on the system order, including delayed input and output values. The code then estimates the parameters of the transfer function model by solving a linear equation using the LS method. The numerator (b) and denominator (a) coefficients of the estimated transfer function are extracted. Next, the frequency response of the identified model is computed using the 'bode' function, and the magnitude and phase response are stored in variables (mag_db and phase_dgr) as shown in *figure 7*. The code plots a comparison between the measured output (y) and the simulated output (y_sim) using the PRBS signal, providing visual verification of the identified model as shown in *figure 8*. The identified model is used to simulate the output response (y_sim_val) using the preprocessed validation chirp input signal. A comparison plot between the measured output (y_validation_preprocessed) and the simulated output (y_sim_val) using the chirp signal is generated as shown in *figure 9*. The code also calculates the standard deviations of the estimated parameters as shown in *table 1*. It computes the residuals, variance of residuals (sigma2), covariance matrix of the estimated parameters (cov_theta), variances of estimated parameters (var_theta), and the standard deviation of the estimated parameters (std_theta). The standard deviations provide an indication of the uncertainty associated with the estimated parameter values.

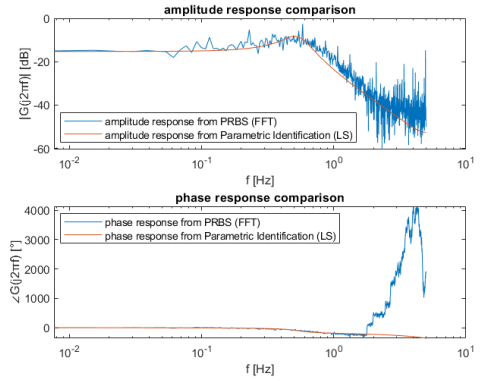

Figure 7 shows a comparison of the amplitude and phase responses obtained from the PRBS signal using FFT and the parametric identification (LS) method. The upper subplot displays the amplitude response, while the lower subplot shows the phase response. The comparison allows for evaluating the agreement between the identified model and the PRBS signal in terms of amplitude and phase characteristics across different frequencies

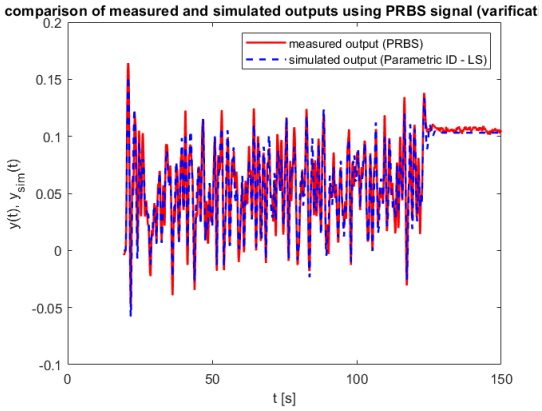

Figure 8 showcases a comparison between the measured output (y) and the simulated output (y_sim) using the PRBS signal for verification. The red line represents the measured output, while the blue dashed line corresponds to the simulated output obtained through the parametric identification (LS) method.

The plot allows for visual assessment of the agreement and accuracy between the measured and simulated outputs over the given time duration.

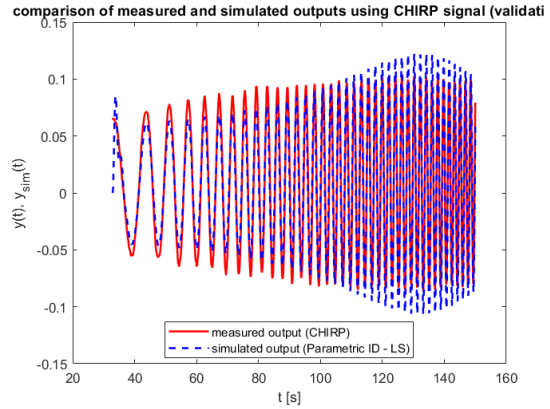

Figure 9 presents a comparison between the measured output (y_validation_preprocessed) and the simulated output (y_sim_val) using the chirp signal for validation. The red line represents the measured output, while the blue dashed line corresponds to the simulated output obtained through the parametric identification (LS) method. The plot allows for visual assessment of the agreement and accuracy between the measured and simulated outputs during the validation period using the chirp signal.

Table 1 presents the standard deviation values (LS) for the estimated parameters obtained through the LS method. These standard deviations quantify the uncertainty and variability associated with each parameter, providing insights into the reliability and robustness of the estimated values.

standard deviation of the parameters (LS)	
a1	0.0083
a2	0.0082
b1	0.0003
b2	0.0003

Table 2 includes the computed error metrics (LS) for the obtained model using both the training and validation data. It presents the root mean square error (RMSE) and mean absolute error (MAE) values, which quantify the discrepancy between the measured outputs.

data	estimated error of the obtained model (LS)	
	RMSE	MAE
training data	0.0092	0.0071
validation data	0.0139	0.0118

5 Nonparametric Model Identification:

I implemented a non-parametric identification method known as orthogonal correlation to estimate the frequency response of a process. The frequency response describes how the process behaves at different frequencies. In this code, the process is excited with input signals generated using cosine and sine functions at various frequencies. The output signals are then recorded and pre-processed to remove any unwanted components.

Next, the code calculates the amplitude and phase of the frequency response using the orthogonal correlation technique. This involves analysing the relationship between the input and output signals at each frequency. By measuring the correlation between these signals, the code determines the amplitude (magnitude) and phase shift of the process at different frequencies.

To validate the identified model, the code compares the estimated frequency response with the true response. It plots both the identified and true frequency responses, showing the amplitude and phase values at different frequencies shown in *figure 10*. Additionally, a confidence band is included around the identified amplitude response, indicating the estimated standard deviation of the amplitude response error (see *figure 23*).

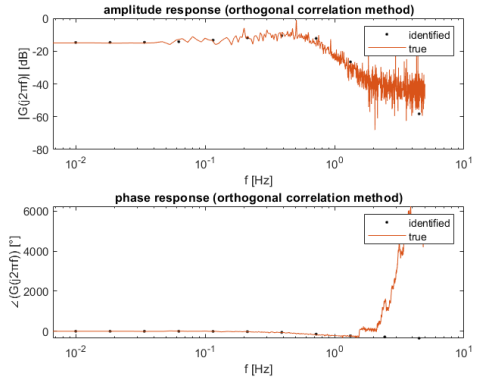

Figure 10 displays the comparison between the identified and true frequency response using the orthogonal correlation method. The top subplot shows the amplitude response, with the identified response represented by black dots and the true response represented by the solid line. The bottom subplot presents the phase response, again with the identified response shown as black dots and the true response displayed as the solid line.

6 Alternative Identification Method (RLS):

I implemented Recursive Least Squares (RLS) for parametric identification of a discrete-time system using the pre-processed PRBS (Pseudo-Random Binary Sequence) input signal u and output signal y. The code constructs the input matrix PSI_for based on the system order n_s and the delay d. Then, it applies the RLS method to estimate the system parameters a and b. The code also plots the live parameter updates for a1, a2, b1, and b2 as shown in *figure 14*.

The identified system is represented as a transfer function G, and the code computes the Bode plot (magnitude and phase) of the identified system as shown in *figure 11*. It then simulates the output of the identified system (y_sim and y_sim_val) using the PRBS signal for verification and the CHIRP signal for validation (*figure 12 and 13*).

Code computes the Root Mean Square Error (RMSE) and Mean Absolute Error (MAE) for both the training and validation data and displays the results (*table 4*).

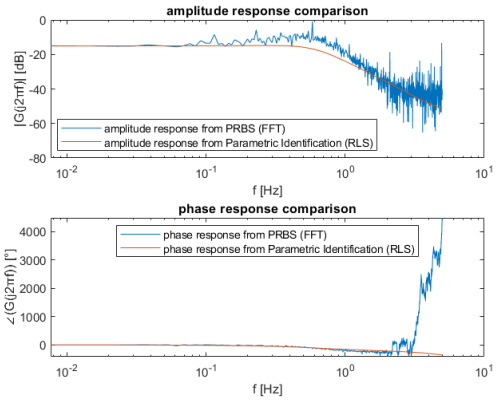

Figure 11 represents the comparison of the amplitude and phase responses obtained from the PRBS (FFT) method and the Parametric Identification (RLS) method. The two subplots showcase the frequency response of the system, highlighting the effectiveness of both methods in capturing the system's dynamic behavior.

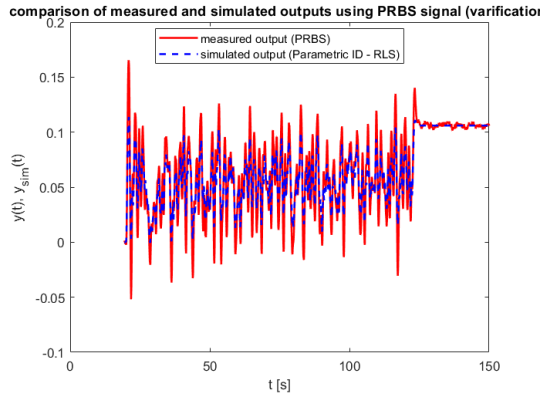

Figure 12 represents the comparison between the measured output y and the simulated output y_sim obtained from the identified system using the PRBS signal for verification. It showcases the performance of the Parametric ID - RLS method in modeling the system's behavior.

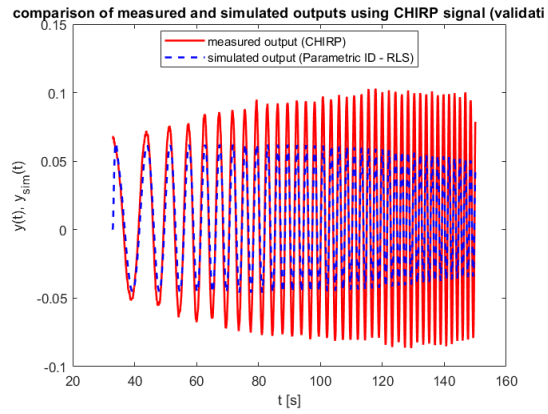

Figure 13 represents the comparison between the measured output y_validation_preprocessed and the simulated output y_sim_val obtained from the identified system using the CHIRP signal for validation. It highlights the effectiveness of the Parametric ID - RLS method in predicting the system's response to a different input signal.

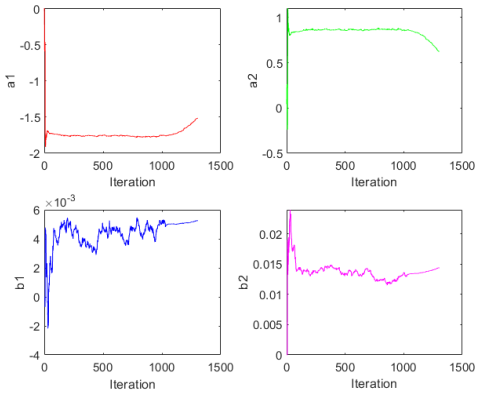

Figure 14 represents the live parameter updates for the estimated parameters a1, a2, b1, and b2 during the Recursive Least Squares (RLS) estimation process. The four subplots visualize the convergence of the algorithm and the stability of the parameters as the iterations progress.

Table 3 represents the standard deviation of the estimated parameters a and b obtained from the Recursive Least Squares (RLS) method, providing insight into the variability of the parameters during the estimation process.

standard deviation of the parameters (RLS)	
a1	0.1085
a2	0.1138
b1	0.0006
b2	0.0013

Table 4 represents the Root Mean Square Error (RMSE) and Mean Absolute Error (MAE) for the training and validation datasets, which are used to evaluate the performance of the obtained model using the Recursive Least Squares (RLS) method. These error metrics provide an assessment of the model's accuracy in representing the system.

data	estimated error of the obtained model (RLS)	
	RMSE	MAE
training data	0.0137	0.0101
validation data	0.0234	0.0200

7 Additional Identification Method (IV):

Finally, I implemented parametric system identification of a linear time-invariant (LTI) system using the Instrumental Variables (IV) method. The system is excited by a PRBS (Pseudo-Random Binary Sequence) signal, and the response is pre-processed. The code estimates the parameters of the system transfer function and compares the performance of the identified model against measured data.

In the first part, the input and output data are pre-processed, and the input matrix PSI is constructed based on the system order. Next, the parameter estimates are obtained, and the transfer function is calculated. Using the identified model, the simulated response is computed, and the model is validated against

measured data (*figure 16* and *figure 17*). The figures show comparisons of measured and simulated outputs using PRBS and CHIRP signals for verification and validation, respectively.

The code then constructs the instrumental variables W matrix and estimates the parameters using the IV method. The identified model using the IV method is compared with measured data in *figure 18* and *figure 19*, like the previous figures. The amplitude and phase responses are compared in *figure 15*, which shows the Bode diagrams of the identified models.

The standard deviation of the estimated parameters is calculated to assess the uncertainty of the parameter estimates (*table 5*). Furthermore, the error of the obtained model is estimated for both training and validation data, including the calculation of the Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE) in *table 6*.

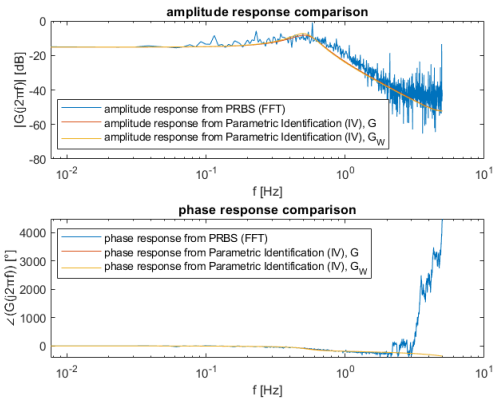

Figure 15 represents a comparison of the amplitude and phase responses for the system using different identification techniques. It displays the frequency response obtained from PRBS (FFT) and the responses from G and G_W models. The subplots provide a visual assessment of the similarities and differences between the identification techniques in both amplitude and phase domains.

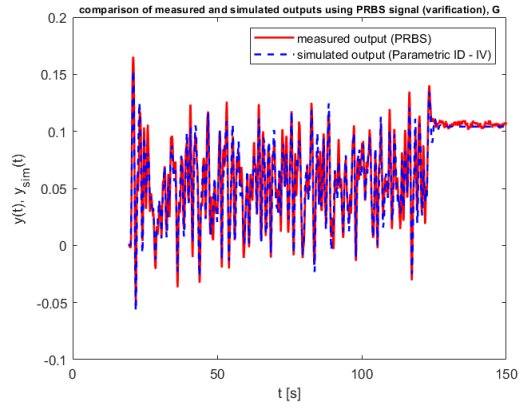

Figure 16 represents a comparison between the measured output (PRBS) and the simulated output (Parametric ID - IV) for the identified transfer function G. It visually demonstrates how well the identified model approximates the actual system response during the verification process.

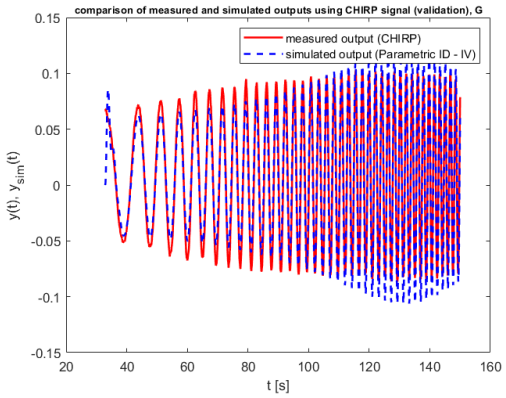

Figure 17 represents a comparison between the measured output (CHIRP) and the simulated output (Parametric ID - IV) for the identified transfer function G. It visually evaluates the model's performance in approximating the actual system response when subjected to a different input signal, the CHIRP signal, during the validation process.

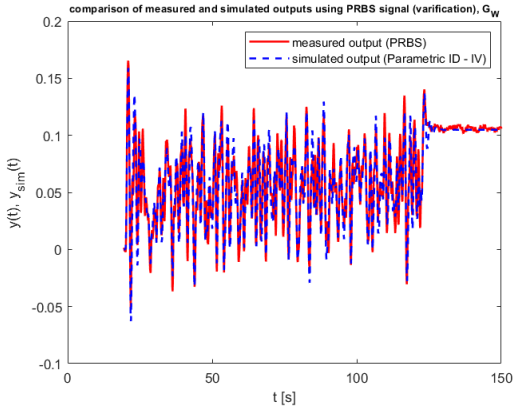

Figure 18 represents a comparison between the measured output (PRBS) and the simulated output (Parametric ID - IV) for the identified transfer function G_W. It visually demonstrates the effectiveness of the G_W model in approximating the actual system response when subjected to a PRBS signal during the verification process.

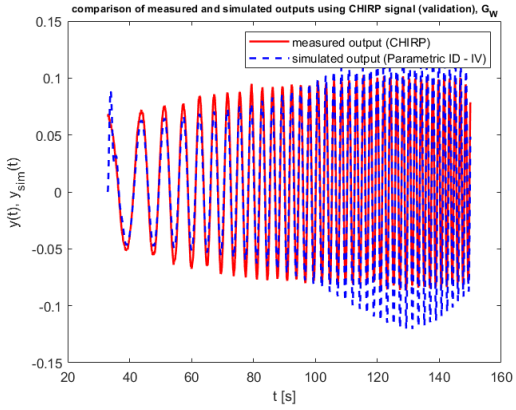

Figure 19 represents a comparison between the measured output (CHIRP) and the simulated output (Parametric ID - IV) for the identified transfer function G_W. It visually evaluates the G_W model's ability to approximate the actual system response when subjected to a different input signal, the CHIRP signal, during the validation process.

Table 5 represents the standard deviation of the estimated parameters for the Instrumental Variables (IV) method. It provides a quantitative measure of the uncertainty or variability associated with the model's parameter estimates, helping to assess the reliability and precision of the model obtained through this identification technique.

standard deviation of the parameters (IV)	
a1	0.0277
a2	0.0273
b1	0.0008
b2	0.0009

Table 6 represents the estimated errors of the obtained model using the Instrumental Variables (IV) method, G. It showcases the Root Mean Square Error (RMSE) and Mean Absolute Error (MAE) for both training and validation data. This table provides a way to evaluate the model's performance by quantifying the discrepancies between the model's predictions and the actual measurements, helping to understand the model's reliability and accuracy.

data	estimated error of the obtained model (IV), G	
	RMSE	MAE
training data	0.0093	0.0071
validation data	0.0139	0.0118

Table 7 represents the performance evaluation of the obtained model using the Instrumental Variables (IV) method, G_W. It showcases the Root Mean Square Error (RMSE) and Mean Absolute Error (MAE) for both training and validation data sets, providing an insight into the model's accuracy and reliability. These error metrics help quantify the model's ability to predict system outputs based on the input signals, ultimately determining its effectiveness for the given application.

data	estimated error of the obtained model (IV), G_W	
	RMSE	MAE
training data	0.0142	0.0107
validation data	0.0255	0.0197

8 Results and Discussion:

The results of each identification method are presented and discussed. A comparison is made to determine the most accurate model for the given process.

data	estimated error of the obtained model (LS)	
	RMSE	MAE
training data	0.0092	0.0071
validation data	0.0139	0.0118
data	**estimated error of the obtained model (RLS)**	
	RMSE	**MAE**
training data	0.0137	0.0101
validation data	0.0234	0.0200
data	**estimated error of the obtained model (IV), G**	
	RMSE	**MAE**
training data	0.0093	0.0071
validation data	0.0139	0.0118
data	**estimated error of the obtained model (IV), G_W**	
	RMSE	**MAE**
training data	0.0142	0.0107
validation data	0.0255	0.0197

The results presented in the table provide a comparison of the performance of three different system identification methods: Least Squares (LS), Recursive Least Squares (RLS), Instrumental Variables (IV) with G, and Instrumental Variables (IV) with G_W. The performance is evaluated based on Root Mean Square Error (RMSE) and Mean Absolute Error (MAE) for both training and validation data sets.

From the table, we can observe the following:

1. For the LS method, the estimated errors are relatively low for both training and validation data, with RMSE values of 0.0092 and 0.0139, and MAE values of 0.0071 and 0.0118, respectively. This indicates that the LS method provides a good fit to the data.

2. The RLS method shows slightly higher errors compared to the LS method. The RMSE values for the training and validation data are 0.0137 and 0.0234, while the MAE values are 0.0101 and 0.0200, respectively. This suggests that the RLS method may not perform as well as the LS method for this particular system.

3. The IV method with G shows similar performance to the LS method, with RMSE values of 0.0093 and 0.0139 and MAE values of 0.0071 and 0.0118 for the training and validation data, respectively. This indicates that the IV method with G is also a good fit for the data.

4. Lastly, the IV method with G_W has the highest errors among the four methods. The RMSE values for the training and validation data are 0.0142 and 0.0255, and the MAE values are 0.0107 and 0.0197, respectively. This suggests that the IV method with G_W may not be as accurate as the other methods for this specific system.

In summary, the LS seem to perform the best for this particular system, offering the lowest errors for both training and validation data sets. The RLS and IV methods show slightly higher errors, which may suggest that they are less suitable for this specific application.

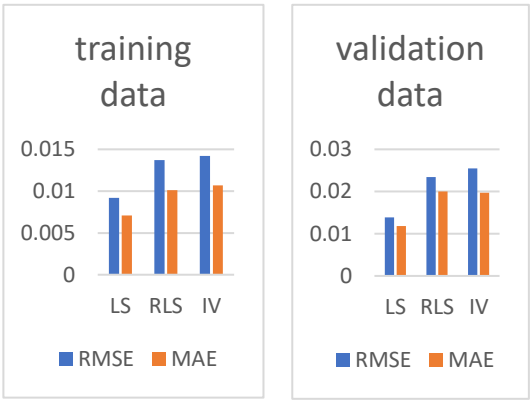

Figure 20 graphs presents the performance of three system identification methods (LS, RLS, IV) based on RMSE and MAE values for training and validation data sets. The LS method yield the lowest errors, indicating the best fit for this system, while the RLS and IV methods exhibit higher errors, suggesting they may be less suitable for this particular application.

9 Conclusions:

In conclusion, the system identification methods compared in this study – Least Squares (LS), Recursive Least Squares (RLS), Instrumental Variables (IV) – demonstrate varying levels of performance when applied to this particular system. The LS method stand out as the most accurate, providing the lowest errors for both training and validation data sets. However, the RLS and IV methods exhibit higher errors, indicating that they may not be the best fit for this specific application.

It is essential to acknowledge that the selection of a suitable system identification method depends on the particular system characteristics, underlying assumptions, and the quality of the available data. As a result, it is crucial to carefully consider the specific needs and constraints of the system under investigation when selecting an appropriate identification method.

10 Appendix - NOISE:

In the first part of the code, a noisy input signal is generated and used to simulate a Simulink model. The output signal, which contains the noise component, is then processed to extract the noise. The code trims the signals to remove any initial transient behaviour and calculates the mean value of the output signal to center it around zero.

Next, the code calculates the frequency domain representation of the noise signal using the Fast Fourier Transform (FFT). The frequency vector and half of the FFT spectrum are computed to plot the noise FFT. Additionally, a model of the noise is defined based on low-frequency (N0) and high-frequency (Ninf) amplitudes, as well as a frequency parameter (wg).

In *figure 21*, three subplots are displayed. The first subplot shows the input signal, which is a constant value over time. The second subplot presents the output signal, which represents the noise. Finally, the third subplot compares the FFT of the noise signal with the model of the noise.

Moving on, the code focuses on the analysis of standard deviation. It calculates the measured standard deviation of the absolute error between the noise FFT and the FFT of the reference signal (Y_prbs). Additionally, it computes the theoretical standard deviation based on the noise model defined earlier.

Figure 22 is plotted to visualize the comparison between the measured and theoretical standard deviations of the absolute error. The x-axis represents the frequency (w) in Hz, while the y-axis corresponds to the standard deviation of the absolute error. The blue curve represents the measured standard deviation, while the red curve represents the theoretical (model) standard deviation.

Finally, in *figure 23*, the code demonstrates the application of the standard deviation of the absolute error to the reference signal. The middle curve (red) represents the reference signal in decibels (dB). The lower curve (blue) corresponds to the middle curve plus the theoretical standard deviation, and the upper curve (green) corresponds to the middle curve minus the theoretical standard deviation.

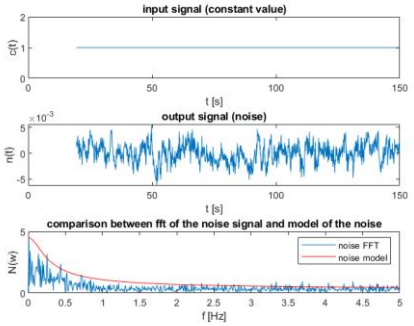

Figure 21 consists of three subplots: the first plot shows the input signal (constant value) over time, the second plot displays the output signal (noise), and the third plot compares the FFT of the noise signal with the model of the noise. The plots provide a visual representation of the input and output signals, as well as the comparison between the measured noise FFT and the modeled noise.

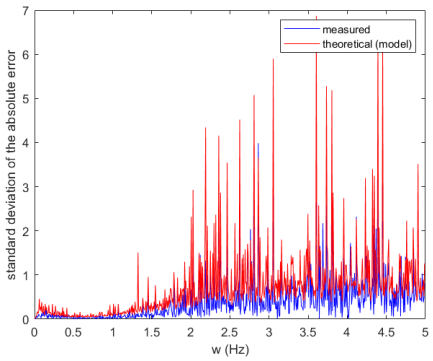

Figure 22 presents the standard deviation of the absolute error as a function of frequency. It includes two curves: the blue curve represents the measured standard deviation, while the red curve represents the theoretical (model) standard deviation. The figure allows for a visual comparison between the measured and theoretical estimates of the error in the system.

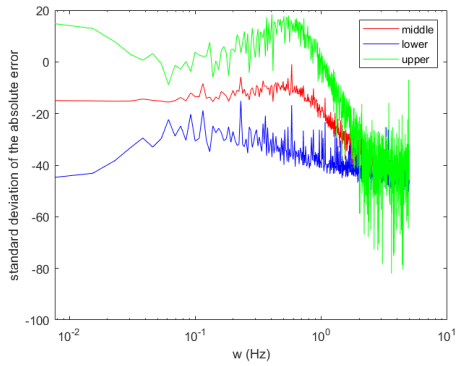

***Figure 23** presents the standard deviation of the absolute error in decibels (dB) as a function of frequency on a logarithmic scale. The figure includes three curves: the red curve represents the middle value of the error, while the blue and green curves depict the upper and lower bounds, respectively, based on the theoretical standard deviation. This figure visually illustrates the range of error values around the middle value in the frequency domain.*