Estimation and Learning in Aerospace 2024/2025 Project

Baniamein Mina, Bellezza Alessandro,

Castelanelli Oddo Alessandro

POLITECNICO
MILANO 1863

ASCL
AEROSPACE SYSTEMS & CONTROL LABORATORY

Table of contents

Physical Parameters

Weight: 270g
Dimensions: 20 x 20 x 4 cm
Diameter (motor-to-motor distance): 16cm
Hovering time: 7’30’’

Linearised model equations for longitudinal dynamics:

State equations:

$$\begin{bmatrix} \dot{u} \\ \dot{q} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} X_u & X_q & -g \\ M_u & M_q & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} u \\ q \\ \theta \end{bmatrix} + \begin{bmatrix} X_\delta \\ M_\delta \\ 0 \end{bmatrix} \delta_{lon}$$

- u Longitudinal (body) velocity [m/s]
- q Pitch rate [rad/s]
- θ Pitch angle [rad]
- a_x Longitudinal (body) acceleration [m/s^2]
- δ_{lon} Pitch moment [normalised to -1, +1 range]

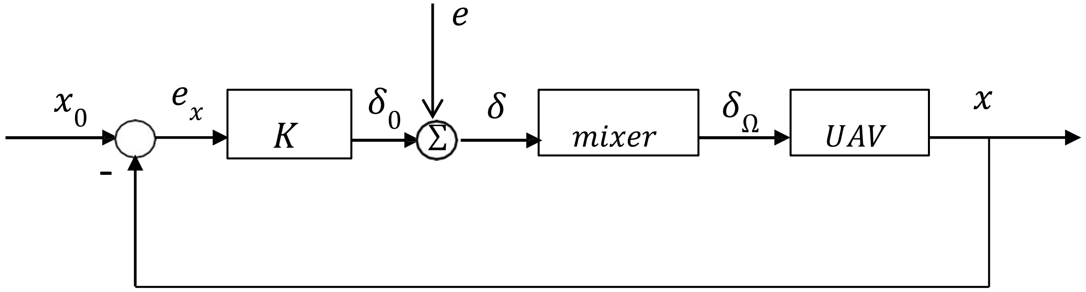
Decoupled dynamics

- *Longitudinal dynamics*
  - Input: δ_{lon} / Outputs: q

Output equations:

$$\begin{bmatrix} q \\ a_x \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ X_u & X_q & 0 \end{bmatrix} \begin{bmatrix} u \\ q \\ \phi \end{bmatrix} + \begin{bmatrix} 0 \\ X_\delta \end{bmatrix} \delta_{lon}$$

Model parameters: $\Theta = \begin{bmatrix} X_u \\ X_q \\ M_u \\ M_q \\ X_\delta \\ M_\delta \end{bmatrix} = \begin{bmatrix} -0.1068 \\ 0.1192 \\ -5.9755 \\ -2.6478 \\ -10.1647 \\ 450.71 \end{bmatrix}$



Task 1.1: grey box identification from sweep

Use the simulator and the provided baseline excitation to:

- T1.1
  - Identify a grey-box model for the longitudinal dynamics of the multirotor (input: total pitching moment; output: pitch rate) using the response to the sweep provided in the ExcitationM.mat file
- T1.2
  - Assess the uncertainty of the identified model in terms of asymptotic variance of the parameter estimates
- T1.3
  - Generate a suitable 3211 sequence and use it to validate the identified model

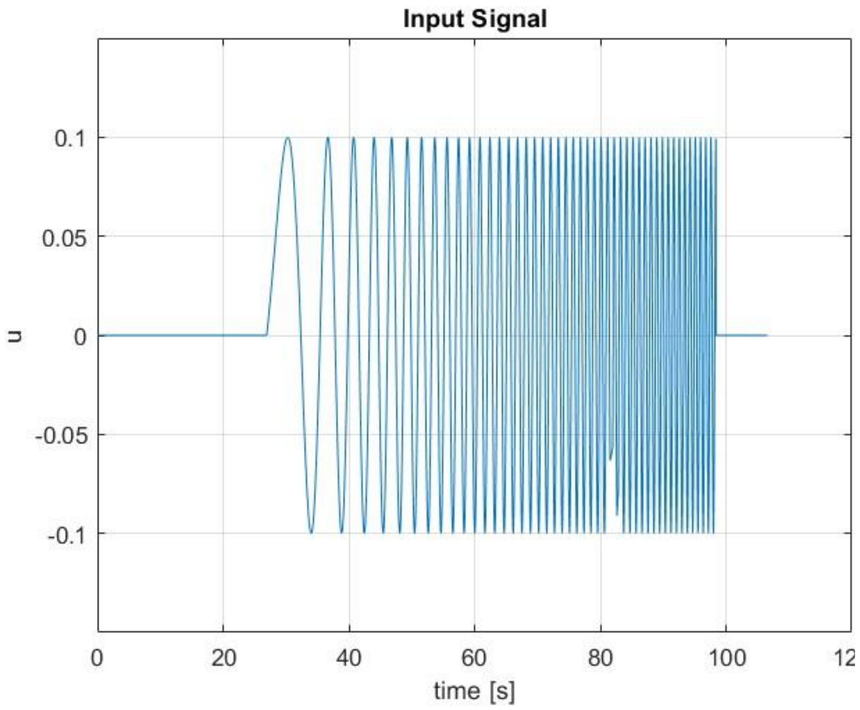
Notes:

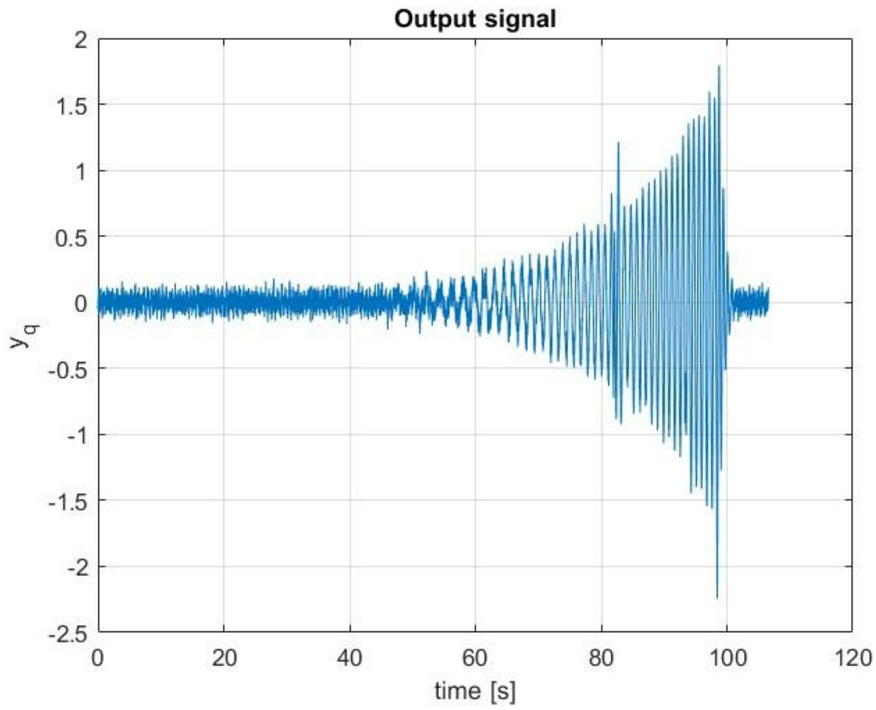
- the open loop dynamics is *unstable*
- MATLAB functions *iddata* and *greyest* can be used for the implementation

Task 1.1: grey box identification from sweep

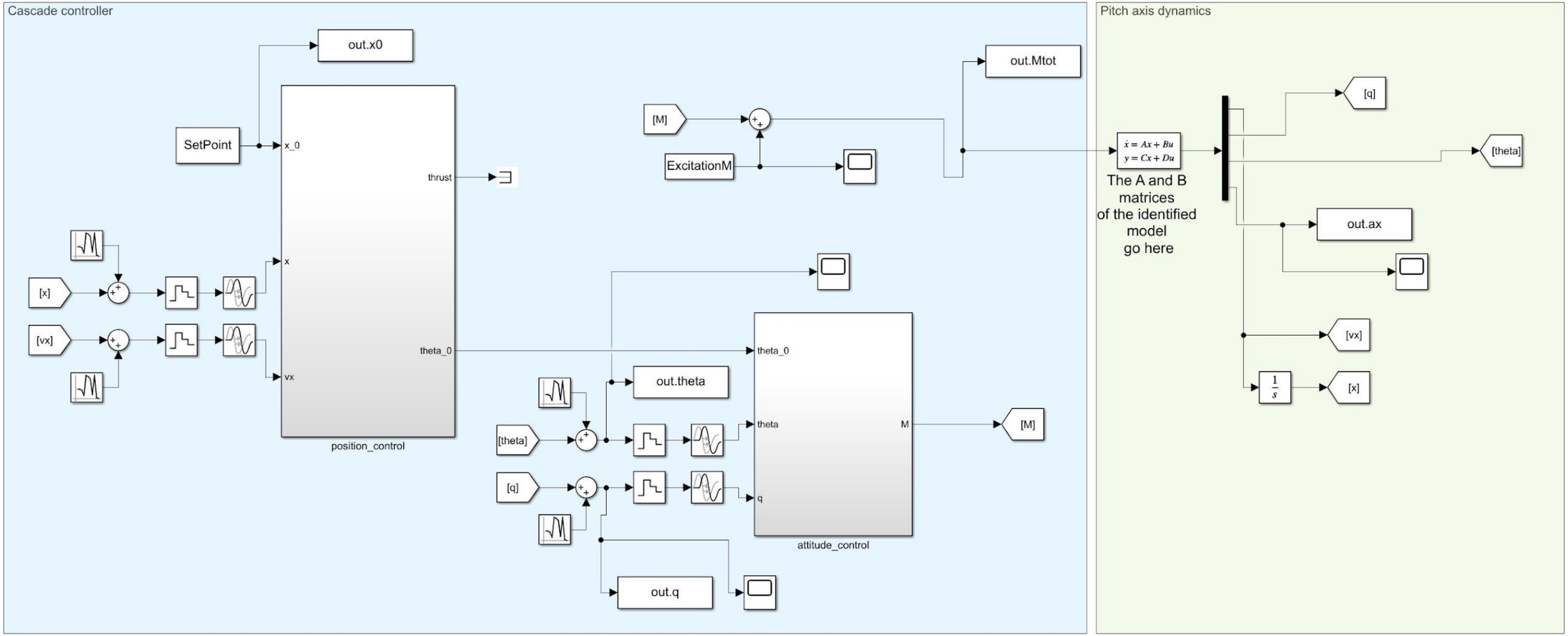
Procedure:

1. A long duration sweep around the pitching moment is applied to excite low-frequency translational dynamics
2. The response output values of q during time are collected
3. Input and output data are used to perform the model identification through the grey box method





Simulator



Cascade controller
out.x0
SetPoint
x_0
thrust
x
vx
theta_0
position_control
[x]
[vx]
[M]
ExcitationM
out.Mtot
out.theta
[theta]
[q]
theta_0
theta
q
M
attitude_control
[M]
out.q
Pitch axis dynamics
$\dot{x} = Ax + Bu$
$y = Cx + Du$
The A and B matrices of the identified model go here
[q]
[theta]
out.ax
[vx]
$\frac{1}{s}$
[x]

Task 1.1: grey box algorithm description

1. Input Data Preparation

1. Load simulation outputs: acceleration (ax) and pitch rate (q).
2. Load control input: total moment (Mtot)
3. Arrange data as input–output pairs.

2. Transform Data

1. Build an iddata object
2. Transform signals to the frequency domain (FFT).

```
% Data ordering and bring those in frequency domain
sim_data = iddata(y, u, sample_time);
data_fd = fft(sim_data); % output of the simulation in the frequency domain

% Initial guess for the identification
sys_init = idgrey(model_fun, guess, 'c');

% Actual Model Identification
identification = struct;
estimated_model = greyest(data_fd, sys_init);
```

3. Model Setup

1. Define initial parameter guess
2. Create a grey-box model (idgrey) using the provided model function.

```
function [A,B,C,D] = drone_model_grey(params,~)

% Parameters definition
Xu= params(1);
Xq = params(2);
Mu = params(3);
Mq = params(4);
Xd = params(5);
Md = params(6);

% Model definition
A=[Xu, Xq, -9.81; Mu, Mq, 0; 0, 1, 0];
B=[Xd; Md; 0];
C=[0, 1, 0; Xu, Xq, 0];
D=[0; Xd];

end
```

4. Model Estimation

1. Run grey-box estimation (greyest) on the frequency-domain data.
2. Extract:
   a. *Identified parameters*
   b. *Fit percentage*
   c. *Final Prediction Error (FPE)*
   d. *Covariance of estimated parameters*
   e. *State-space matrices (A, B, C, D)*

Task 1.1: grey box identification from sweep

Input guess: randomly perturbed assigned initial parameters
Results with one output signal:

Parameter	θ_{true}	θ_{grey}
Xu	-0.1068	-0.0955
Xq	0.1192	0.1225
Mu	-5.9755	-5.9032
Mq	-2.6478	-2.6815
Xd	-10.1647	-10.4862
Md	450.71	450.7453

The values estimated are near to the true one, with a FIT value of 86.1942 % and an angular rate standard deviation σ of 0.0175

Uncertainty assessment procedure:

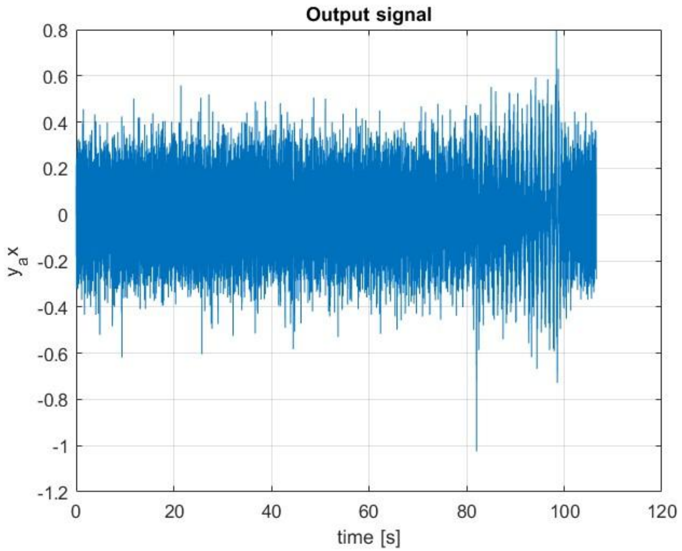
1. Retrieve covariance matrix of estimated parameters
2. Extract diagonal root σ terms
3. Eventually compute 3 σ gaussian confidence intervals

Results with 1 output:

Parameter	θ_{true}	θ_{grey}	σ
Xu	-0.1068	-0.0955	2.7118933×10^{6}
Xq	0.1192	0.1225	1.1879804×10^{6}
Mu	-5.9755	-5.9032	3.658942×10^{-1}
Mq	-2.6478	-2.6815	2.7118869×10^{6}
Xd	-10.1647	-10.4862	2.0706793×10^{8}
Md	450.71	450.7453	1.20484053×10^{1}

Some parameters exhibit a very high uncertainty value, making it hard to be uniquely identifiable

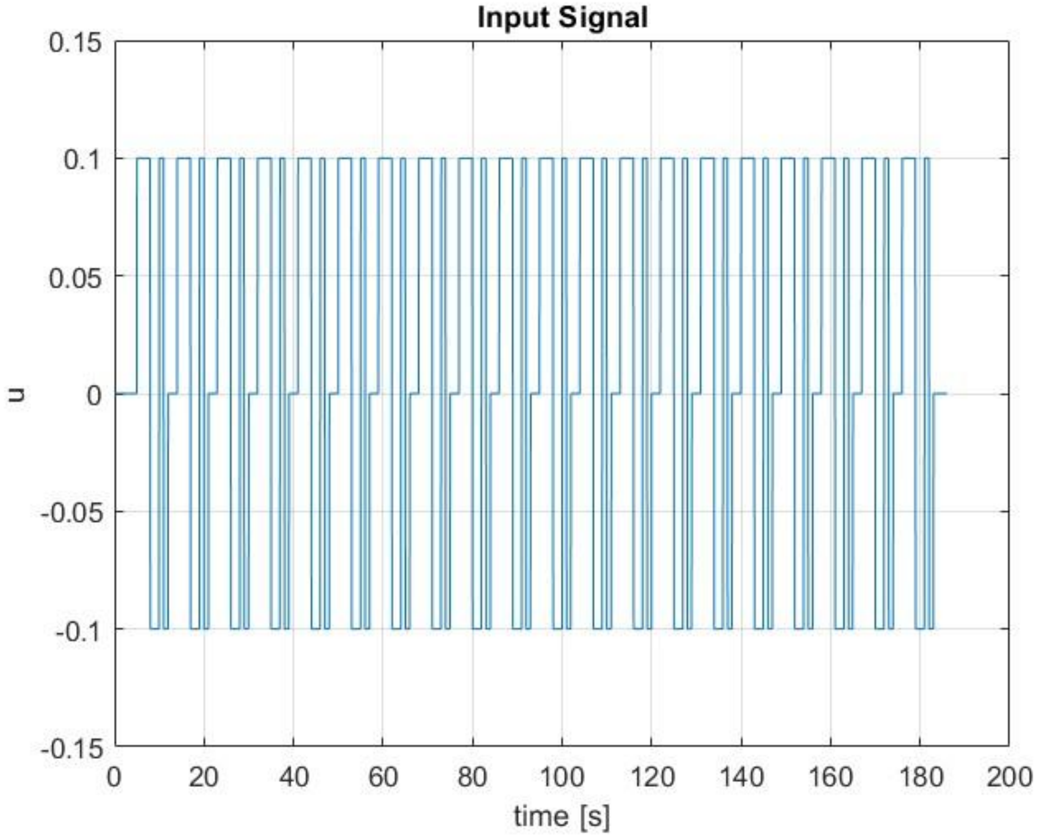
To improve the parameters estimation a second output is considered, $\mathbf{a_x}$:

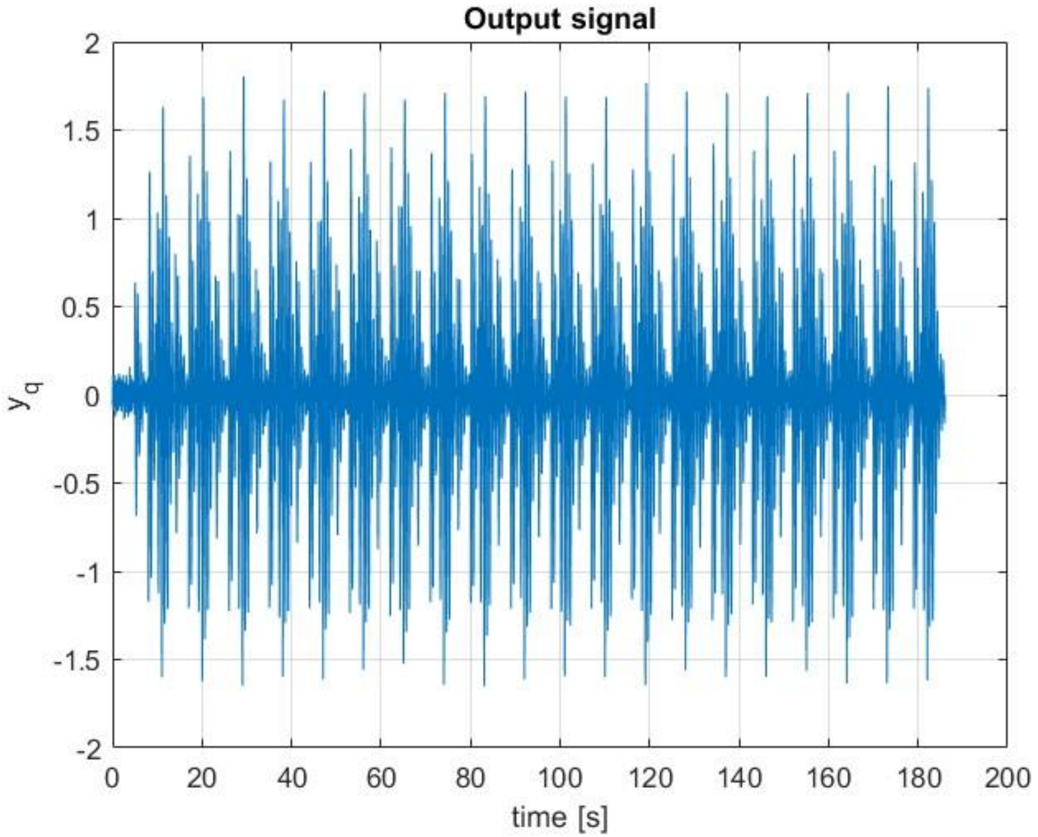


Results with 2 outputs:

Parameter	θ_{true}	θ_{grey}	σ	-3σ	$+3\sigma$
Xu	-0.1068	-0.1068	0.0001	-0.1078	-0.1056
Xq	0.1192	0.1192	0.0001	0.1191	0.1193
Mu	-5.9755	-5.9750	0.0010	-6.0796	-5.8317
Mq	-2.6478	-2.6480	0.0003	-2.6618	-2.6374
Xd	-10.1647	-10.1647	0.0000	-10.1651	-10.1644
Md	450.71	450.8380	0.2805	450.4855	450.8603

Generated an extended u3211 command sequence in time to improve the parameters estimation process





Task 1.3: 3211 validation

Results with 2 outputs:

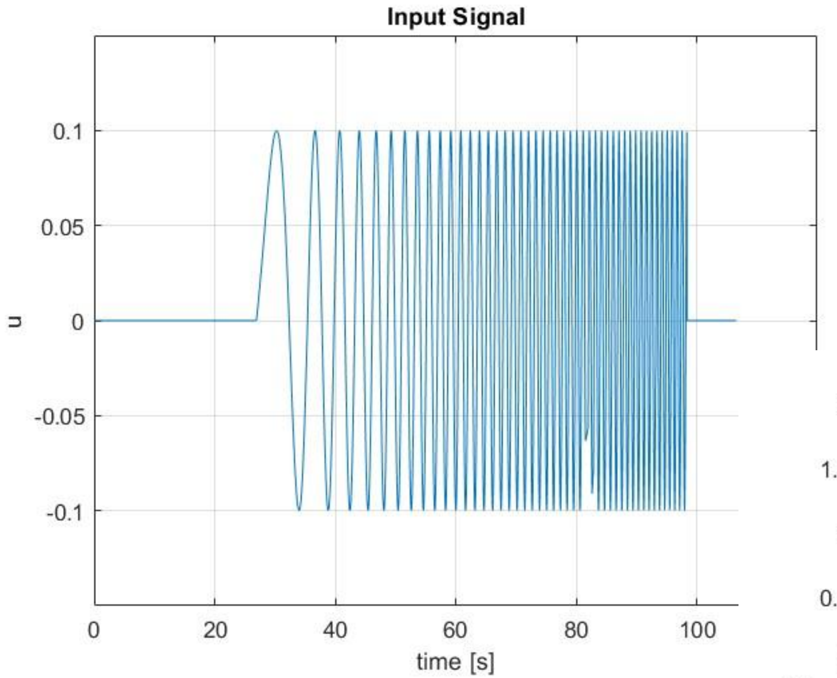
Parameter	θ_{true}	θ_{u3211}	θ_{grey}	$-3\sigma_{u3211}$	$3\sigma_{u3211}$	σ_{u3211}
Xu	-0.1068	-0.1067	-0.1068	-0.1078	-0.1056	0.0006
Xq	0.1192	0.1192	0.1192	0.1191	0.1193	0.0001
Mu	-5.9755	-5.9556	-5.9750	-6.0796	-5.8317	0.0632
Mq	-2.6478	-2.6496	-2.6480	-2.6618	-2.6374	0.0062
Xd	-10.1647	-10.1647	-10.1647	-10.1651	-10.1644	0.0002
Md	450.7100	450.6729	450.8380	450.4855	450.8603	0.0956

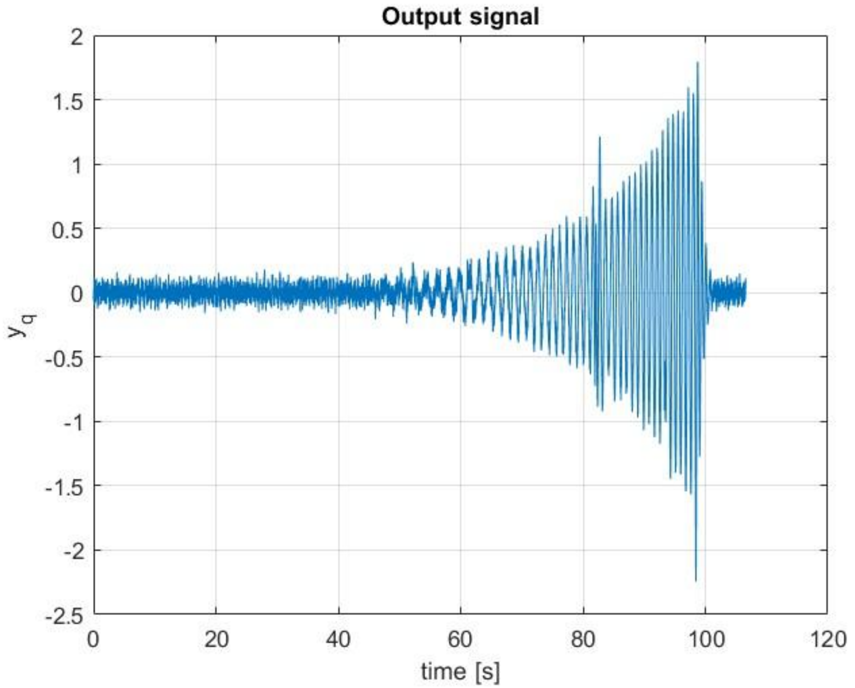
Task 2: black-box model identification

- T2.1
  - Implement the PBSID algorithm, as presented in class
  - Implement the structuring method using either *greyest* or *hinfstruct*
- T2.2
  - Validate the implementations of the algorithms using a simple first order example and an experiment of your choice
- T2.3
  - Identify a black-box model for the longitudinal dynamics of the multirotor (input: total pitching moment; outputs: pitch rate) using the implemented PBSID. The response to the sweep provided in the ExcitationM.mat file should be used for parameter estimation and for choice of *n*, *p*, *f* in cross-validation
- T2.4
  - Construct a structured model consistent with the linearised state space model class $M(\Theta)$ using the structuring method implemented in T2.1
- T2.5
  - Use the 3211 sequence generated in T1.3 to validate the identified model

Procedure:

1. A long-duration sine sweep is applied as the input to excite the multirotor longitudinal dynamics across a wide frequency range.
2. The system's response is recorded, primarily capturing the pitch rate (**q**) and longitudinal acceleration ($\mathbf{a_x}$).
3. Input and output **q** are used to perform a SISO model identification throughout the PBSID method





Task 2.2 : Black-Box Identification Workflow

1. Data Acquisition & Preparation	2. Fine-tune parameters	3. PBSID Model Identification	4. Structuring Approach	5. Validation

1. Data Acquisition & Preparation

In order to acquire high-quality data and prepare it for identification.

- Excited the system using a sine sweep
- Split the dataset to perform a further cross-validation
- Performed spectral analysis
- Applied normalization and filters to clean and smooth the signals.

2. Fine-tune parameters

In order to find the optimal parameters for the PBSID algorithm.

- Assumption that future window (**f**) equal to the past window (**p**).
- A linear search to find the optimal **p** minimizing the simulation error

3. PBSID Model Identification

In order to compute the initial black-box model.

- Estimated the system order **n** by analysing the singular value spectrum (SVD)
- Executed the PBSID algorithm with the optimal **p** and the estimated order **n**.
- Obtained the initial numerical state-space matrices

4. Structuring Approach

In order to convert the numerical black-box model into a meaningful model.

- Used the frequency response as the fitting target
- Defined the known grey-box structure with physical parameters
- Estimated the physical parameters using MATLAB's *greyest* function.

5. Validation

In order to verify the accuracy of the final model.

- Validated the final structured model with a 3-2-1-1 sequence.
- Compared the model's time response, Bode plot, and Pole-Zero map to the true system.
- Quantified the final accuracy

Task 2.3 : Theoretical Foundations: The State-Space Model

1. General State-Space Model

We start by considering a linear, time-invariant (LTI) system described by the following discrete-time state-space equations:

$$\mathbf{x}_{k+1} = A\mathbf{x}_k + B\mathbf{u}_k + \mathbf{w}_k$$
$$\mathbf{y}_k = C\mathbf{x}_k + D\mathbf{u}_k + \mathbf{v}_k$$

2. Kalman Innovation Form

For identification purposes, it's convenient to rewrite the model using the innovation sequence ($\mathbf{e}_k = \mathbf{y}_k - \hat{\mathbf{y}}_{k|k-1}$). The innovation represents the unpredictable part of the output.
This will leads to the **Kalman Innovation State-Space Model** :

$$\mathbf{x}_{k+1} = A\mathbf{x}_k + B\mathbf{u}_k + K\mathbf{e}_k$$
$$\mathbf{y}_k = C\mathbf{x}_k + D\mathbf{u}_k + \mathbf{e}_k$$

The Goal of PBSID:
The primary objective of the PBSID algorithm is to estimate the system matrices (A,B,C,D) and the Kalman gain K directly from the measured input $\mathbf{u}_k$ and output $\mathbf{y}_k$ data.

Task 2.3 : Theoretical Foundations: Main Steps of PBSID

1. Regression on Past Data and System Order Estimation

The algorithm first relates the current output, y_k, to a window of past data $(u_{k-1},y_{k-1},...,u_{k-p},y_{k-p})$.
This is done by solving a least-squares problem to estimate a set of intermediate coefficients, known as Markov parameters.
The estimated Markov parameters are arranged into a special block-Hankel matrix. An SVD is then performed on this matrix.

2. State Sequence and Matrix Estimation

The SVD results (S_{SVD},V_{SVD}) are used to compute an estimate of the system's internal state sequence $X_{est,n}$.
Now that the states are known, a second least-squares problem is solved to find the remaining system matrices (A, B, C) and the Kalman Gain (K).

2. Structuring and Refinement

The output of PBSID is a purely numerical black-box model. This final phase gives it physical meaning, by a Black-Box simulation to generate a clean output; then *greyest* function is used to fit the physical parameters of a predefined grey-box structure to the frequency-domain data of the black-box model's output

```
% Vector Z generation
for k = p:N-1
    % Generate Z^(k-1, k-p) using u and y
    Z_kp_k = Z_k2_k1(u, y, k-1, k-p);  % Get p elements
    % Fill the regression matrix Phi
    Phi(k - p+1, 1:2*p) = Z_kp_k(:)';  % Flatten Z into a row
    Phi(k - p+1, end) = u(k);  % Last column is u(k)
end
% LSQR solution
x = lsqr(Phi,Y);
% Separation of results from LSQR
C_Delta_p = x(1:(length(x)-1));
D = x(end);
```

```
% Assumption on past and future window (f == p) and Hankel's matrix construction
Gamma_Delta_pp = zeros(p,2*p);
C_Delta_p = [C_Delta_p', zeros(1,2*p)];
for i = 1:p
    Gamma_Delta_pp(i,:) = C_Delta_p(2*i-1: 2*i + 2*p -2);
end

% Full Z-array constuction for SVD
Z = [];
for k = 0 : N-p-1
    Z_temp = Z_k2_k1(u, y, p+k-1, k);
    Z = [Z, Z_temp];
end
% Singular Value Decomposition
[~, S_svd, V_svd] = svd(Gamma_Delta_pp * Z , 'econ');
```

```
% State estimation creation and input vector without past widow
X_est_n = sqrtm(S_svd(1:n, 1:n)) * V_svd(:, 1:n)';
```

```
% C matrix estimation by LSQR
b_C = Y - D*u_C; % Right-hand side
A_C = X_est_n'; % Left-hand side
C = lsqr(A_C,b_C); % LSQR solution
```

```
 A,B and K matricies estimation
BK = lsqr(A_ABK, b_ABK);
~ Solution reshaping
A = reshape(ABK(1:n^2),n,n);
B = ABK(n^2+1:n^2+n);
K = ABK(n^2+n+1:end);
```

```
% Parameters definition
Xu = params(1);
Xq = params(2);
Mu = params(3);
Mq = params(4);
Xd = params(5);
Md = params(6);

% Model definition
A=[Xu, Xq, -9.81; Mu, Mq, 0; 0, 1, 0];
B=[Xd; Md; 0];
C=[0, 1, 0];
D=0;

end
```

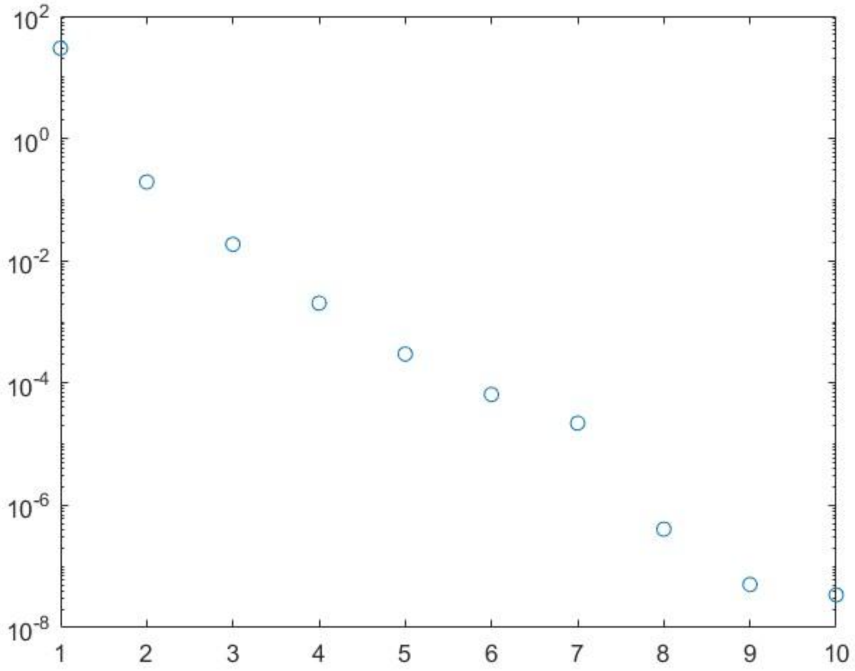
```
% Structuring with greyest
y_id = lsim(ss(A_PBSID,B_PBSID,C_PBSID,D_PBSID), u_3ord, t_3ord);
theta0 = [Xu Xq Mu Mq Xd Md] + 0.01 * (-0.5 + rand(1,6)).*[Xu Xq Mu Mq Xd Md];
model_fun = @drone_model_PBSID;
[identification_PBSID, error_PBSID] = greyest_structuring(u_3ord,y_id,Ts_3ord,theta0,model_fun,real_parameters);
```

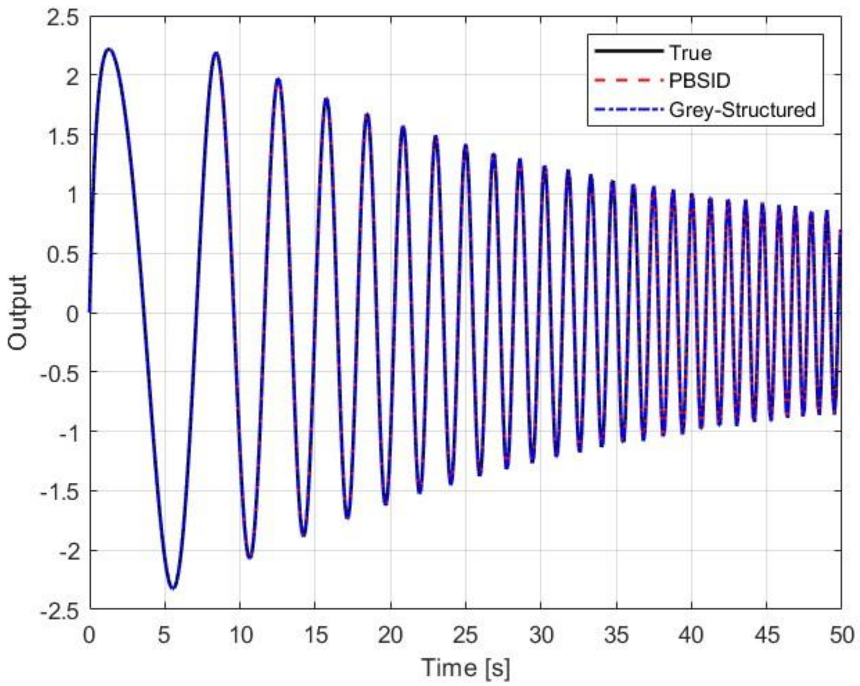
```
% Greyest Options
options = greyestOptions('Display', 'on', 'SearchMethod', 'lsqnonlin');
options.SearchOptions.FunctionTolerance = 1e-6;
% Actual Model Identification
estimated_model = greyest(data_fd, sys_init);
```

Task 2.3 : order 1 experiment validation

$$\dot{x}(t) = A\,x(t) + B\,u(t) = 0.8\,x(t) + 0.5\,u(t),$$
$$y(t) = C\,x(t) + D\,u(t) = 1 \cdot x(t) + 0 \cdot u(t) = x(t)$$

u(t): sweep signal from 0.05 hz to 1 hz





Task 2.4 : Data Acquisition & Preparation (Part 1)

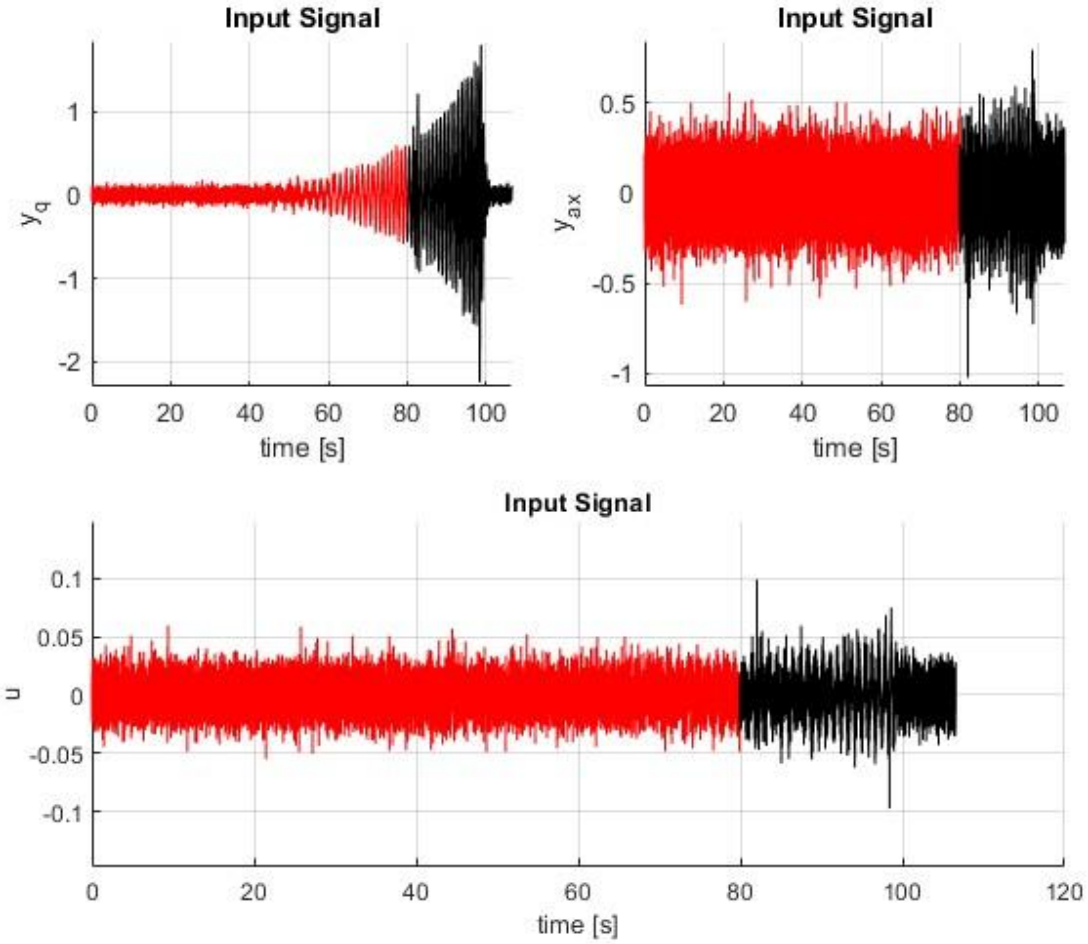
In order to ensure the identified model is robust and can generalize to new data, the dataset was split before any identification or cleaning procedures.

1. The Principle of Cross-Validation

The core idea is to train and develop the model using one portion of the data, and then to test its final performance on a separate portion that it has never seen before.

2. Separation

The dataset from the sweep excitation was divided in Identification Set (75%), shown in red in the side plot, and Validation Set (25%), in black.

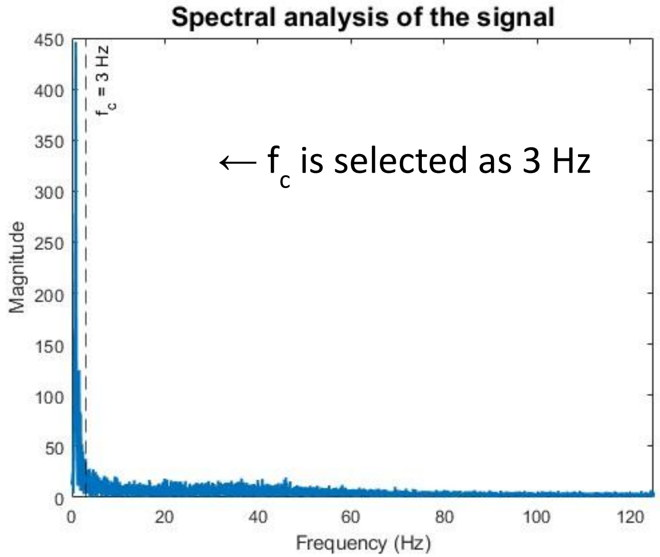


Task 2.4 : Data Acquisition & Preparation (Part 2)

1. Spectral Analysis

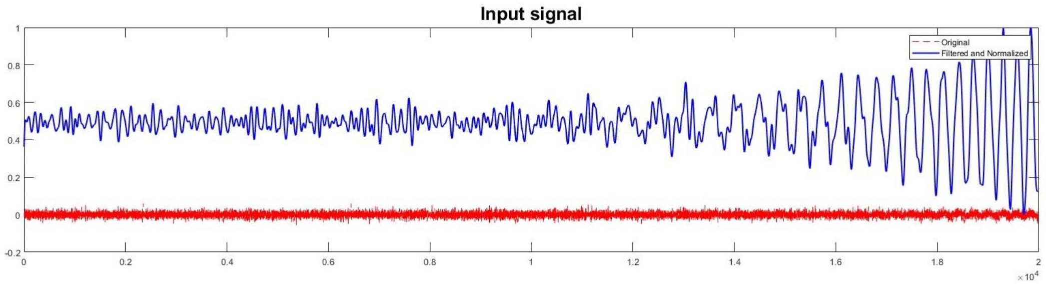
Before filtering, a Fast Fourier Transform (FFT) was applied to the raw pitch rate output (**q**) to analyze its frequency content.
A cutoff frequency (f_c) was chosen based on the point where the signal's magnitude drops and noise begins to dominate.

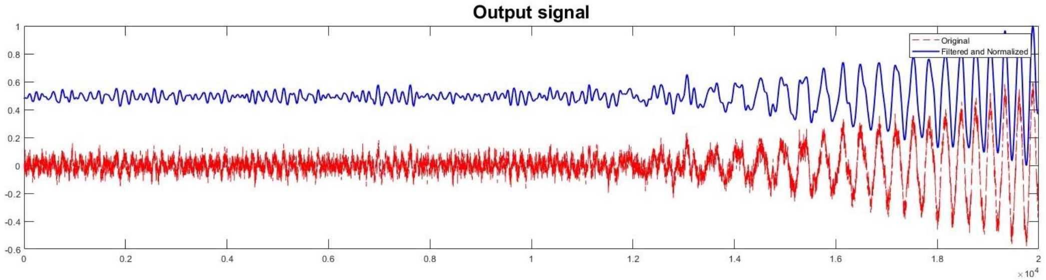
```
fs = 1/Ts; % Sampling frequency
n = length(y);
frequenze = linspace(0, fs/2, floor(n/2));

% Fourier's Transform
Y_fft = abs(fft(y));
Y_fft = Y_fft(1:floor(n/2)); % Positive half
```



2. Filtering and Smoothing

Based on the spectral analysis, a multi-step cleaning process was applied to both the input and output signals.
A 4th-order Low-Pass Filtering was applied using the selected cutoff frequency f_c . This effectively removes the high-frequency noise from the data.
Then a Savitzky-Golay filter was then used to smooth the signal. This filter is excellent for preserving the main features of the signal while removing smaller, unwanted fluctuations.





Objective: To systematically determine the optimal value for the hyperparameter **p**, which defines the "past window size" or memory of the PBSID algorithm.

1. Methodology

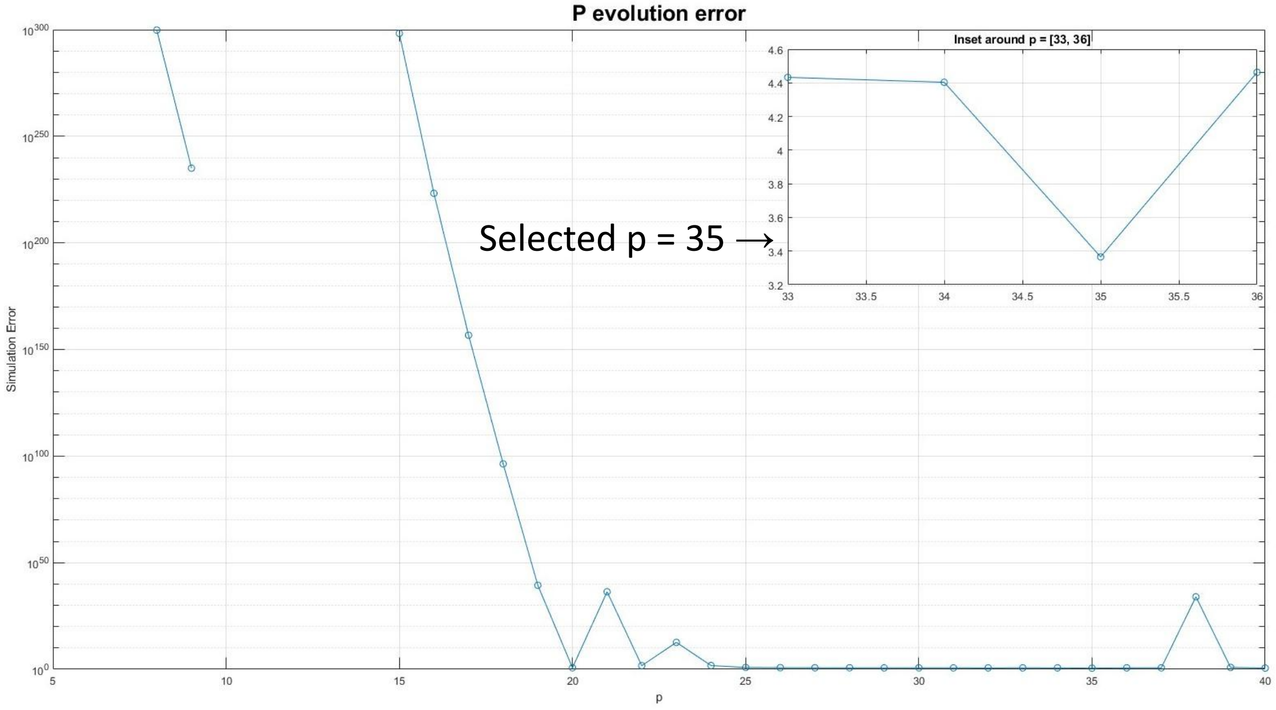
The PBSID implementation uses the assumption that the future window is equal to the past window (f = p). This simplifies the tuning process to finding a single optimal parameter.
A linear search is performed over a predefined range of p values [6,40].
For each p in the range, a complete black-box model is identified. The L2-norm of the simulation error is then calculated to score the model's accuracy as :

```
error(i) = norm(y - lsim(sys_id, u, t)); % Errore di simulazione
```

The final value for p is chosen as the one that minimizes the simulation error, ensuring the most accurate model based on the chosen metric, as could be seen in the side picture.

2. Results

The plot below shows the simulation error for each tested value of p. The minimum of the curve clearly indicates the optimal choice for the past window size, that is found as p = 35 .



Task 2.4 : PBSID Model Identification (Part 1)

1. Matrix Construction

The algorithm first uses the results of the initial regression to form a block-Hankel matrix ($\boldsymbol{\Gamma}_p \boldsymbol{\Delta}_{,p}$), which implicitly contains information about the system's observability.

2. SVD Application

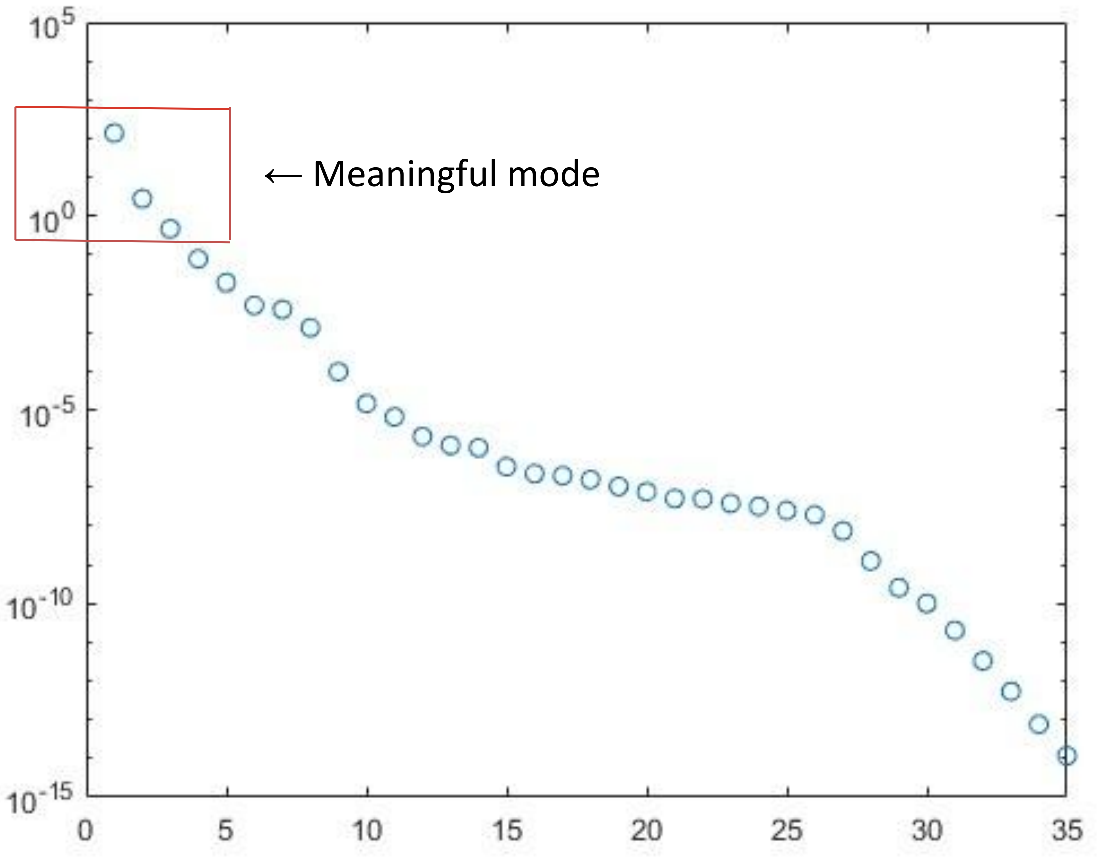
Then the SVD is applied to this matrix multiplied by the past input-output data. This decomposition breaks down the system into its fundamental modes. The resulting singular values (the diagonal of the S_{SVD} matrix) quantify the "energy" or importance of each mode. The singular values corresponding to the system's actual dynamics will be much larger than those related to noise.

```
% Singular Value Decomposition
[~, S_svd, V_svd] = svd(Gamma_Delta_pp * Z , 'econ');
```

3. Results and Selection

The system order n is chosen by visually inspecting the semilog plot of the singular values and looking for a significant "gap" or "elbow". This gap separates the dominant modes of the system from the noise floor.

As the plot clearly shows, there is a sharp drop in magnitude after the third singular value. So based on this evidence, the system order was selected to be n = 3.



Task 2.4 : PBSID Model Identification (Part 2)

With the system order n and the state sequence now estimated, the final phase of the PBSID algorithm is to compute the specific state-space matrices (A,B,C,D) and the Kalman Gain K. This is achieved by solving two distinct least-squares problems.

4. Estimation of the Output Matrices (C and D)

The matrices **C** and **D** are determined from the system's output equation:

$$\mathbf{y_k \sim Cx_k + Du_k}$$

This equation is framed as a linear least-squares problem. The feedthrough matrix D was already obtained during the initial regression.

```
% C matrix estimation by LSQR
b_C = Y - D*u_C; % Right-hand side
A_C = X_est_n'; % Left-hand side
C = lsqr(A_C,b_C); % LSQR solution
```

5. Estimation of the State & Gain Matrices (A, B, K)

Once **C** and **D** are known, the remaining matrices are found using the state update equation:

$$\mathbf{x_{k+1} \sim Ax_k + Bu_k + Ke_k}$$

Firstly, the innovation sequence e_k is calculated, then larger least-squares problem is then constructed to solve for **A**, **B**, and **K** simultaneously.

```
% A,B and K matricies estimation
ABK = lsqr(A_ABK, b_ABK);
% Solution reshaping
A = reshape(ABK(1:n^2),n,n);
B = ABK(n^2+1:n^2+n);
K = ABK(n^2+n+1:end);
```

The result of this entire process is a complete, numerically-defined black-box model. This model accurately describes the system's input-output behavior but its matrix elements do not have direct physical meaning.

Task 2.4 : Structuring Approach

The black-box model accurately captures the input-output dynamics, but its matrix elements are just numbers. Structuring is the process of finding the underlying physical parameters that produce this behavior. This provides valuable physical insight into the drone's dynamics.

1. Generate Target Data

The identified black-box model is used to simulate a "clean" output response (y_{id}). This response is then converted into the frequency domain. This frequency response serves as the target for the optimization.

2. Define Grey-Box Structure

A model structure is defined in a separate function with a single output (pitch rate **q**). This model is explicitly written in terms of the physical parameters that we want to estimate.

3. Perform Optimization

MATLAB's *greyest* function is the core of this step. It is an optimization algorithm that iteratively adjusts the physical parameters of the grey-box model until its frequency response matches the target frequency response of the black-box model.

Parameter	θ_{true}	θ_{PBSID}	θ_{true} - θ_{PBSID}
Xu	-0.1068	-0.1070	0.1596 %
Xq	0.1192	0.1192	0.0186 %
Mu	-5.9755	-6.0038	0.4730 %
Mq	-2.6478	-2.6517	0.1490 %
Xd	-10.1647	-10.1952	0.3003 %
Md	450.71	450.502	-0.0462 %

Task 2.4 : Validation via Cross-Validation

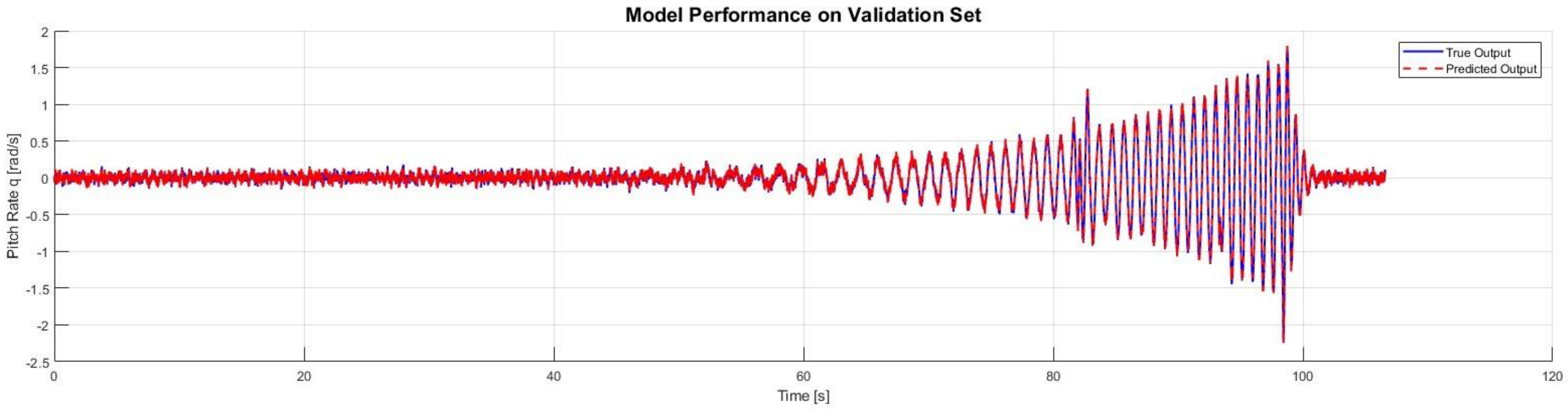
In order to perform the ultimate test of the model's generalization capability by evaluating its performance on the 25% validation set reserved at the beginning.

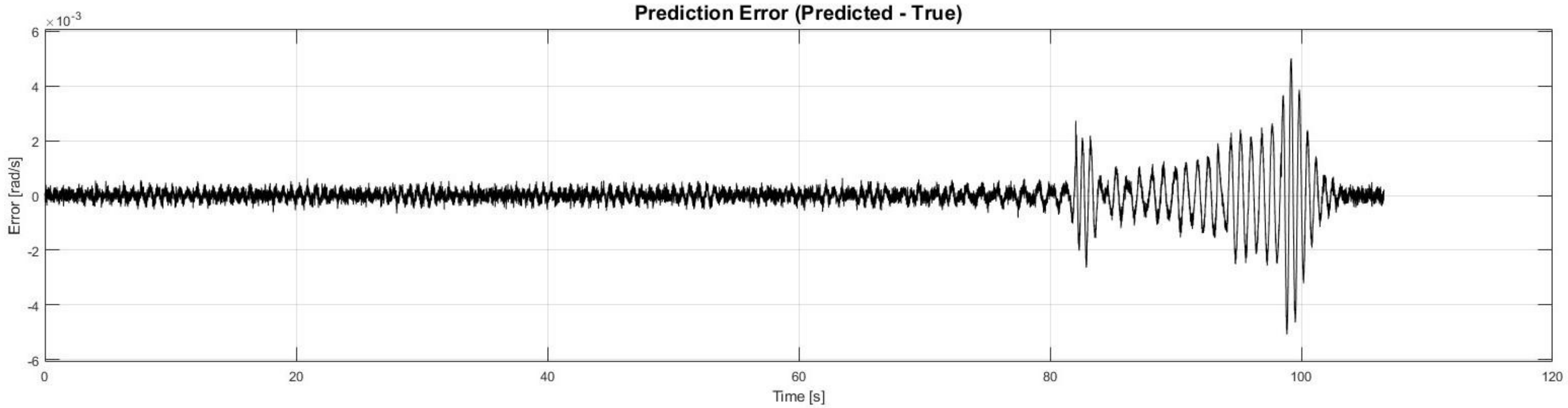
1. Methodology

The validation input signal was fed into the final, structured state-space model. The model's simulated output was then compared against the true measured output from the validation set.

2. Results

The plots show the comparison between the output and the residual error.
The top plot shows an excellent match between the model's prediction and the true system output and the bottom plot shows that the prediction error remains small throughout the entire validation sequence. The maximum absolute error is 0.0051 rad/s

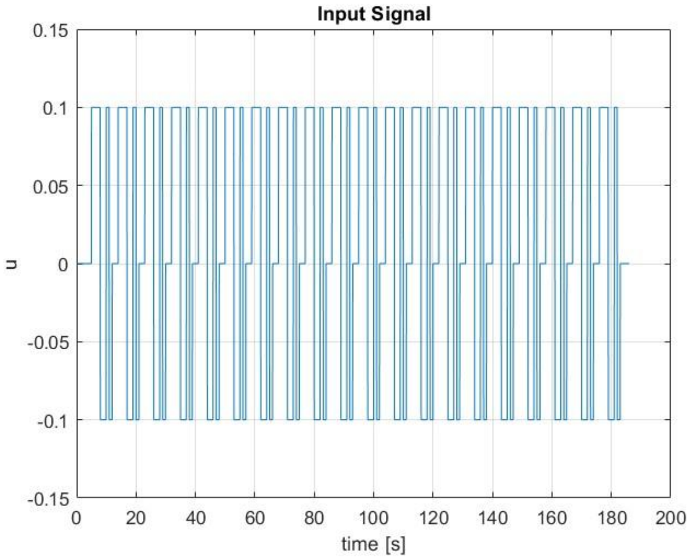


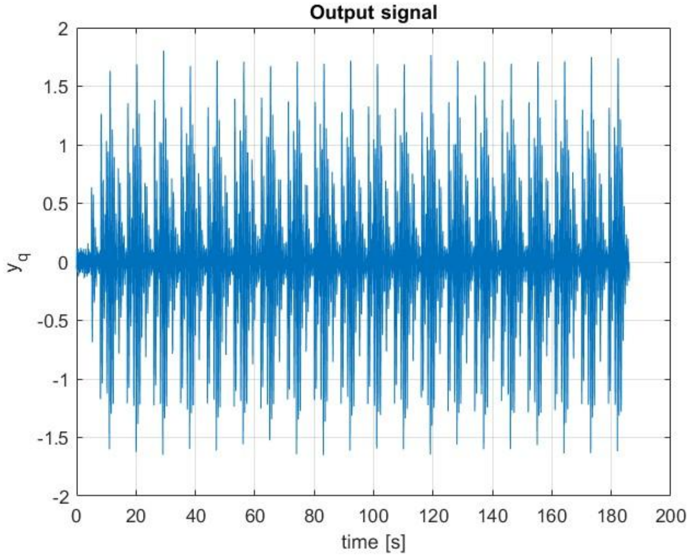


Task 2.4 : Validation via 3-2-1-1 sequence (Part 1)

To test the generalization capability of the identified model, a standard aerospace validation signal (3-2-1-1 sequence) was used. This input is fundamentally different from the sweep signal used for the identification process.
This input is composed of a series of repeating, alternating pulses where the pulse widths follow a ratio of 3-2-1-1.
An important insight was gained during this process, because the cutting frequency (~ 5 Hz) and the past window size (15) will change depending on the input signal given to the system.
With this signal the parameters of the system found are described in this side table :

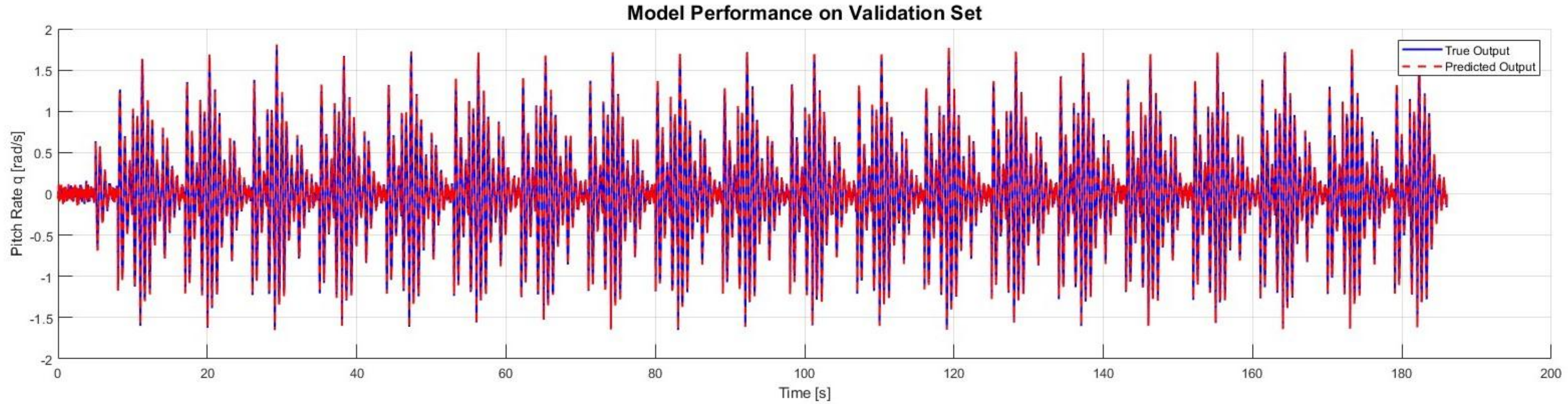
Parameter	θ_{true}	θ_{PBSID}	θ_{true}- θ_{PBSID}
Xu	-0.1068	-0.1068	0.0225 %
Xq	0.1192	0.1198	0.4937 %
Mu	-5.9755	-5.9587	-0.2813 %
Mq	-2.6478	-2.6374	-0.3942 %
Xd	-10.1647	-10.125	-0.3903 %
Md	450.71	448.743	-0.4364 %

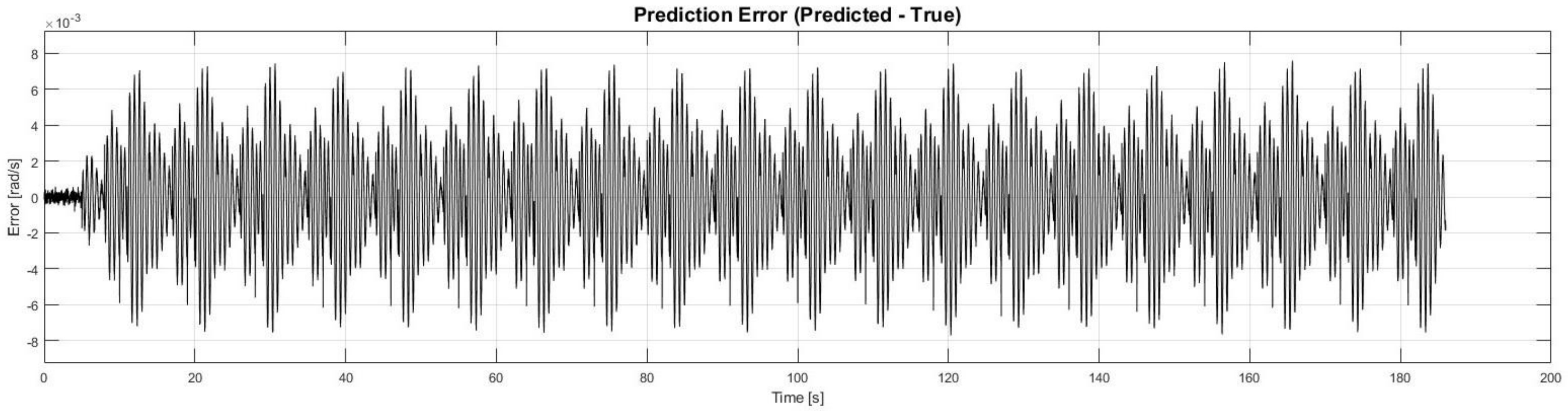




Task 2.4 : Validation via 3-2-1-1 sequence (Part 2)

To further confirm the robustness of the identification methodology itself, a separate cross-validation experiment was conducted using *only* the 3-2-1-1 dataset. The same trusted workflow was applied to the new dataset, using the splitting 75/25.



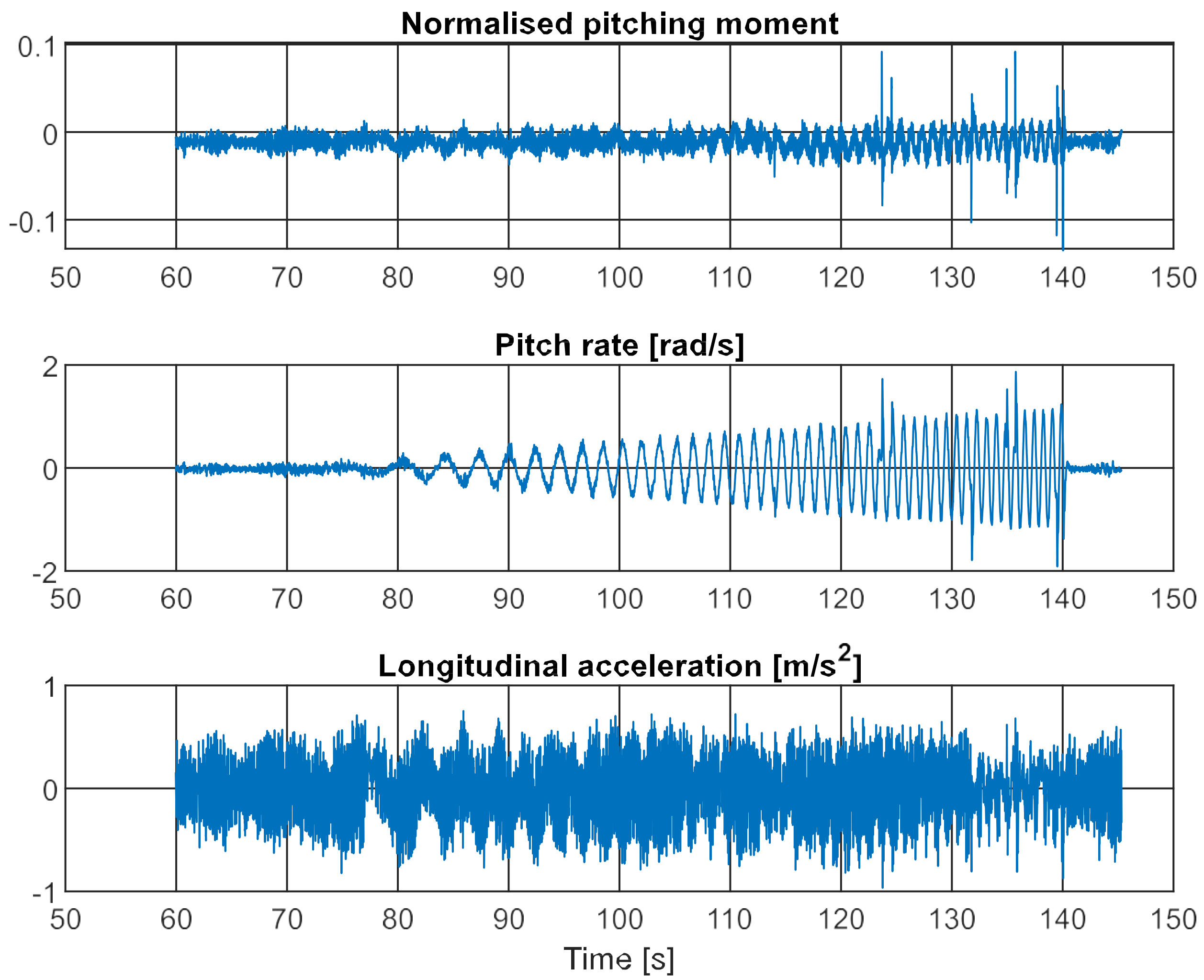


The maximum error evaluated with the cross validation was 0.0077 rad/s

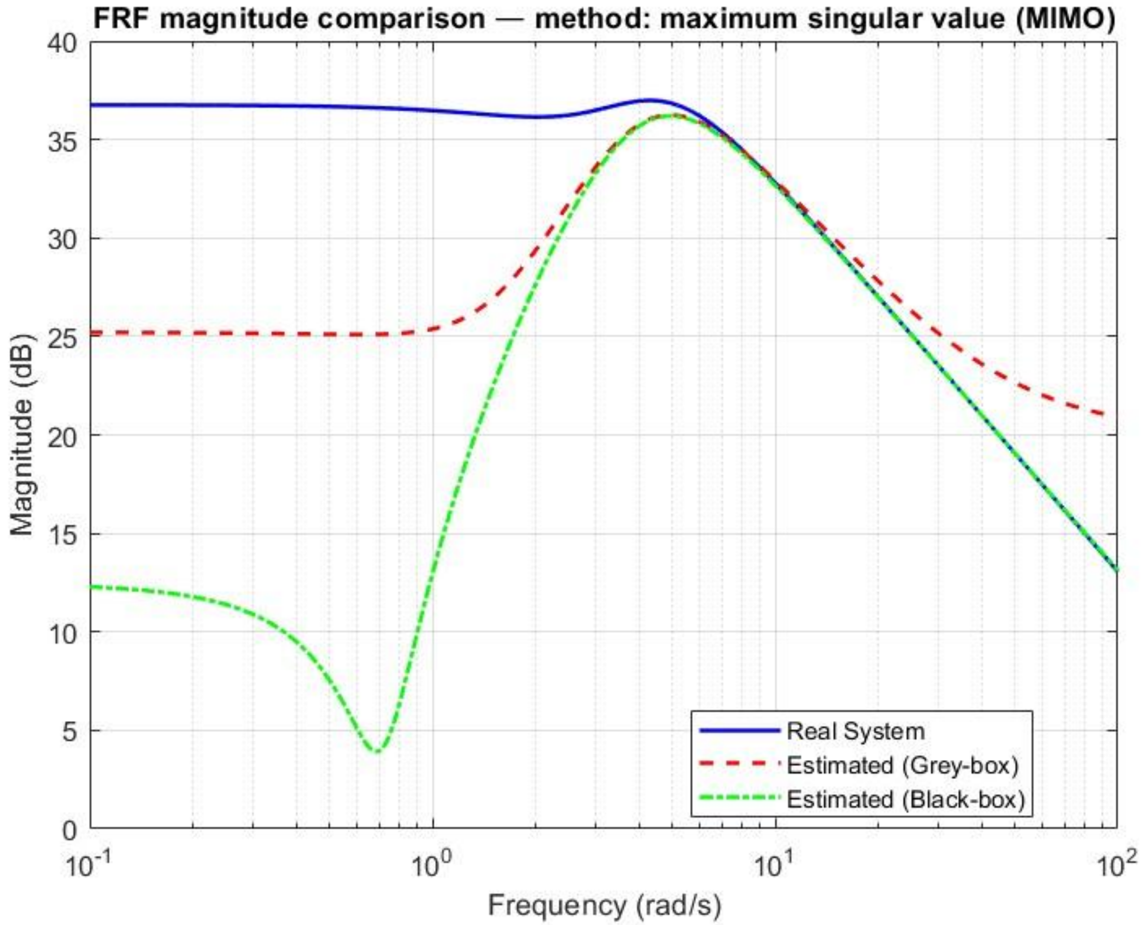
Longitudinal dynamics: excitation signal

- Experiments are carried out under position and attitude feedback, in order to guarantee safe operation.

- Baseline excitation input is a long duration sweep: excite low-frequency translational dynamics



Estimated parameter values



Parameter	θ_{true}	θ_{PBSID}	θ_{grey}
Xu	-0.1068	-0.1068	-0.0955
Xq	0.1192	0.1198	0.1225
Mu	-5.9755	-5.9587	-5.9032
Mq	-2.6478	-2.6374	-2.6815
Xd	-10.1647	-10.125	-10.4862
Md	450.71	448.743	450.7453

Task 3.1 : Model Comparison

Estimated Zeros and Poles

Noise Realization Setup

Noise Seed Randomization

Randomized noise seed for x, v_x , θ, q before system simulation

Model simulation at each iteration

Simulations with ‘fast Restart’ option allowed for execution time saving

identified Parameters pool for Statistical Analysis

For each method the following metrics are determined: parameter’s mean values and standard deviations, associated FRF and its Zeros and Poles

Choice of N

Preliminary estimate of necessary runs

Necessity to obtain N pairs of estimated parameters

↓

Minimum N that guarantees convergence for both methods

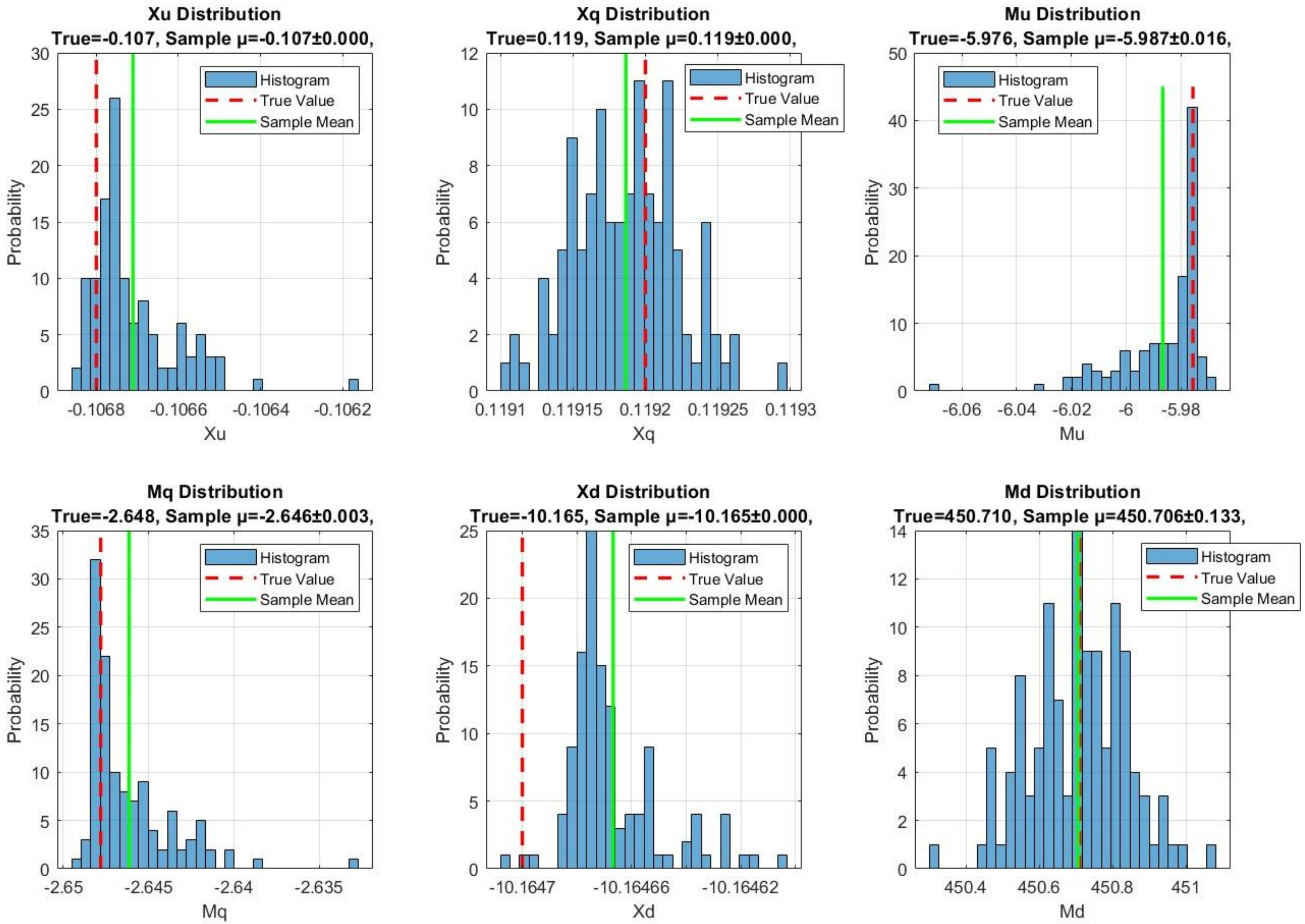
+

Iterative convergence check

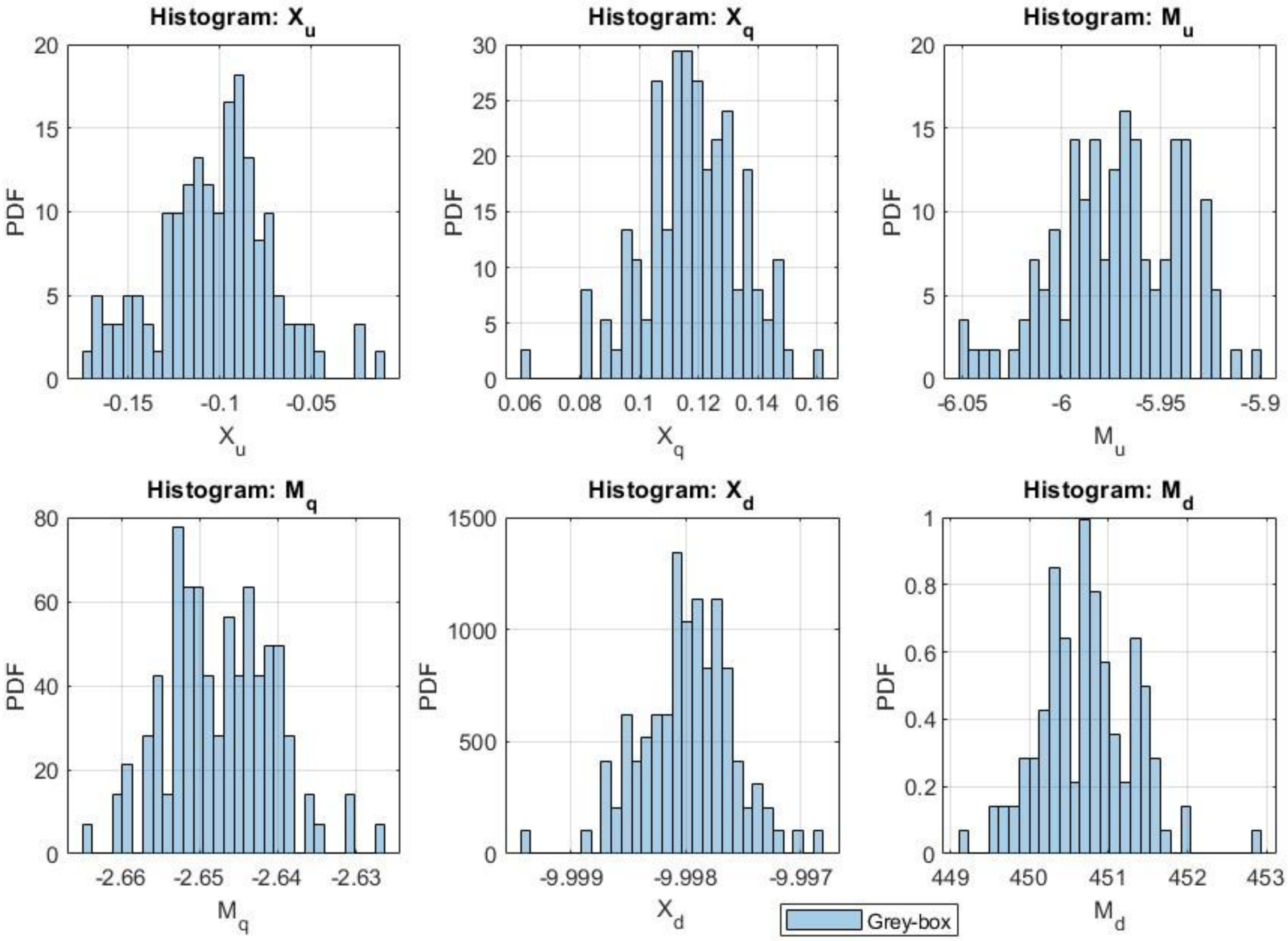
Iterative convergence check that stops the run when the estimated parameters differ less than a threshold ε = 10^{-4} for the last 20 consecutive iterations

Final Choice for N = 120

Task 3.3: Monte Carlo Results, Grey Box Method



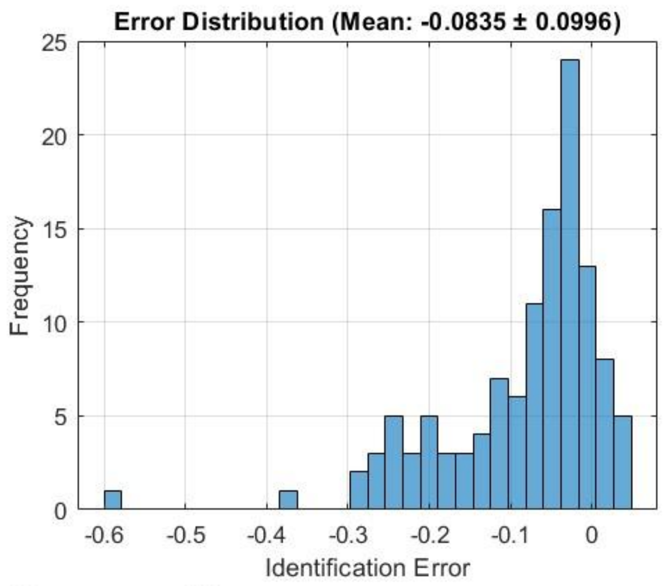
Task 3.3: Monte Carlo Results, Grey Box Method

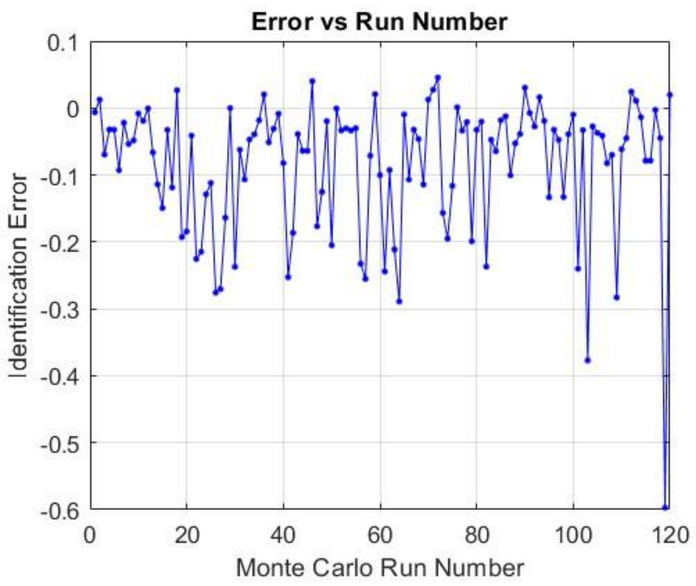


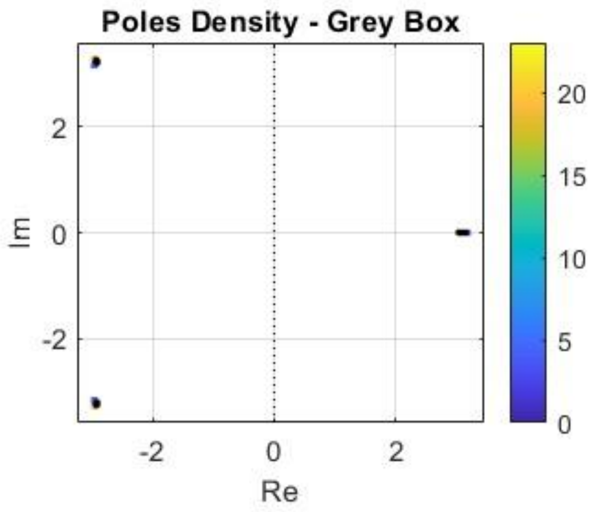
Parameter	Mean value ± σ
X_u	-0.0983 ± 0.0121
X_q	0.1189 ± 0.0043
M_u	-5.9730 ± 0.1245
M_q	-2.652 ± 0.0142
X_d	-9.9981 ± 0.0004
M_d	450.53 ± 1.4620

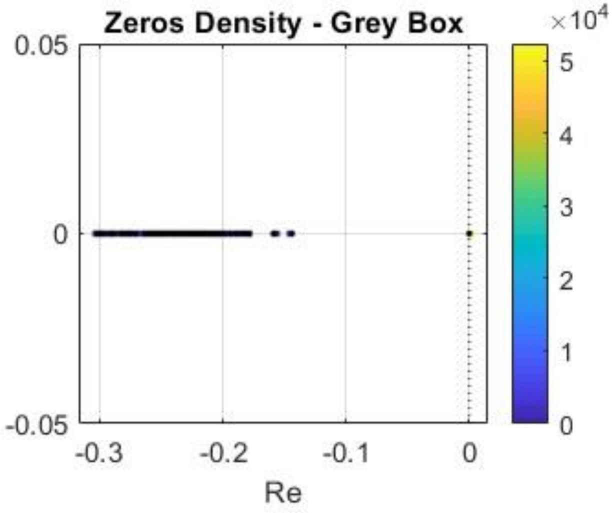
Task 3.3: Monte Carlo Results: Grey Box Method

Error distribution and independence from run number are guaranteed form random noise seed choice

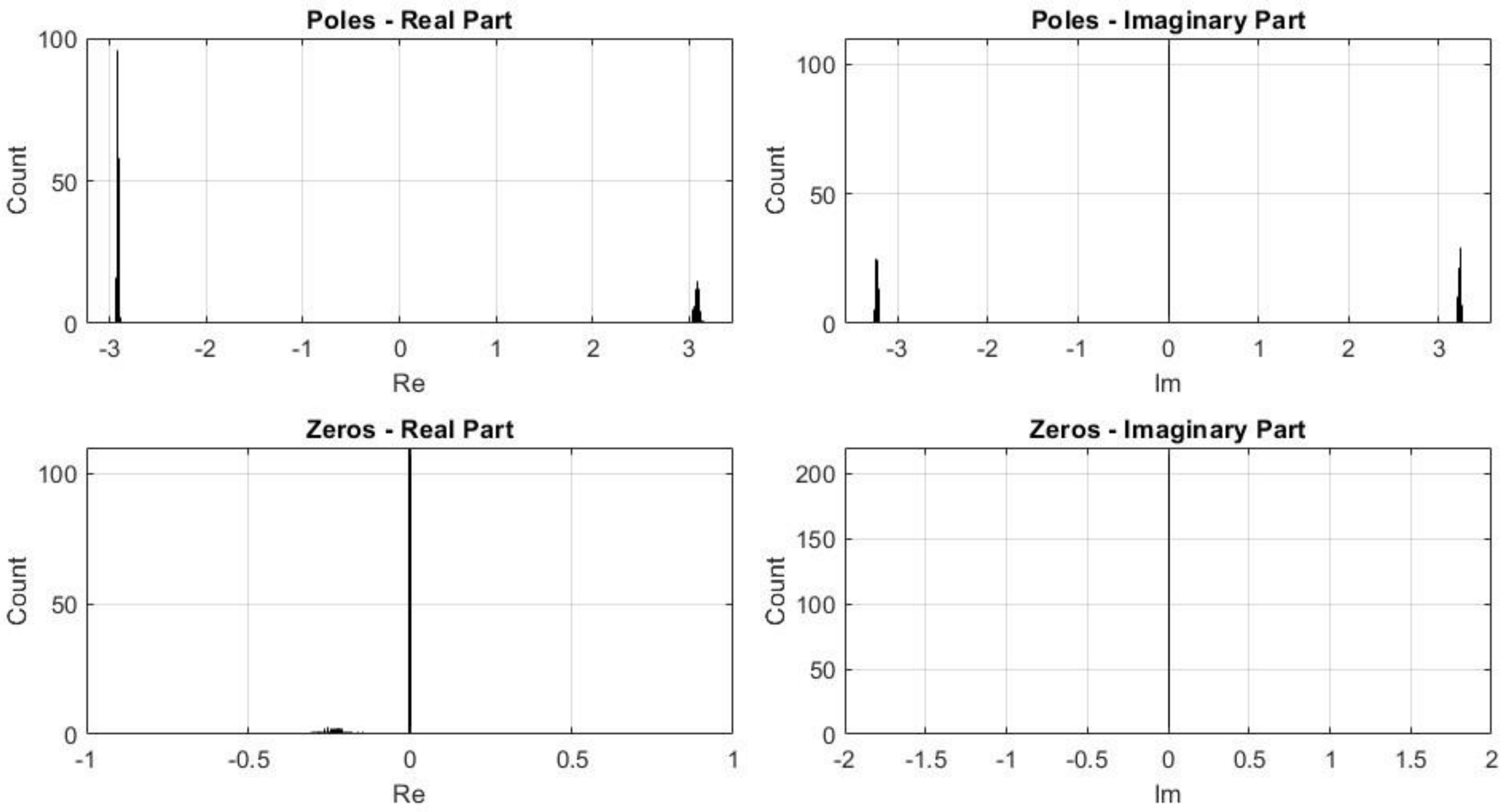




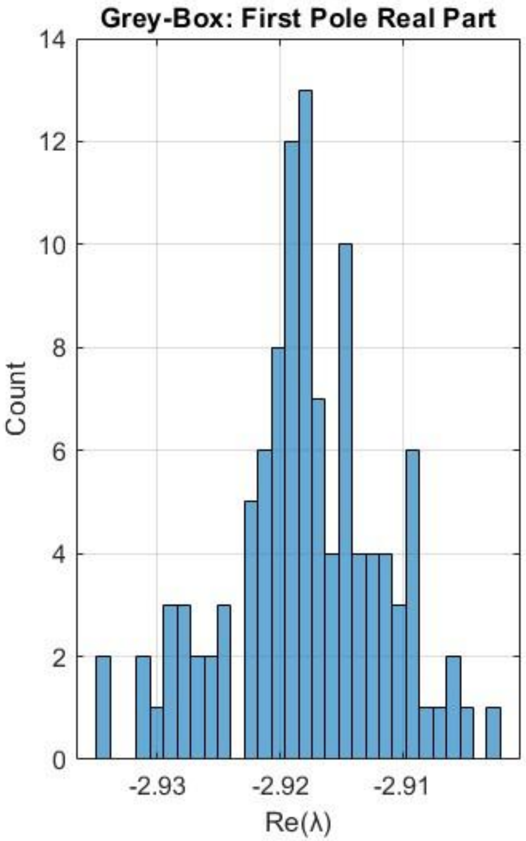


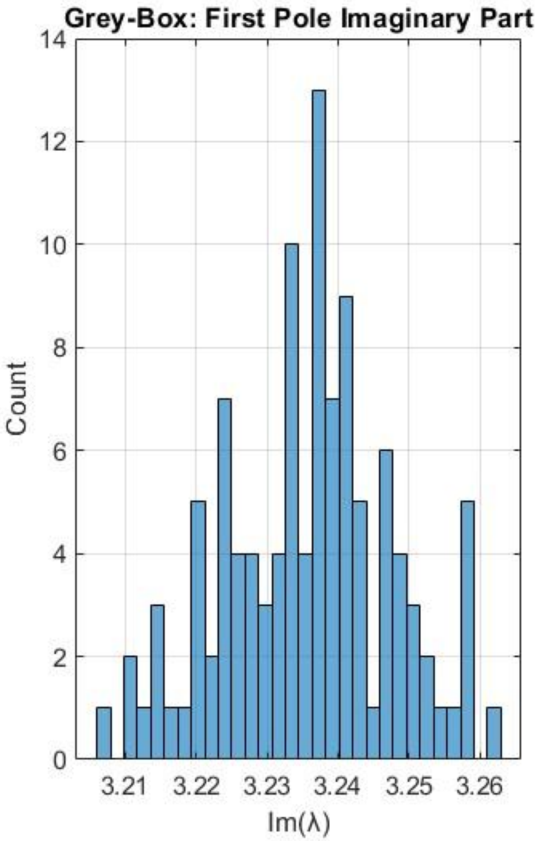


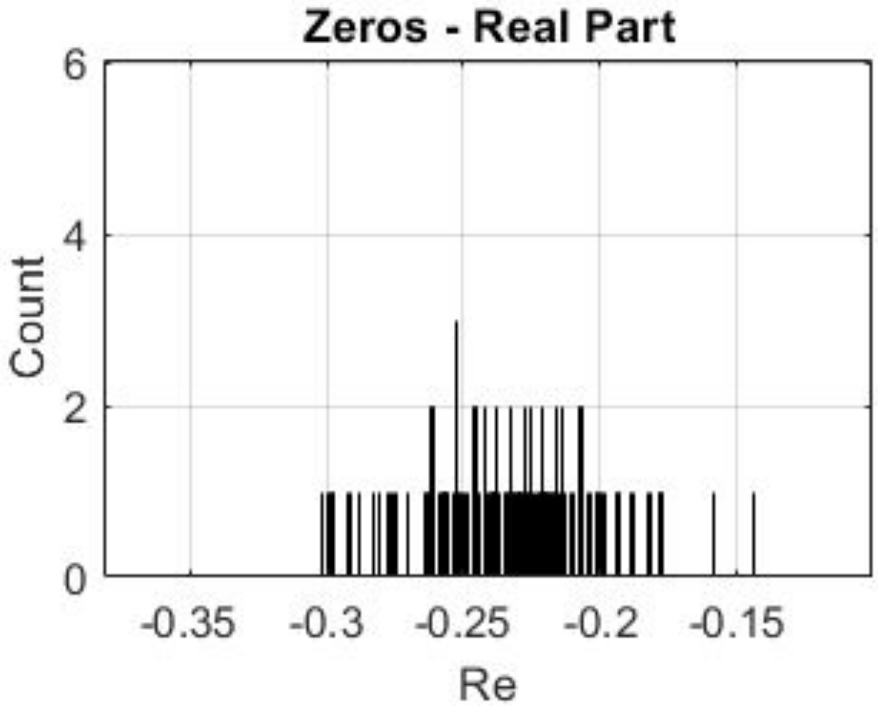
Task 3.3: Monte Carlo Results: Grey Box Method



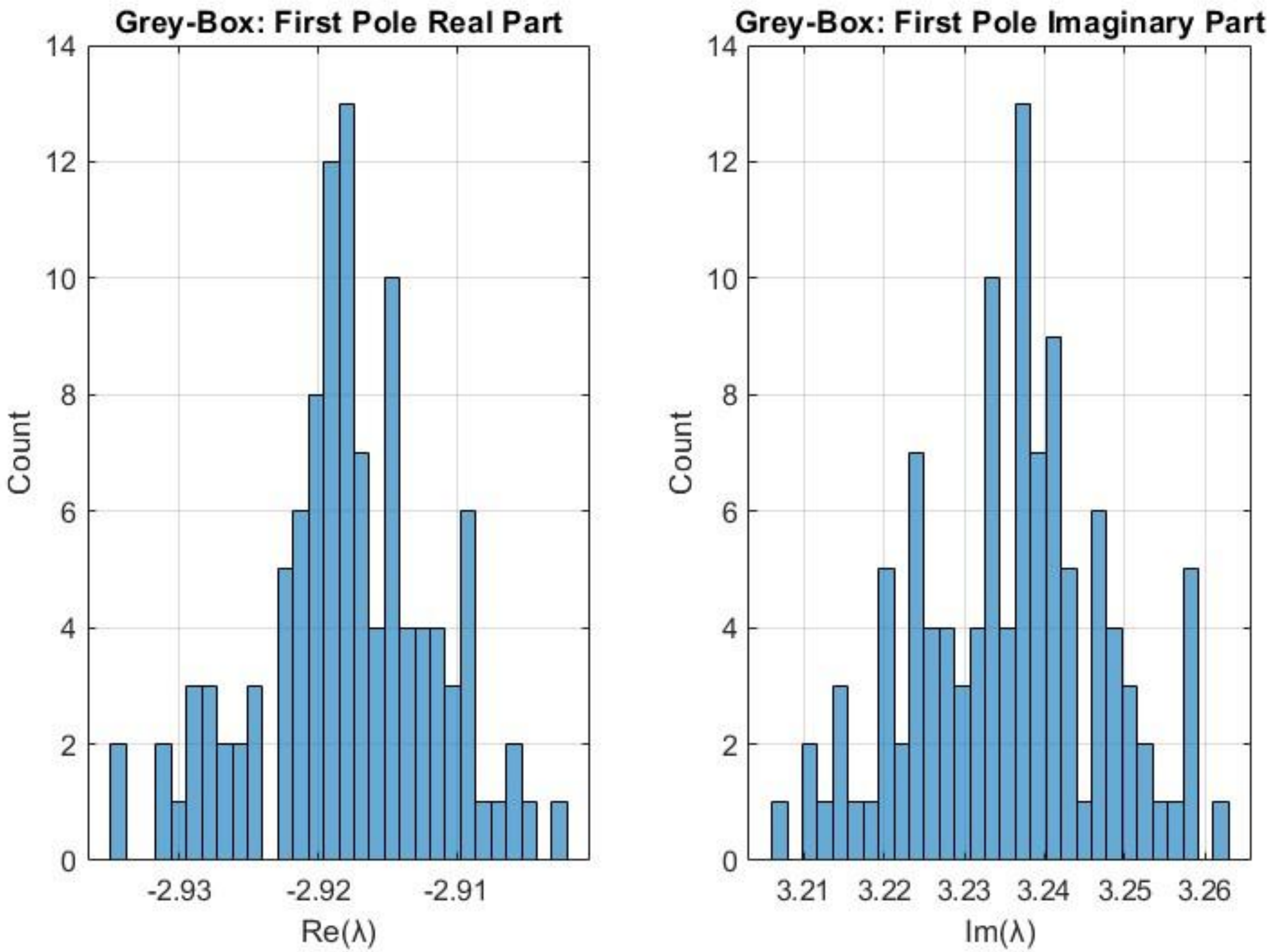
Task 3.3: Monte Carlo Results: Grey Box Method



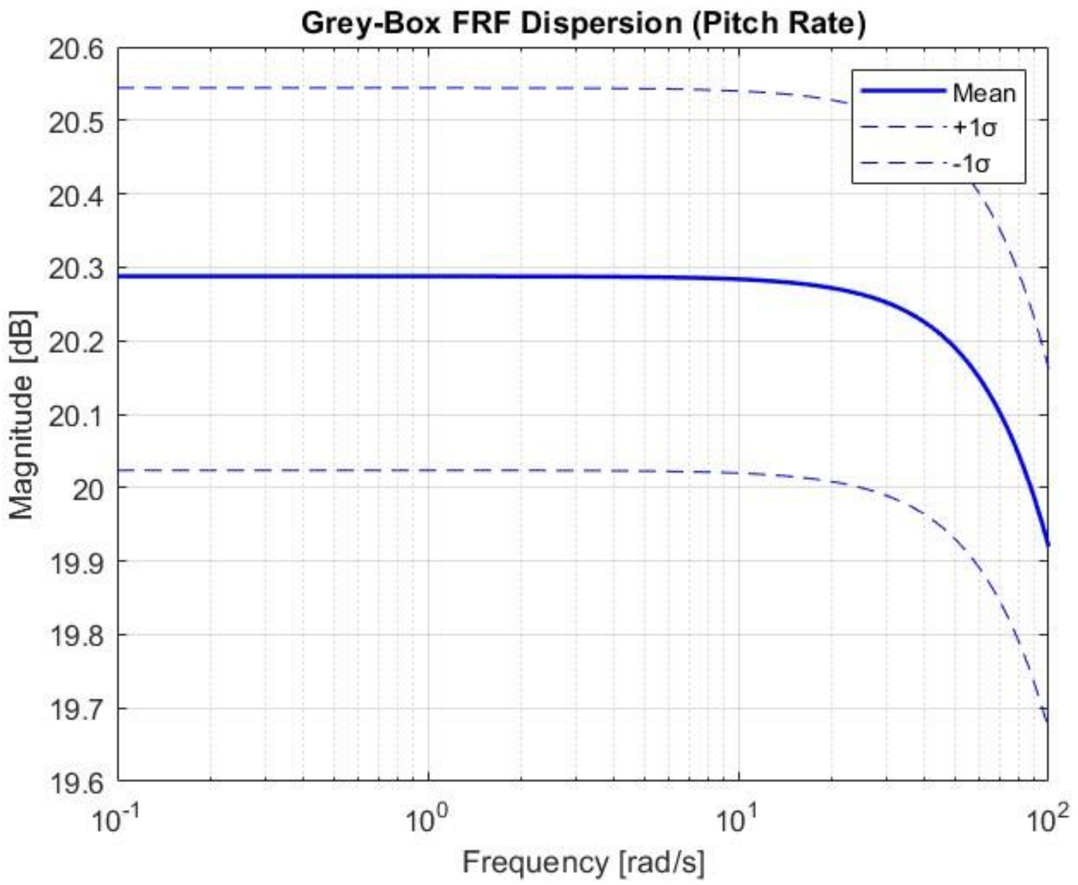


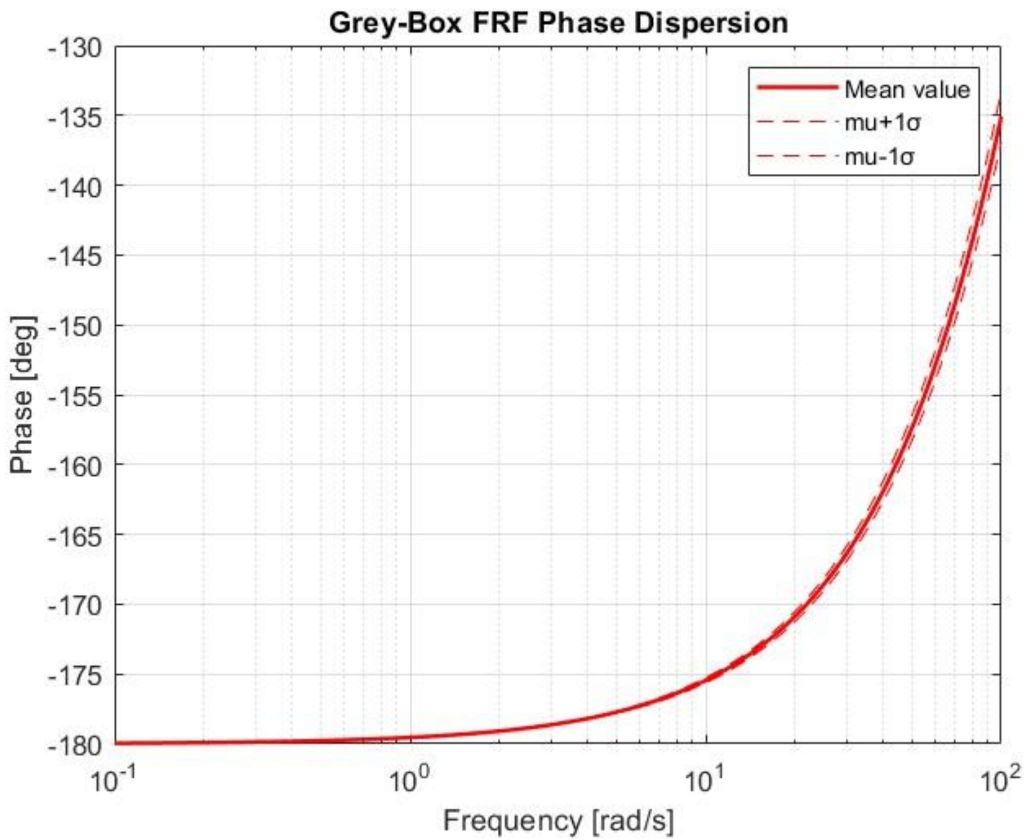


Task 3.3: Monte Carlo Results: Grey Box Method



Task 3.3: Monte Carlo Results: Grey Box Method

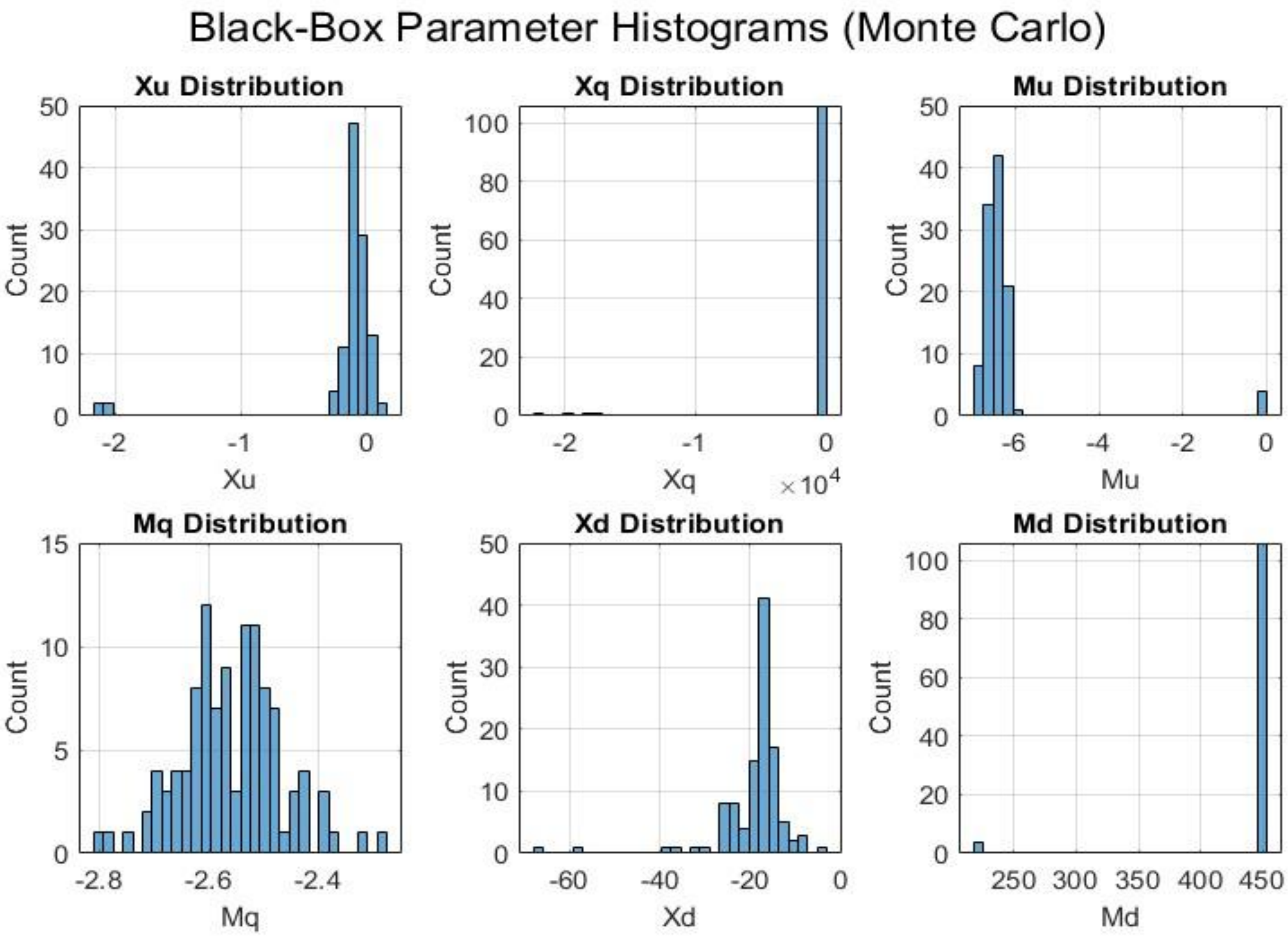




Task 3.3: Monte Carlo Results

Black Box method

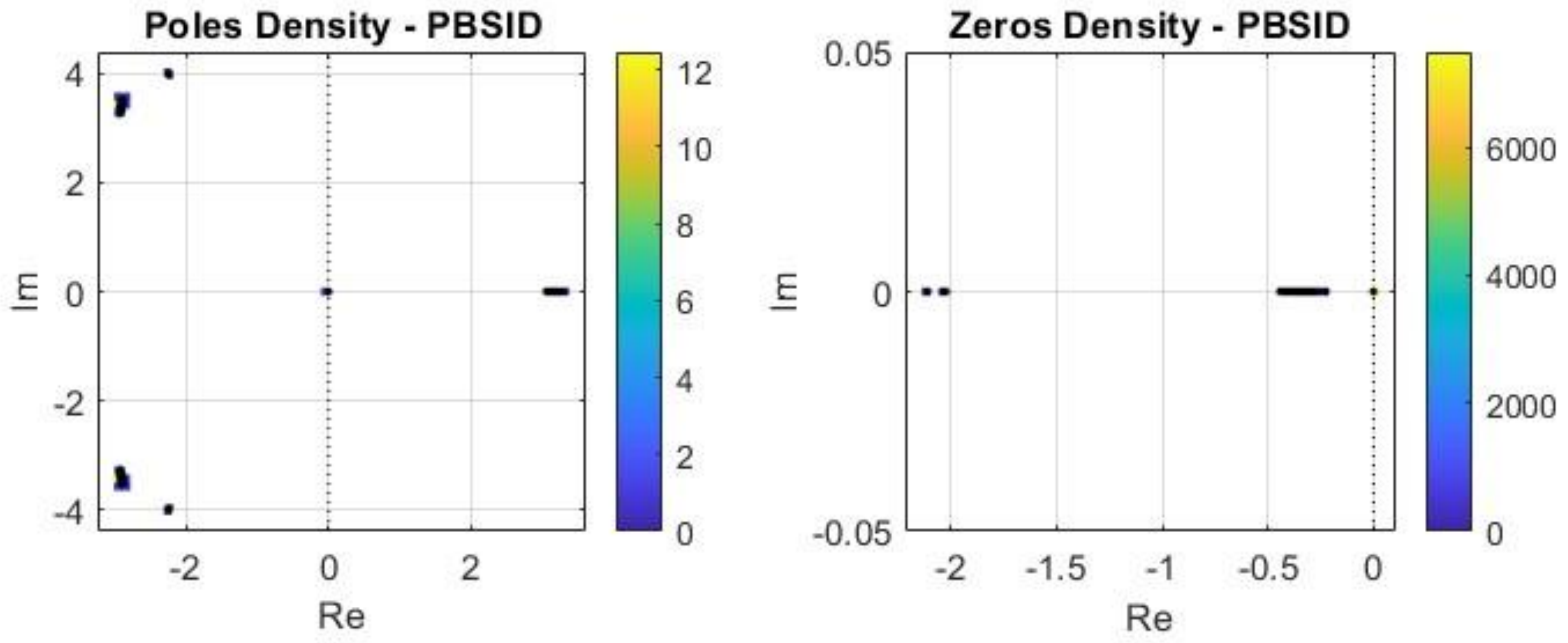
Task 3.3: Monte Carlo Results, Black Box Method



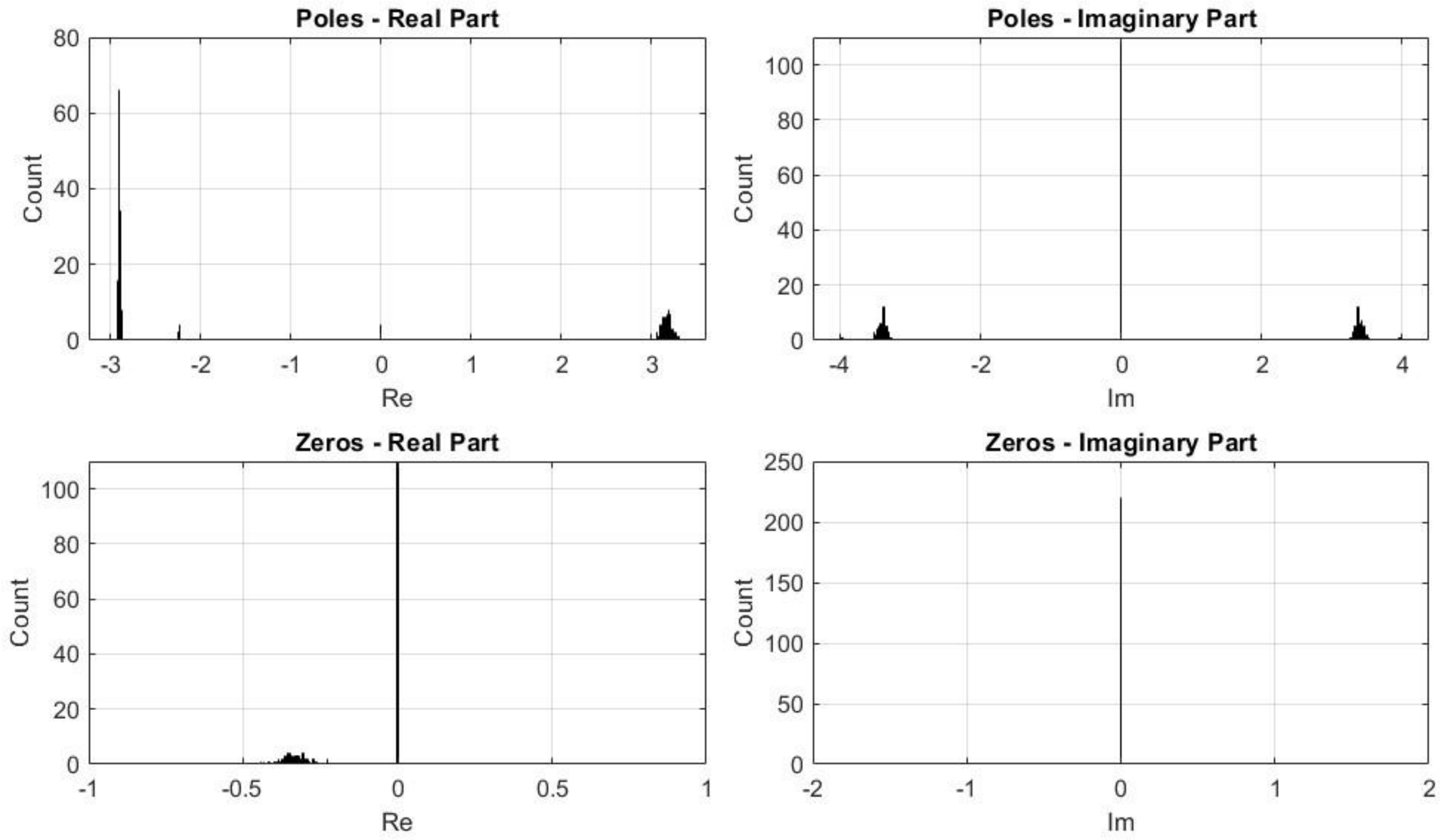
Parameter	Mean value ± σ
X_u	
X_q	
M_u	
M_q	
X_d	
M_d	

Task 3.3: Monte Carlo Results: Black Box Method

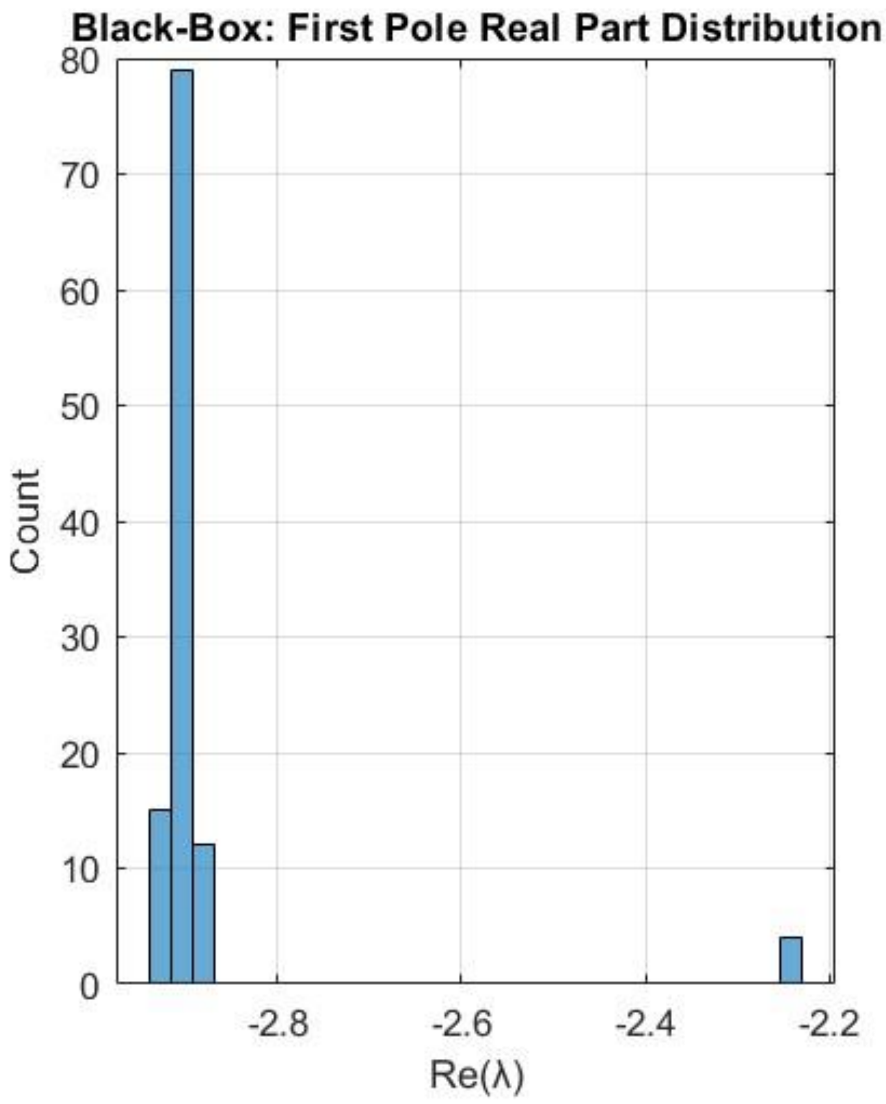
The presence of outliers cases is to be expected in the Black Box case, due to the absence of physical significance of the state space variables of the PBSID method

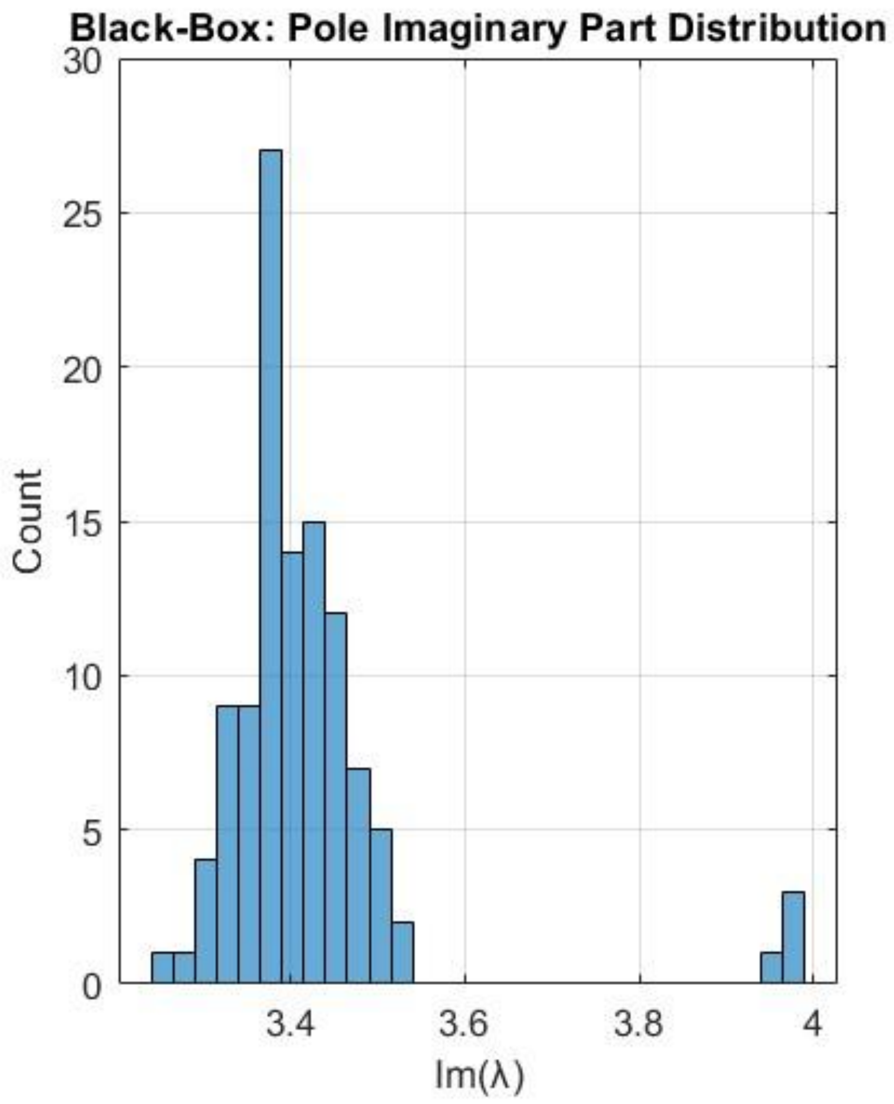


Task 3.3: Monte Carlo Results: Black Box Method



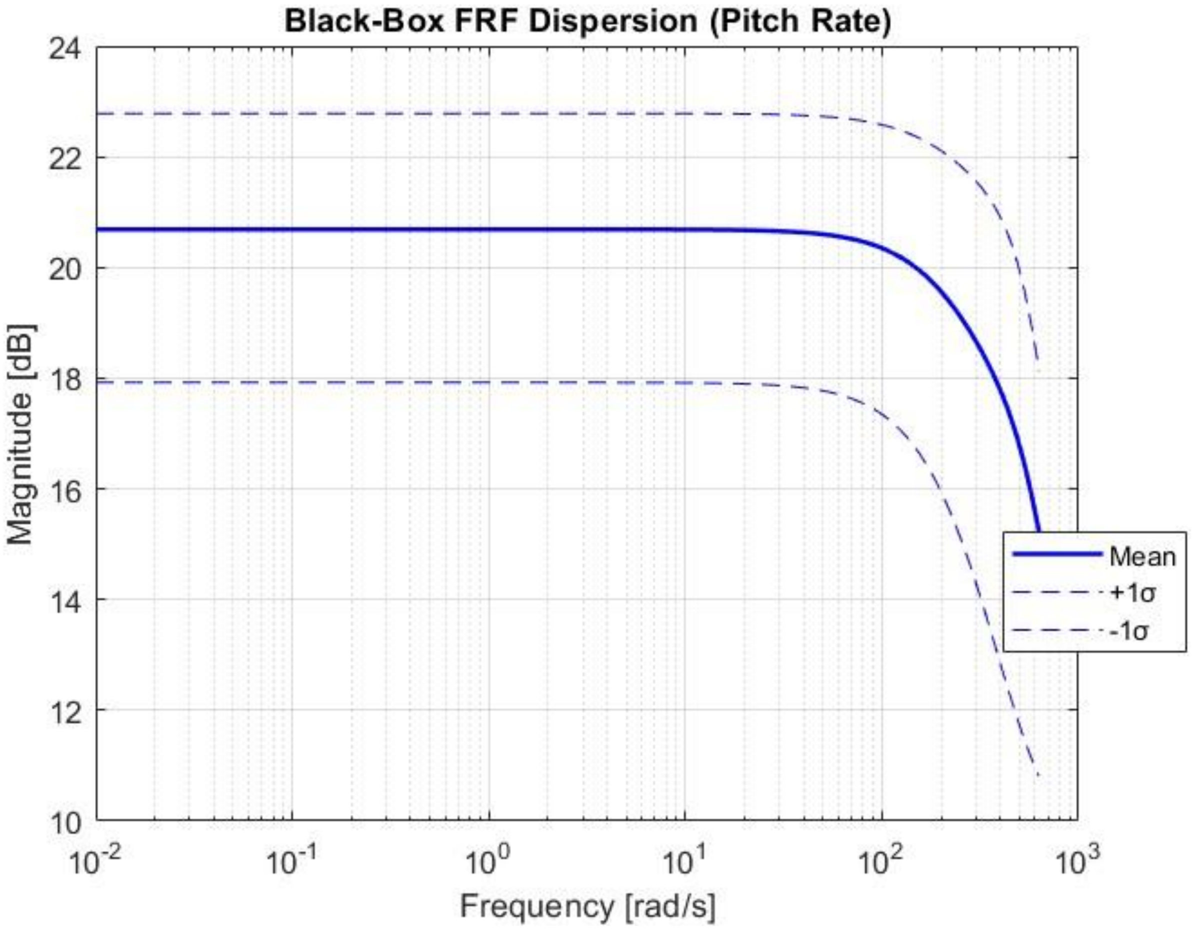
Task 3.3: Monte Carlo Results: Black Box Method

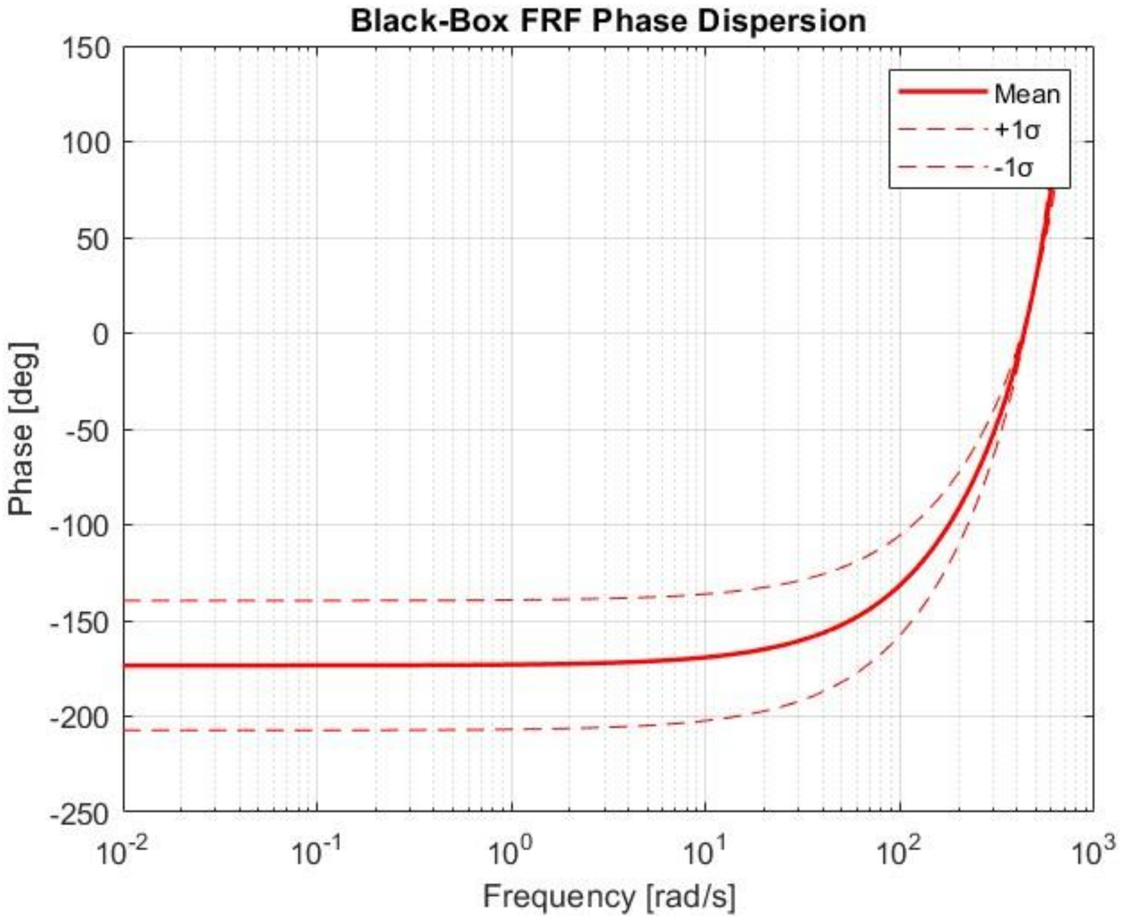




Task 3.3: Monte Carlo results

Dispersion Plots





Task 3.3: Monte Carlo results

Considerations