NTNU
Kunnskap for en bedre verden

Department of Engineering Cybernetics

Specialization project

A take home portable MagLev lab for learning electronics and control

Sacit Ali Senkaya

Supervised by

Damiano Varagnolo

December 18, 2023

Abstract

This study delves into the methodologies and techniques employed in the field of system identification, with a primary emphasis on linear models. As a fundamental process in engineering and scientific teaching, system identification plays a crucial role in unveiling the hidden dynamics of a system based on observed input-output data. In this study, we analyze real-world data to comprehensively understand the complexities of system behaviors.

By utilizing black-box estimation methods, this research aims to reveal the complex relationships between input and output variables. The outcomes of this study not only shed light on the distinction of linear system modeling but also contribute valuable insights into the broader spectrum of system identification. This deeper understanding of model structures not only enhances our theoretical knowledge but also holds practical implications for applications ranging from control systems to predictive modeling in various scientific and engineering domains. As we navigate the complexities of system identification, this research serves as a foundation for improving our understanding and manipulation of complex systems.

Preface

This specialization project serves as a foundational exploration paving the way for the forthcoming master's thesis. It offers a comprehensive overview of the underlying theory, methodologies employed, and the results obtained, setting the stage for a more in-depth investigation in the master's thesis.

I extend my heartfelt appreciation to Professor Damiano Varagnolo and Research Fellow Hans Alvar Engmark for their invaluable contributions and unwavering support throughout this project. Their insightful feedback and encouragement have significantly enriched the quality of this work. I am truly grateful for the opportunity to collaborate with such esteemed mentors.

Table of Contents

List of Figures

1 Introduction

The field of magnetic levitation is evergrowing and extends into many different domains and applications. Among the most notable examples being high-speed trains and even elevators. Magnetism is not a new concept, and was already known by ancient peoples through the lodestone, which is a magnetized piece of iron ore. It wasn't before the 1800s the link between electric current and magnetism was discovered, and it laid the foundations for the electromagnet.

The magnetic levitation system (from now on called maglev) is developed with a clear goal of being an interactive educational tool. The system can be used to learn basic electronics and cybernetics concepts, and can double as a lab that can be successfully carried out at home.

This paper places a particular emphasis on the system identification part, by aiming to construct model structures through the analysis of gathered data. The primary objective is to enhance our understanding of the underlying system dynamics. Furthermore, this study seeks to combine the knowledge with feedback linearization techniques and Kalman Filter design, with the ultimate aim of achieving system stability.

2 Theory

The complete model description of magnetic levitation platform can be derived on the form [1]

$$\dot{x} = Ax + Bf(x,u) \tag{1a}$$

$$y = Cx \tag{1b}$$

where $x = [x, y, z, \alpha, \beta, \gamma, \dot{x}, \dot{y}, \dot{z}, \dot{\alpha}, \dot{\beta}, \dot{\gamma}]$ are the Cartesian coordinates and the Euler rotation angles of the center of gravity of the levitating magnet, and u is a vector containing all the solenoid currents. The system matrices are defined as

$$\mathbf{A} = \begin{bmatrix} \mathbf{0}_{6\times6} & \mathbf{I}_6 \\ \mathbf{0}_{6\times6} & \mathbf{0}_{6\times6} \end{bmatrix} \tag{1c}$$

$$\mathbf{B} = \begin{bmatrix} \mathbf{0}_{6\times6} \\ \mathbf{I}_6 \end{bmatrix} \tag{1d}$$

$$\mathbf{C} = \mathbf{I}_{12} \tag{1e}$$

$$f(x,u) = \begin{bmatrix} m\mathbf{I}_3 & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathcal{I} \end{bmatrix}^{-1} \cdot \left(\begin{bmatrix} \boldsymbol{F(x,u)} \\ \boldsymbol{\tau(x,u)} \end{bmatrix} - \begin{bmatrix} \boldsymbol{F_g} \\ \boldsymbol{\omega} \times \mathcal{I}\boldsymbol{\omega} \end{bmatrix} \right) \tag{1f}$$

where $f(x,u)$ is nonlinear function of the magnetic force $\boldsymbol{F(x,u)}$ and the torque $\boldsymbol{\tau(x,u)}$ acting on the magnet as a function of the states and input. The magnetic force and torque can be approximated with the following equations

$$\boldsymbol{F} = \frac{2\pi Rl}{n} \sum_{i=1}^{n} \boldsymbol{K}(\alpha,\beta,\gamma) \times \boldsymbol{B}_{\text{base}}\,(x_i, y_i, z_i)$$

$$\boldsymbol{\tau} = \frac{2\pi Rl}{n\mu_0} \sum_{i=1}^{n} \boldsymbol{J}(\alpha,\beta,\gamma) \times \boldsymbol{B}_{\text{base}}\,(x_i, y_i, z_i)$$

Here, $\{xi, yi, zi\}$ are n evenly distributed points on the circumference of a cross-section of the center of the levitating magnet, perpendicular to its central axis. The relation between the magnetization surface current K and the magnetic polarization J is given by

$$\boldsymbol{K} = \frac{\boldsymbol{J} \times \boldsymbol{n}}{\mu_0} \tag{2}$$

n represent the normal vector to J and μ_0 the permeability in vacuum. Furthermore, the expression for the total magnetic field produced by the base is given in Cartesian coordinates by

$$B_{base}(x_i, y_i, z_i) := \sum_{j=1}^{n_s+n_m} B(x_i - \chi_j, y_i - \gamma_j, z_i - \zeta_j) \tag{3}$$

where $(\chi_j,\ \gamma_j,\ \zeta_j)$ are the Cartesian coordinates of the center of the j´th solenoid/permanent magnet, and n_s and n_m are the total numbers of solenoids and permanent magnets, respectively. For the i-th point on the circumference of the center of the levitating magnet and the j-th center point of the solenoid/permanent magnet, the comprehensive magnetic field is expressed through Cartesian coordinates. These coordinates are derived systematically from cylindrical coordinates, offering a comprehensive representation of the magnetic field produced at each specific point.

$$B(x_i-\chi_j, y_i-\gamma_j, z_i-\zeta_j) = \begin{bmatrix} B_\rho \cos B_\phi \\ B_\rho \sin B_\phi \\ B_z \end{bmatrix} \tag{4}$$

where

$$\begin{bmatrix} B_\phi \\ B_\rho \\ B_z \end{bmatrix} = \begin{bmatrix} \phi \\ -\frac{z_i-\zeta_j}{\rho_{ij}} c_{ij} k_{ij} \left[\mathrm{K}\left(k_{ij}^2\right) - \frac{\rho_{ij}^2+R^2+(z_i-\zeta_j)^2}{(\rho_{ij}-R)^2+(z_i-\zeta_j)^2} \mathrm{E}\left(k_{ij}^2\right) \right] \\ c_{ij} k_{ij} \left[\mathrm{K}\left(k_{ij}^2\right) - \frac{\rho_{ij}^2-R^2+(z_i-\zeta_j)^2}{(\rho_{ij}-R)^2+(z_i-\zeta_j)^2} \mathrm{E}\left(k_{ij}^2\right) \right] \end{bmatrix} \tag{5}$$

given us the expression for the total magnetic field from the base as

$$\boldsymbol{B}_{\text{base}}\left(x_i, y_i, z_i\right) = \sum_{j=1}^{n_s+n_m} \begin{bmatrix} -\frac{z_i-\zeta_j}{\rho_{ij}} c_{ij} k_{ij} \left[\mathrm{K}\left(k_{ij}^2\right) - \frac{\rho_{ij}^2+R^2+(z_i-\zeta_j)^2}{(\rho_{ij}-R)^2+(z_i-\zeta_j)^2} \mathrm{E}\left(k_{ij}^2\right) \right] cos(\phi_{ij}) \\ -\frac{z_i-\zeta_j}{\rho_{ij}} c_{ij} k_{ij} \left[\mathrm{K}\left(k_{ij}^2\right) - \frac{\rho_{ij}^2+R^2+(z_i-\zeta_j)^2}{(\rho_{ij}-R)^2+(z_i-\zeta_j)^2} \mathrm{E}\left(k_{ij}^2\right) \right] sin(\phi_{ij}) \\ c_{ij} k_{ij} \left[\mathrm{K}\left(k_{ij}^2\right) - \frac{\rho_{ij}^2-R^2+(z_i-\zeta_j)^2}{(\rho_{ij}-R)^2+(z_i-\zeta_j)^2} \mathrm{E}\left(k_{ij}^2\right) \right] \end{bmatrix} \tag{6}$$

where

$$\sum_{j=1}^{n_s+n_m} \rho_{ij} = \sqrt{(x_i-\chi_j)^2+(y_i-\gamma_j)^2} \quad and \quad \sum_{j=1}^{n_s+n_m} \phi_{ij} = \arctan \frac{y_i-\gamma_j}{x_i-\chi_j}$$

and

$$\sum_{j=1}^{n_s+n_m} k_{ij}^2 = \frac{4R\rho_j}{(R+\rho_j)^2+(z_i-\zeta_j)^2} \quad and \quad \sum_{j=1}^{n_s+n_m} c_{ij} = \frac{\mu_0}{4\pi\sqrt{R\rho_j}} NI \tag{7}$$

For further details read [1].

2.1 Equilibrium

The equilibrium of the system is found by

$$\dot{x} = Ax_{eq} + Bf(x_{eq}, u_{eq}) = 0 \tag{8}$$

Moreover, to seek an analytical solution for the equilibrium point, certain assumptions are made. Initially, we assume that all states at the equilibrium point are zero, except for the vertical position z. Additionally, we presume the absence of torque or rotation at equilibrium, with only a force from the magnetic field countering gravity in the z direction. With these considerations, we express the force using the distributive property and the cross product to simplify.[2].

$$\boldsymbol{F} = \frac{2\pi Rl}{n} \sum_{i=1}^{n} \sum_{j=1}^{n_s+n_m} \boldsymbol{K}(\alpha, \beta, \gamma) \times B(x_i - \chi_j, y_i - \gamma_j, z_i - \zeta_j) \tag{9}$$

Further, we express the force only in the z direction as

$$\boldsymbol{F}_z = \frac{2\pi Rl}{n} \sum_{i=1}^{n} \sum_{j=1}^{n_s+n_m} K_1(\alpha, \beta, \gamma) B_\rho \sin B_\phi - K_2(\alpha, \beta, \gamma) B_\rho \cos B_\phi \tag{10}$$

This needs to equal the force from gravity at equilibrium

$$F_z = F_g \tag{11}$$

$$\frac{2\pi Rl}{n} \sum_{i=1}^{n} \sum_{j=1}^{8} K_1(\alpha, \beta, \gamma) B_\rho \sin B_\phi - K_2(\alpha, \beta, \gamma) B_\rho \cos B_\phi = -mg \tag{12}$$

since B_ρ contains complete elliptic integral of the first and second kind the solution for analytical solution of the equilibrium point is not straightforward. We assume that the center of levitating magnet coincide with the fictional center that includes all the permanent magnets/solenoids, such that $K_1 = |K|sin(\phi)$ and $K_2 = |K|cos(\phi)$

$$\boldsymbol{F}_z = \frac{2\pi Rl}{n} \sum_{i=1}^{n} \sum_{j=1}^{n_s+n_m} K(\alpha, \beta, \gamma) B_\rho \sin^2(B_\phi) - K(\alpha, \beta, \gamma) B_\rho \cos^2(B_\phi) \tag{13}$$

using the double angle formulae from trigonometric identities [3] we get

$$\boldsymbol{F}_z = \frac{2\pi Rl}{n} \sum_{i=1}^{n} \sum_{j=1}^{n_s+n_m} -|K|(\alpha, \beta, \gamma) B_\rho cos(2B_\phi) \tag{14}$$

which should equal the gravity force. Hence we can set up the cost function to minimize as

$$\boldsymbol{J}(z) = || \frac{2\pi Rl}{n} \sum_{i=1}^{n} \sum_{j=1}^{n_s+n_m} -|K|(\alpha, \beta, \gamma) B_\rho cos(2B_\phi) + mg||^2 \tag{15}$$

2.2 Linearized model strucuture

The model is linearized numerically due to the presence of equations involving double summations and complete elliptic integrals [1] and is on the form

$$A = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 860.1127 & 0 & 0 & 0 & 8.3692 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 860.1127 & 0 & -8.3692 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1720.2262 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 6.2e-06 & -875169.4637 & 2.3e-06 & -294761.027 & 5e-06 & -3.6e-06 & 0 & 0 & 0 & 0 & 0 & 0 \\ 875169.4637 & 2.9e-06 & 1e-07 & 1.9e-06 & -294761.027 & 6.9e-06 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

The linearized model structure can be further utilized for the system to estimate grey-box models 3.3, enabling their application in feedback linearization.

3 System identification

System identification is a methodology used to construct mathematical models of dynamic systems by utilizing measurements of both input and output signals from the system [4].

The process of system identification involves the following steps:

1. Collect measurements of the input and output signals from the system, either in the time or frequency domain.
2. Choose an appropriate model structure for the system.
3. Utilize an estimation method to determine values for the adjustable parameters within the selected model structure.
4. Evaluate the estimated model to ensure it meets the specific requirements of your application.

3.1 Build models from data

System identification modeling requires selecting a model structure that accurately describes the system's behavior. A model structure represents a mathematical relationship between input and output variables, containing unknown parameters.

Various approaches can be employed to choose the model structure:

- Black-Box Modeling: This approach aims to find a model that reproduces measured data while maintaining simplicity.
- Grey-Box Modeling: In this approach, a specific model structure is desired, often derived from first principles. However, the numerical values of its parameters are unknown. The model structure is represented by a set of equations, and parameter values are estimated from data.

3.2 Black-box

In the black-box approach, linear and nonlinear models of various structures are estimated without requiring the equation of motion for the system. This method is particularly useful when the primary objective is to fit data without a specific concern for the mathematical structure of the model. The black-box modeling process is often characterized by a trial-and-error approach, where parameters of different structures are estimated and compared. Typically, the process begins with a simple linear model structure and advances to more complex structures. In black-box modeling, the selection of a model structure may be influenced by one's understanding of a particular structure or specific application requirements.

The configuration of a model structure in black-box modeling can be achieved through the determination of the model order. For linear AutoRegressive with eXogenous input (ARX) models, the model order can be estimated from the available data.

3.3 Grey-box

In certain scenarios, it is possible to deduce the model structure based on physical principles. The grey-box approach involves using available data to estimate the values of unknown parameters within the specified model structure. The model structure is defined by a set of differential or difference equations in MATLAB, with initial guesses for the unknown parameters.

The construction of grey-box models typically involves the following steps:

- Creating a Template Model Structure: Setting up an initial framework for the model structure.
- Configuring Model Parameters: Setting initial values for the model parameters and specifying any constraints if necessary.
- Applying Estimation Method: Employing an estimation method on the model structure to compute the values of the model parameters.

3.4 Evaluate model quality

Following the model estimation, the evaluation of model quality involves analyzing model uncertainty through various means:

- Comparing model response to measured response: Evaluating how well the model response fits with the measured output for the same input signal.
- Analyzing Residuals: Examining the residuals, which are the differences between the model predictions and the actual measurements.
- Analyzing Model Uncertainty: Evaluating the uncertainty linked to the estimated model.

Typically, the quality of a model is evaluated by comparing its response to the measured output for the same input signal. The fit percentage provides a measure of agreement between the model response and the measured output: a fit percentage of 100 indicates a perfect match, while 0 suggests a poor fit.

3.5 ARX

An ARX model is a combination of an autoregressive model (AR) and an exogenous input model (X) [5]. It used to represent the dynamics of a system and is commonly used in control engineering to model and analyze dynamic systems.

An autoregressive model is a type of statistical model that expresses a time series as a linear combination of its past values and a stochastic process. It is mathematically represented by the following equation:

$$y(t) = c + a_1 y(t-1) + a_2 y(t-2) + \cdots + a_p y(t-p) + e(t)$$

where $y(t)$ represents the value of the time series at time t, c is a constant term, $a_1, a_2, \ldots, a_p$ are the autoregressive coefficients, $y_1, y_2, \ldots, y_p$ are the past values of the time series, and $e(t)$ is a random error term.

An exogenous input model describes a time series as a linear combination of its past values and a set of exogenous inputs, which are external variables. The model is represented by the following equation:

$$y(t) = c + b_1 u_1(t) + b_2 u_2(t) + \cdots + b_q u_q(t) + e(t)$$

where y(t) is the value of the time series at t, c is a constant term, $u_1(t), u_2(t), \ldots, u_q(t)$ are the exogenous input variables, and $b_1, b_2, \ldots, b_q$ are the coefficients that capture the relationship between the input variables and output.

An ARX model results from combining an AutoRegressive (AR) model with an eXogenous (X) model. This model is expressed by the following equation:

$$y(t) = c + a_1 y(t-1) + a_2 y(t-2) + \cdots + a_p y(t-p) + b_1 u_1(t) + b_2 u_2(t) + \cdots + b_q u_q(t) + e(t)$$

ARX time series models provide a linear representation of a dynamic system in discrete time. Transforming a model into ARX form serves as the foundation for numerous methods in process dynamics and control analysis. The following is the time series model with a single input and single output, where k is an index representing the time step.

$$y_{k+1} = \sum_{i=1}^{n_a=1} a_i y_{k-i+1} + \sum_{i=1}^{n_b=1} b_i u_{k-i+1} + e(t)$$

With $n_a = 3, n_b = 2, n_u = 1, n_y = 1$, the time series model is:

$$y_{k+1} = a_1 y_k + a_2 y_{k-1} + a_3 y_{k-2} + b_1 u_k + b_2 u_{k-1}$$

There may also ble multiple inputs and multiple outputs such as when $n_a = 1, n_b = 1, n_u = 2, n_y = 2$

$$y1_{k+1} = a_{11} y1_k + b_{11} u1_k + b_{12} u2_k \tag{16}$$

$$y2_{k+1} = a_{12} y2_k + b_{21} u1_k + b_{22} u2_k \tag{17}$$

3.6 ARMAX

Unlike the autoregressive with exogenous terms (ARX) model, the system structure ARMAX model includes the stochastic dynamics [6]. ARMAX models become particularly valuable when there are dominant disturbances entering the system early in the process, such as at the input stage. Compared to the ARX model, the ARMAX model offers greater flexibility in handling models that include disturbances.

The following equation shows the form of the ARMAX model

$$A(z)y(k) = B(z)u(k-n) + C(z)e(k)$$

where y(k) is the system outputs, u(k) is the system inputs, n is the system delay, e(k) is the system disturbance. A(z), B(z), C(z) are the polynomial with respect to the backward shift operator z^{-1}, $z^{-1}y(k) = y(k-1)$ and defined by the following equations

$$A(z) = 1 + a_1 z^{-1} + \cdots + a_{n_a} z^{-n_a}$$
$$B(z) = b_0 + b_1 z^{-1} + \cdots + b_{n_b-1} z^{-(n_b-1)}$$
$$C(z) = 1 + c_1 z^{-1} + \cdots + c_{n_c} z^{-n_c}$$

if we derive the equations for a SISO system we get

$$y_k + a_1 y_{k-1} + \cdots + a_{n_a} y_{k-n_a} = b_0 u(k-n) + b_1 u_{k-n-1} + \cdots + b_{n_b-1} u_{k-n-n_b-1} + e(k) + c_1 e(k-1) + \cdots + c_{n_c} e(k-n_c)$$

where n_a is the A order, n_b is the B order, n_c is the C order, n is the system delay and e(k) is the system disturbance.

4 System identification of the magnetic field

The Systems Identification Toolbox [7] offers a range of MATLAB functions, Simulink blocks, and a user-friendly app designed for dynamic system modeling, time-series analysis, and forecasting. This toolbox allows exploring dynamic relationships among measured variables, making it easy to create transfer functions, process models, and state-space models in continuous or discrete time, using data in the time or frequency domain. Additionally, the toolbox supports the forecasting of time series through various techniques, including AR, ARMA, and other linear and nonlinear autoregressive modeling methods.

In this project, we'll begin by focusing on a single-input, single-output (SISO) system before advancing to a multiple-input, multiple-output (MIMO) system. Start with preparing the data from the magnetic sensors by loading data.mat file into the MATLAB workspace by writing

```
load "data.mat"
```

Separate the data into the MATLAB workspace as inputs and outputs of your measurements by

```
data_x = DATA(1,:)';
data_ux = DATA(4,:)';

data_y = DATA(2,:)';
data_uy = DATA(5,:)';
```

Here $data_x$ and $data_y$ represent the measured magnetic field in x and y direction, and $data_{ux}$ and $data_{uy}$ are the control input for the permanent magnets controlling the corresponding direction.

Create an "iddata object" [8] to encapsulate input and output measurement data for the system you want to identify. The system identification function uses measurements to estimate a model. Model validation functions use input measurements for simulations and output measurements to calculate how accurately the estimated model response fits the original data.

```
Ts = 0.0002; %Sample time

% Create iddata
x = iddata(data_x, data_ux, Ts);
y = iddata(data_y, data_uy, Ts);

% First 12500 samples used for estimation
xe = x(1:12500);
ye = y(1:12500);

% Remaining samples used for validation
xv = x(12501:end);
yv = y(12501:end);
```

Open the System Identification app by writing

```
systemIdentification
```

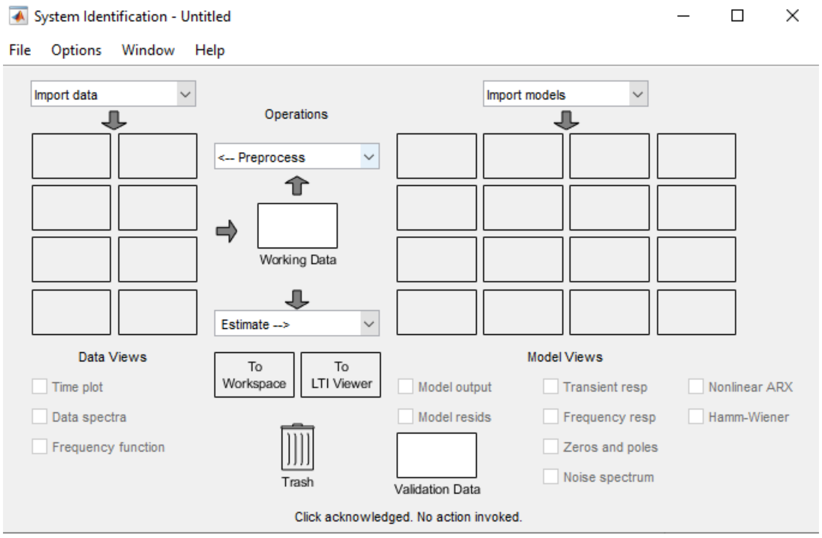

Figure 1: System identification app overview

Import the iddata object into the system identification app. Select **Import data** > **Data object** and enter "xe" and "xv" in the object field to import the estimation and validation data.

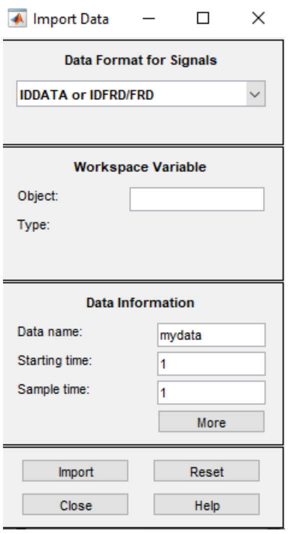

Figure 2: Import data overview

Drag the estimation data icon to the working data rectangle to "preprocess" the data, removing means, and filtering out the noise.

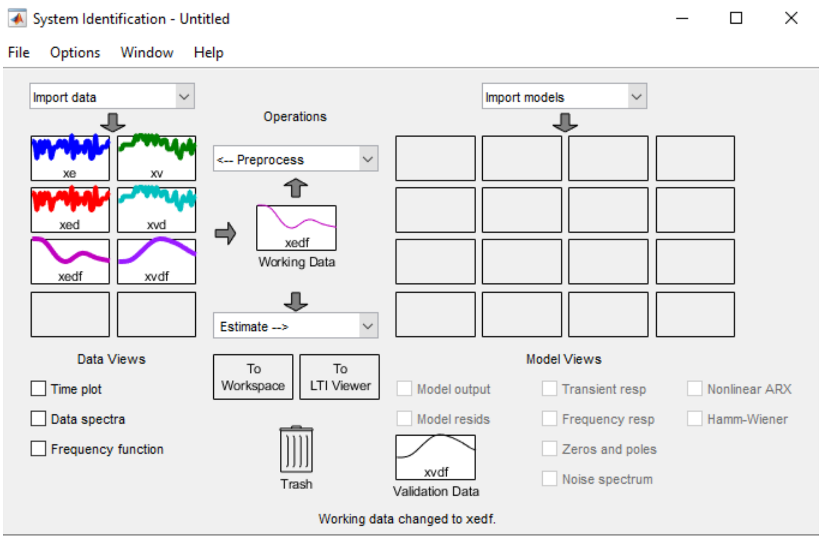

Figure 3: Estimation and validation reprocessed overview

Drag the validation data icon "xvdf" to the validation data rectangle and "xedf" icon to the working data rectangle. We can now use strategies for estimating accurate models. In system identification app, select **Estimate** > **Polynomial orders**. ARX is already selected by default.

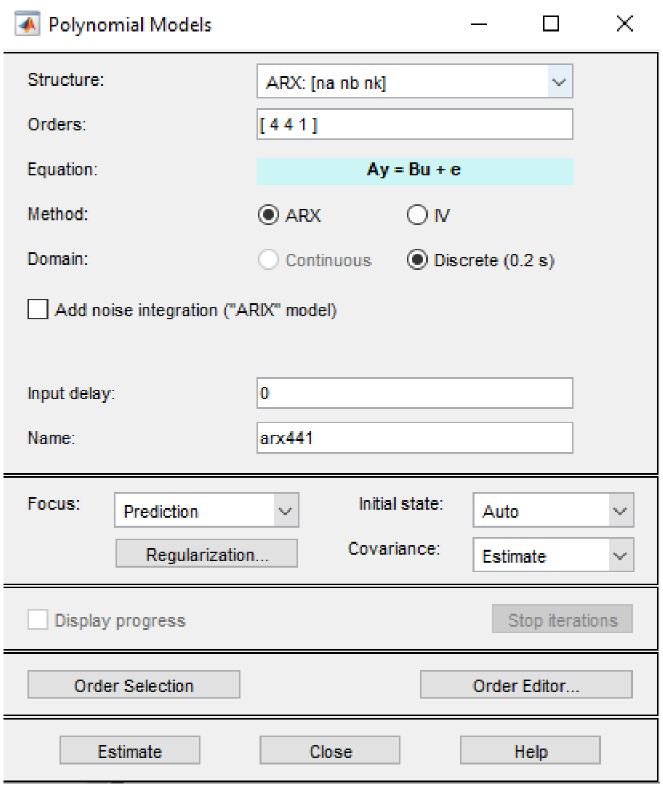

Figure 4: Polynomial structures

The black-box approach allows us to determine a model structure through trial and error. This involves experimenting with various combinations of polynomial orders to identify the most suitable configuration.

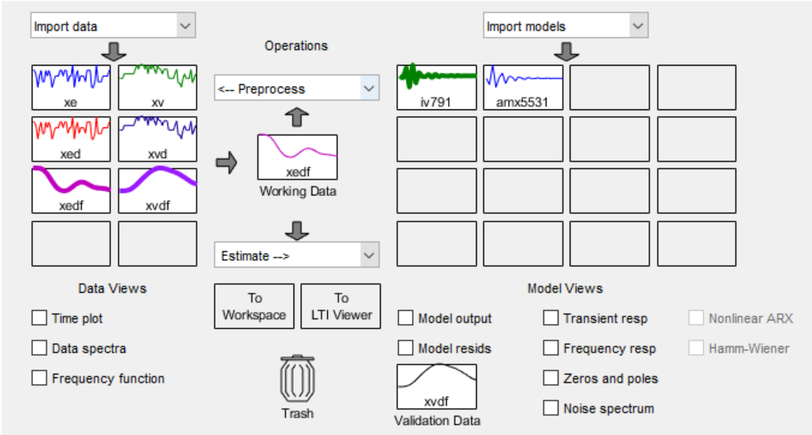

Figure 5: Model structure of IV and ARMAX

Here ”iv791” represent the model estimation with instrumental variable technique [9] with Polynomial orders: n_a=7 n_b=9 n_k=1. If we double click on ”iv791” we get the discrete-time ARX model and the fit to estimation data. Since we don't want to over fit the model to the estimation data we use the validation data to compare it with the estimated model structure. The same method listed for the ARX can also be done for ARMAX model structures. Change to ARMAX in structure field and estimate with the default order amx2222. We can repeat the steps using high orders to find a better fit between the model structure and validation data.

5 Results

5.1 Black-box structure estimation of the magnetic field in x direction

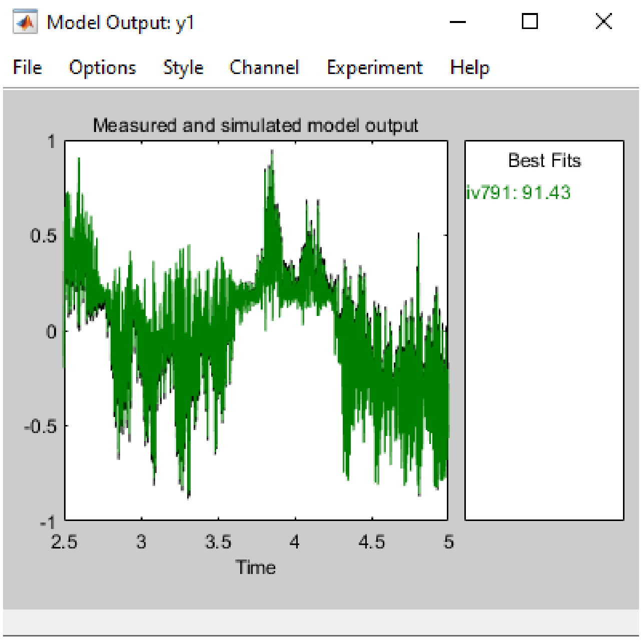

Figure 6: Comparison between the proposed model structure IV and validation data in x-direction measured in millitesla [mT]

```
Discrete-time ARX model: A(z)y(t) = B(z)u(t) + e(t)


A(z) = 1 - 1.009 z^-1 - 0.3897 z^-2 + 0.003786 z^-3 + 0.3113 z^-4 +
     0.2901 z^-5 + 0.1087 z^-6 - 0.3077 z^-7

B(z) = -0.08631 z^-1 + 0.348 z^-2 - 0.5991 z^-3 + 0.6246 z^-4 -
    0.4755 z^-5 + 0.2641 z^-6 - 0.08941 z^-7 + 0.01683 z^-8 -
    0.003229 z^-9


Polynomial orders: na=7, nb=9, nk=1
Fit to estimation data: 99.86% (prediction focus)
Fit to validation data: 95.09%
```

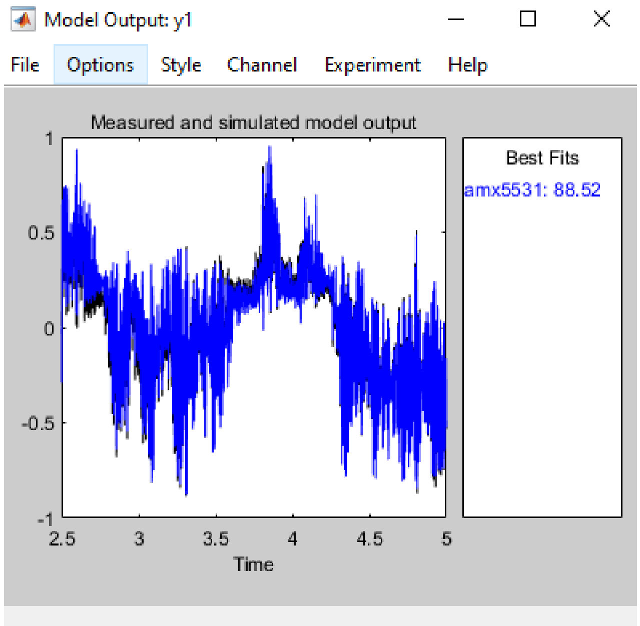

Figure 7: Comparison between the proposed model structure ARMAX and validation data in x-direction measured in millitesla [mT]

```
Discrete-time ARMAX model: A(z)y(t) = B(z)u(t) + C(z)e(t)

A(z) = 1 - 1.892 z^-1 + 0.6592 z^-2 - 0.03099 z^-3 + 0.8335 z^-4 -
    0.5699 z^-5
B(z) = -0.05332 z^-1 + 0.2158 z^-2 - 0.3297 z^-3 + 0.2261 z^-4 -
    0.05882 z^-5
C(z) =  1 + 0.6359 z^-1 + 0.01475 z^-2 + 0.3788 z^-3


Polynomial orders: na=5, nb=5, nc=3  nk=1
Fit to estimation data: 99.96% (prediction focus)
Fit to validation data: 98.61%
```

5.2 Black-box structure estimation of the magnetic field in y direction

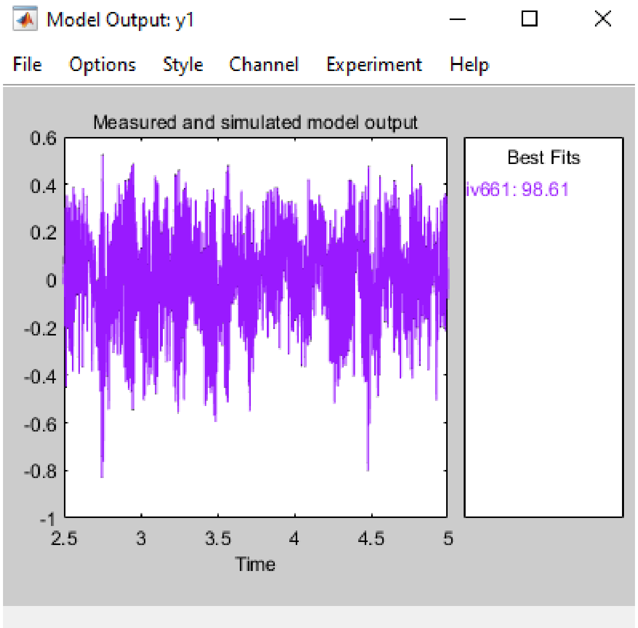

Figure 8: Comparison between the proposed model structure IV and validation data in y-direction measured in millitesla [mT]

```
Discrete-time ARX model: A(z)y(t) = B(z)u(t) + e(t)


A(z) = 1 - 1.787 z^-1 + 0.1515 z^-2 + 0.906 z^-3 + 0.4957 z^-4 -
    1.118 z^-5 + 0.3564 z^-6
B(z) = -0.06409 z^-1 + 0.2797 z^-2 - 0.488 z^-3 + 0.4251 z^-4 -
    0.1835 z^-5 + 0.03085 z^-6


Polynomial orders: na=6, nb=6, nk=1
Fit to estimation data: 99.92% (prediction focus)
Fit to validation data: 88.52%
```

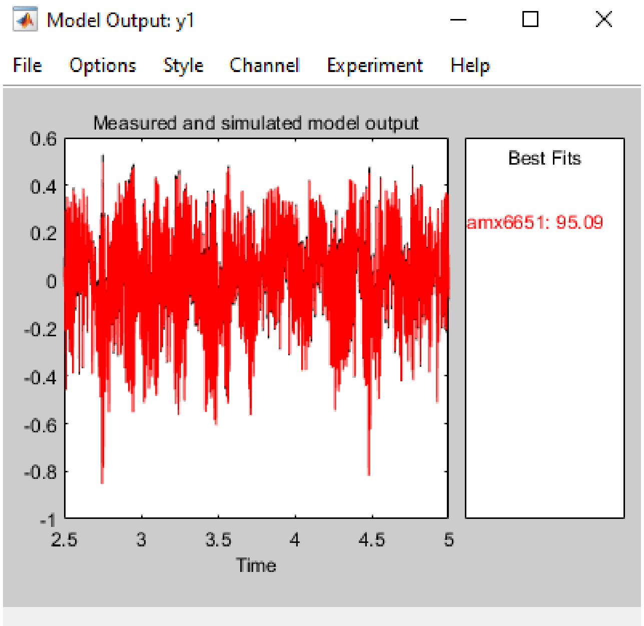

Figure 9: Comparison between the proposed model structure ARMAX and validation data in y-direction measured in millitesla [mT]

```
Discrete-time ARMAX model: A(z)y(t) = B(z)u(t) + C(z)e(t)

A(z) = 1 - 2.371 z^-1 + 1.23 z^-2 + 0.6027 z^-3 + 0.2027 z^-4 -
    1.196 z^-5 + 0.5317 z^-6
B(z) = -0.0586 z^-1 + 0.2811 z^-2 - 0.5461 z^-3 + 0.5382 z^-4 -
    0.2692 z^-5 + 0.0546 z^-6
C(z) =  1 + 1.253 z^-1 + 0.4528 z^-2 + 0.1162 z^-3 - 0.0982 z^-4 -
    0.01454 z^-5


Polynomial orders: na=6, nb=6, nc=5,  nk=1
Fit to estimation data: 99.98% (prediction focus)
Fit to validation data: 95.09%
```

5.2.1 Black-box structure estimation of the magnetic field in x- and y-direction

Here, the model structure for the magnetic field in both the x- and y-direction is estimated simultaneously [appendix A].

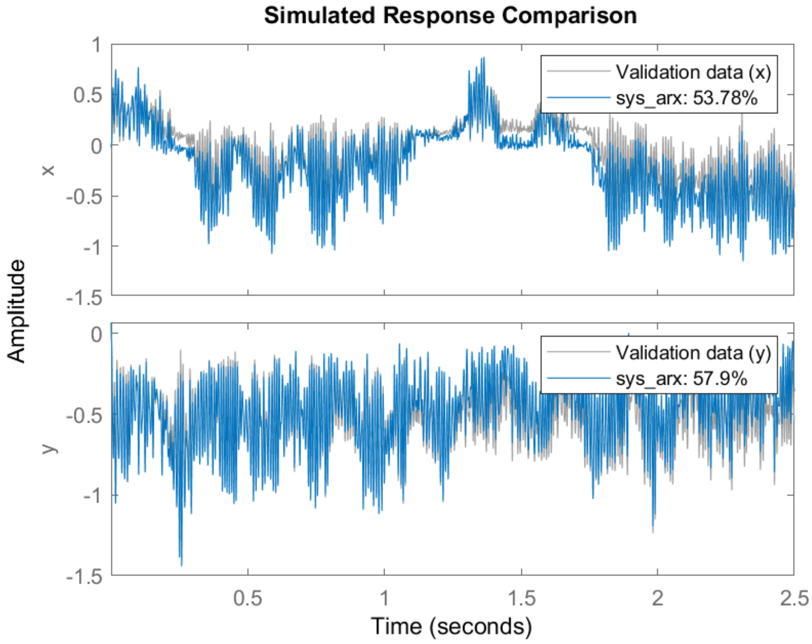

Figure 10: Comparison between the proposed model structure ARX and validation data in x- and y-direction measured in millitesla [mT]

```
Discrete-time ARX model:
  Model for output "y1": A(z)y_1(t) = - A_i(z)y_i(t) + B(z)u(t) +
     e_1(t)
    A(z) = 1 - 3.749 z^-1 + 5.345 z^-2 - 3.43 z^-3 + 0.8348 z^-4

    A_2(z) = -0.001102 z^-1 + 0.003175 z^-2 - 0.00311 z^-3 +
       0.001033 z^-4

    B1(z) = 5.904e-05 z^-1 - 5.897e-05 z^-2

    B2(z) = 7.989e-08 z^-1 - 1.096e-07 z^-2

  Model for output "y2": A(z)y_2(t) = B(z)u(t) + e_2(t)
    A(z) = 1 - 3.755 z^-1 + 5.358 z^-2 - 3.441 z^-3 + 0.8377 z^-4

    B1(z) = 0

    B2(z) = 5.688e-05 z^-1 - 5.687e-05 z^-2

Sample time: 0.0002 seconds

Parameterization:
   Polynomial orders:   na=[4 4;0 4]   nb=[2 2;0 2]   nk=[1 1;1 1]
   Number of free coefficients: 18
   Use "polydata", "getpvec", "getcov" for parameters and their
      uncertainties.
```

```
Status:
Estimated using ARX on time domain data "maglev_estf".
Fit to estimation data: [99.99;99.99]% (prediction focus)
FPE: 4.476e-19, MSE: 1.336e-09
```

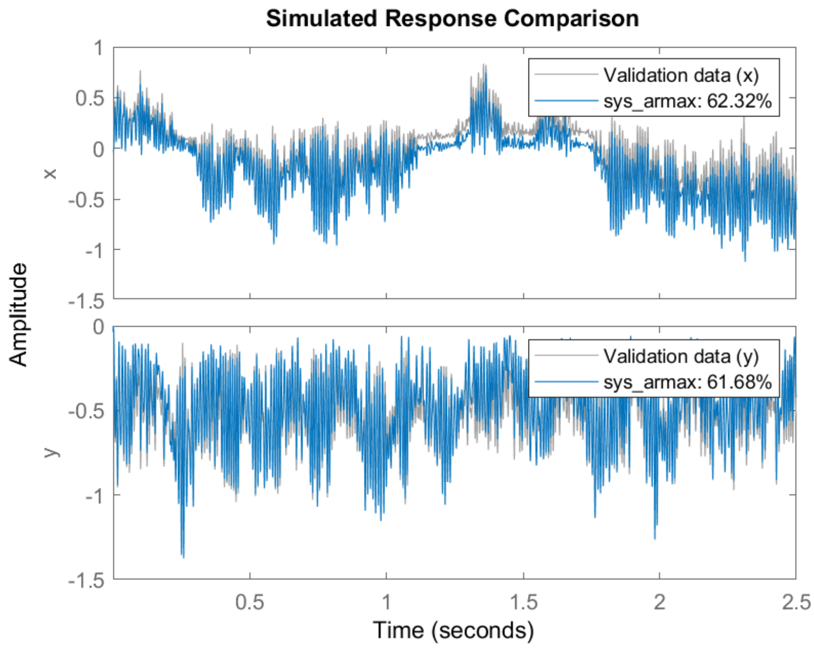

Figure 11: Comparison between the proposed model structure ARMAX and validation data in x- and y-direction measured in millitesla [mT]

```
Discrete-time ARMAX model:
  Model for output "y1": A(z)y_1(t) = - A_i(z)y_i(t) + B(z)u(t) + C
     (z)e_1(t)
    A(z) = 1 - 1.837 z^-1 + 0.8728 z^-2

    A_2(z) = 0.01479 z^-1 - 0.01064 z^-2

    B1(z) = 0.0001222 z^-1

    B2(z) = 1.501e-05 z^-1

    C(z) = 1 + z^-1

  Model for output "y2": A(z)y_2(t) = B(z)u(t) + C(z)e_2(t)
    A(z) = 1 - 0.8305 z^-1 - 0.004239 z^-2

    B1(z) = 0

    B2(z) = 0.001036 z^-1

    C(z) = 1 + z^-1

Sample time: 0.0002 seconds
```

```
Parameterization:
   Polynomial orders:   na=[2 2;0 2]   nb=[1 1;0 1]   nc=[1;1]   nk
      =[1 1;1 1]
   Number of free coefficients: 11
   Use "polydata", "getpvec", "getcov" for parameters and their
      uncertainties.

Status:
Estimated using ARMAX on time domain data "maglev_estf".
Fit to estimation data: [99.76;96]% (prediction focus)
FPE: 3.965e-11, MSE: 8.178e-05
```

6 Conclusion

In conclusion, the estimated models provide a good fit and closely match the observed trend in the validation data. Additionally, lower-order ARMAX models demonstrate comparable accuracy to higher-order ARX models. However, it is crucial to note that the model responses are significantly impacted by noise. This emphasizes the importance of carefully considering and addressing the effects of noise in the modeling process for more robust and reliable results.

Additionally, estimating model structures in the y-direction yields a better fit compared to the model structures in the x-direction. This observation holds true in both SISO and "MIMO" cases.

Considering the results, it's essential to highlight that the fit is reduced when comparing Multiple Input Multiple Output (MIMO) models with Single Input Single Output (SISO) models. This indicates that challenges and complexities in modeling are present in both SISO and MIMO configurations. It underscores the importance of a thoughtful approach and potential adjustments in the modeling process to improve the overall accuracy of system representation.

7 Further work

The model structure of the complete model, as referenced in [1], can be estimated using a linear grey-box approach. This can be implemented efficiently by following the method outlined in [10], utilizing a combination of simulation data and white noise. This approach provides an approximate representation of real sensor data, contributing to a more accurate model.

Furthermore, for future work, one potential option is to explore advanced nonlinear model estimation techniques, such as nlarx (Nonlinear AutoRegressive with eXogenous input) and SINDy (Sparse Identification of Nonlinear Dynamics). These methods can offer an understanding of the nonlinear dynamics inherent in the system, which linear models might fail to capture.

Additionally, considering the impact of external factors or disturbances on the model's performance could be a valuable direction for future investigations. Including robustness analysis or disturbance modeling in the framework can improve the model's predictive capabilities in real-world scenarios.

Moreover, exploring methods for validating the model and quantifying uncertainty can enhance model reliability. Techniques like cross-validation calculate the model's performance on unseen data and provide confidence intervals for estimated parameters.

Additionally, investigating the potential for model simplification or reduced-order modeling may be beneficial. This involves creating simpler models that save essential system dynamics, improving computational efficiency for real-time applications.

In summary, future work could focus on advanced nonlinear modeling techniques, investigating the impact of external factors, improving model validation and uncertainty quantification, and exploring model simplification for enhanced practical applicability.

References

[1] Hans Alvar Engmark and Kiet Tuan Hoang. "Modeling and Control of a Magnetic Levitation Platform". In: (2022). URL: https://www.sciencedirect.com/science/article/pii/S240589632300705X?via%3Dihub.

[2] "Vector Cross Product Distributes over Addition". In: (). URL: https://proofwiki.org/wiki/Vector_Cross_Product_Distributes_over_Addition.

[3] "trigonometric identities". In: (). URL: https://en.wikipedia.org/wiki/List_of_trigonometric_identities.

[4] "System Identification Overview". In: (). URL: https://www.mathworks.com/help/ident/gs/about-system-identification.html.

[5] "ARX Time Series Model". In: (). URL: https://apmonitor.com/pds/index.php/Main/ARXTimeSeries.

[6] "ARMAX Model Definitions (Advanced Signal Processing Toolkit or Control Design and Simulation Module)". In: (). URL: https://www.ni.com/docs/en-US/bundle/labview-control-design-and-simulation-module/page/lvsysidconcepts/modeldefinitionsarmax.html.

[7] "System Identification Toolbox". In: (). URL: https://www.mathworks.com/help/ident/index.html?s_tid=CRUX_lftnav.

[8] "iddata". In: (). URL: https://www.mathworks.com/help/ident/ref/iddata.html.

[9] "Picking Instrumental Variables for System Identification". In: (). URL: https://www.mathworks.com/help/ident/ug/picking-instrumental-variables-for-system-identification.html.

[10] "Estimate Linear Grey-Box Models". In: (). URL: https://www.mathworks.com/help/ident/ug/estimating-linear-grey-box-models.html.

Appendix

A MATLAB

```
%% MIMO
clf

% Time-domain data
maglev = iddata([data_x, data_y], [data_ux, data_uy], Ts);
% figure(1)
% idplot(maglev)

% Frequency Plot
maglev_fft = fft(maglev);
opt = iddataPlotOptions('frequency','identpref');
opt.PhaseVisible = 'off';
% figure(2)
% plot(maglev_fft, opt)
filter = [0, 1000];

% First 12500 samples used for estimation
maglev_est = iddata([data_x(1:12500), data_y(1:12500)], [data_ux
    (1:12500), data_uy(1:12500)], Ts);
maglev_estf = idfilt(maglev_est, filter);
maglev_est.InputName  = {'ux';'uy'};
maglev_est.OutputName = {'x';'y'};

% Remaining samples used for validation
maglev_val = iddata([data_x(12501:25000), data_y(12501:25000)], [
    data_ux(12501:25000), data_uy(12501:25000)], Ts);
maglev_valf = idfilt(maglev_val, filter);
maglev_val.InputName  = {'ux';'uy'};
maglev_val.OutputName = {'x';'y'};


% Orders

% ARX
% na = [4 4; 0 4];
% nb = [2 2; 0 2];
% nk = [1 1; 1 1];


% ARMAX
na = [2 2; 0 2];
nb = [1 1; 0 1];
nc = [1; 1];
nk = [1 1; 1 1];

% ARX
sys_arx = arx(maglev_estf, [na nb nk]);

% ARMAX
sys_armax = armax(maglev_estf, [na nb nc nk]);

% Test
figure(4)
```

```
compare(maglev_valf, sys_arx)
figure(5)
compare(maglev_valf, sys_armax)
```