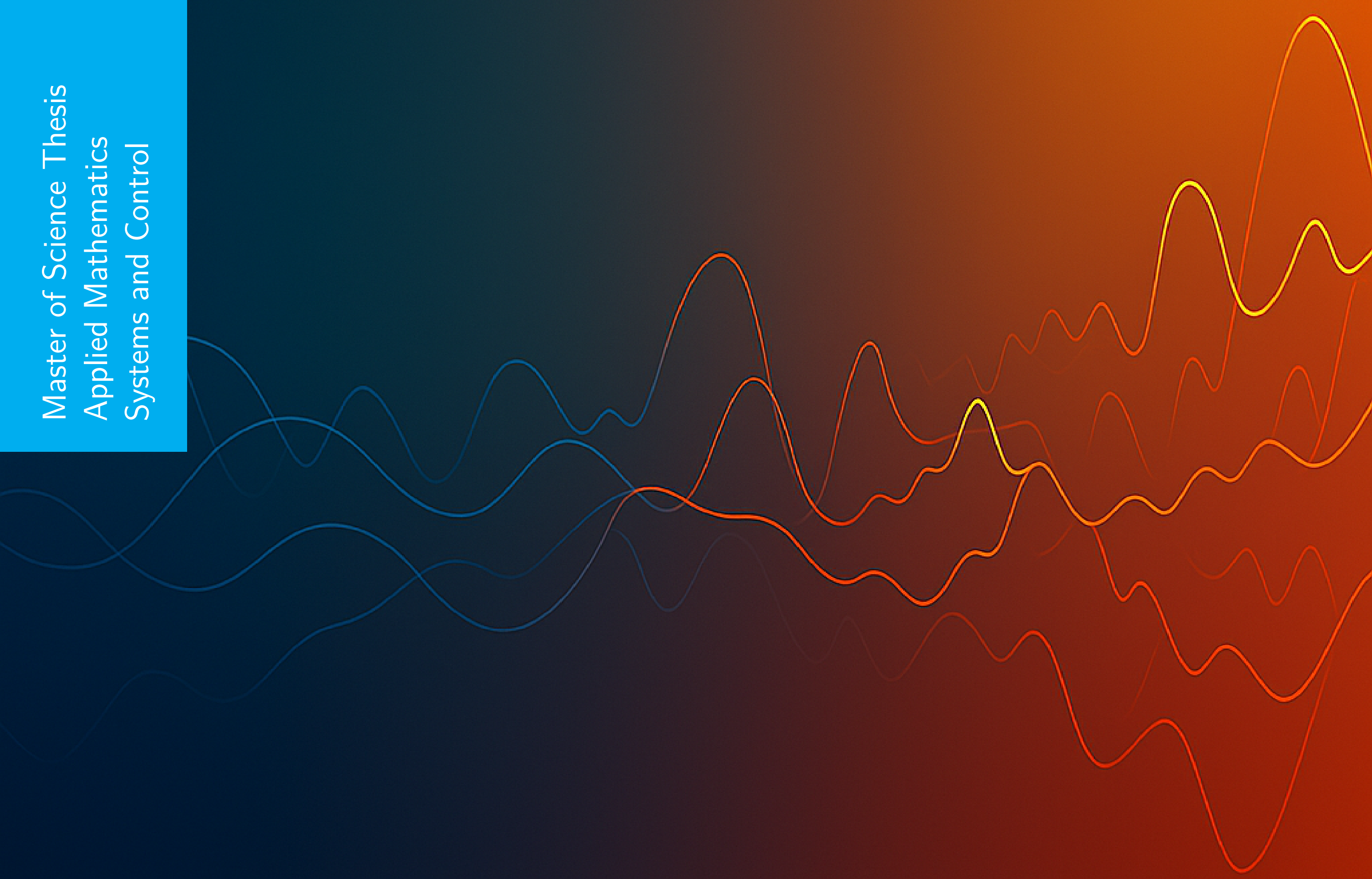
Data-Driven Control

Beyond ARX: Towards ARMAX in Subspace Predictive Control

R.W. Van Weelden

Master of Science Thesis
Applied Mathematics
Systems and Control



Delft University of Technology

Delft Center for Systems and Control

Data-Driven Control

Beyond ARX: Towards ARMAX in Subspace Predictive Control

Master of Science Thesis

For the degree of Master of Science in Systems and Control and Applied Mathematics at Delft University of Technology

R.W. Van Weelden

November 4, 2025

Faculty of Mechanical Engineering (ME)
Electrical Engineering, Mathematics and Computer Science (EEMCS)
Delft University of Technology

Abstract

The field of control engineering has evolved significantly in response to the increasing complexity and uncertainty of modern technological systems. Traditional control methods, which rely on precise analytical models derived from first principles, often encounter limitations when applied to systems with unknown dynamics, time-varying parameters, or unmeasured disturbances. These challenges have motivated the development of data-driven control methodologies, which utilise the growing availability of input-output data to learn a control law directly from data, without the need for an explicit model.

Among the various data-driven approaches, Subspace Predictive Control (SPC) integrates subspace identification with Model Predictive Control (MPC) into a unified data-driven framework. The classical SPC formulation is based on an AutoRegressive with eXogenous input (ARX) model, which restricts its ability to capture coloured noise and complex stochastic dynamics.

This thesis investigates whether SPC can be extended to AutoRegressive Moving Average with eXogenous input (ARMAX) models to enhance noise modelling and control performance. The research addresses two key questions: from a theoretical perspective, how ARMAX models can be integrated into the SPC framework to achieve improved noise representation; and from a practical perspective, how ARMAX-based SPC can be applied to a real-life system exhibiting an anti-resonance.

The proposed framework reformulates the SPC data and prediction equations to include the ARMAX structure and employs Extended Recursive Least Squares for online identification. Both simulation studies and laboratory experiments on an inertia-spring-damping system were conducted to evaluate reference tracking, computational cost, and numerical robustness.

The results demonstrate that lower-order ARMAX models outperform ARX models, achieving substantially lower Integral Absolute Error (IAE), Integral Squared Error (ISE), and Input Energy (InEn) while producing smoother control actions. For higher-order models, however, both methods show comparable control performance, as the deterministic part of the system dynamics becomes well identified. Importantly, the computational cost of the ARMAX-based SPC remains of the same order as the ARX formulation for an equivalent number of parameters, confirming its feasibility for real-time implementation. These findings provide a foundation for future research on multi-input multi-output (MIMO) systems, hybrid SPC formulations, and stochastic predictive control frameworks.

Keywords – Data-Driven Control, Subspace Predictive Control, Model Predictive Control, System Identification, Recursive Least Squares, ARX, ARMAX, Markov Parameters.

Table of Contents

Acknowledgements **xi**

1 Introduction **1**

2 Theoretical Background **3**

2-1 Introduction to State-Space Modelling 4

2-2 Introduction to Polynomial Model Structures 5

2-3 Introduction to System Identification 6

2-4 Least Squares Formulation for the Identification of ARX Models 8

2-5 Relation Between Transfer Functions and ARX Models 11

2-5-1 Transfer Function of the Innovation Form 11

2-5-2 Transfer Function of the Predictor Form 13

2-6 Model Predictive Control for ARX Models 14

2-6-1 Unconstrained Optimisation Solution 17

2-7 Computation of the Least Squares Solution for ARX models 18

2-7-1 Regularised Recursive Least Squares Algorithm 20

2-8 Future Directions and Research Plan 23

2-8-1 Methodology 24

3 Subspace Predictive Control for ARMAX models **27**

3-1 Introduction to ARMAX Models 28

3-2 Least-Squares Formulation for the Identification of ARMAX Models 29

3-3 Model Predictive Control for ARMAX Models 32

3-4 Computation of the Least Squares Solution for ARMAX Models 34

3-5 Comparison Between ARX- and ARMAX-based SPC 36

4 Simulation Results **39**

4-1 Model Descriptions 39

4-2 Identification Results for ARX and ARMAX Models 42

4-2-1 Monte Carlo Variance Estimation in the Frequency Domain 42

4-2-2 First-Order Variance Approximation 48

4-3 Higher-Order ARX Models 58

4-4 Comparison Between ARX- and ARMAX-based Subspace Predictive Control . . . 61

5 Subspace Predictive Control in Practice: Lab Implementation **71**

5-1 System Description 71

5-2 Experiment Design 73

5-3 Identification Results for ARX and ARMAX Models 74

5-4 Comparing ARX- and ARMAX-based Subspace Predictive Control 79

6 Conclusion **91**

6-1 Integration of an ARMAX Model into the Subspace Predictive Control Framework 92

6-2 Application of ARMAX-Based Subspace Predictive Control to a Real-Time System 93

6-3 Discussion 94

6-4 Recommendations 95

A **97**

A-1 Additional Results: Inertia-Spring-Damper Model 98

A-2 Additional Results: Fifth Order SISO Model 104

Bibliography **111**

List of Figures

4-1 Bode plots of the dynamical and noise models for the Inertia-Spring-Damper system 40
4-2 Bode plots of the dynamical and noise models for the fifth order SISO model . . 41
4-3 Effect of noise realisation on variance convergence 43
4-4 Effect of modelling parameters p and σ on the convergence of the variance estimate 44
4-5 Effect of Monte Carlo runs M on the convergence of the Monte Carlo variance estimate for ARX and ARMAX 47
4-6 Ratio between first-order and Monte Carlo variance approximations 55
4-7 Bode plots of Monte Carlo means for ARX and ARMAX models with $M = 1000$ 57
4-8 Modal contributions of a higher-order ARX model at an anti-resonance 59
4-9 Boxplots performance metrics ISE, IAE and InEn for several values of p and different reference waves 65
4-10 Power spectral density of the input signal 65
4-11 Boxplots of ISE, IAE, and InEn for 50 Monte Carlo runs for ARX and ARMAX with the same number of parameters for a Sawtooth reference 67

5-1 Graphical representation of the laboratory setup 72
5-2 Chirp signal (8 to 1 Hz) and corresponding trended/detrended outputs 75
5-3 VAF for various identification algorithms with their training data 76
5-4 Identification results and Bode plots for $p = 10$ and $p = 20$ for the lab setup . . 78
5-5 Barplot and boxplots for the computation times of the identification and MPC step 81
5-6 Bar plots of the performance metrics: ISE, IAE, and InEn 84
5-7 Square-wave (0.5 Hz) reference tracking for ARX and ARMAX $N_p = 10$ 85
5-8 Sinusoidal (4.0 Hz) reference tracking for ARX and ARMAX $N_p = 10$ 86
5-9 Sawtooth (0.75 Hz) reference tracking for ARX and ARMAX $N_p = 10$ 87
5-10 Dynamical and noise models for ARX and ARMAX over time $p = 10$ 89

A-1 Boxplots of integral squared error (ISE) for 50 Monte Carlo runs for ARX and ARMAX with the same number of parameters for model A 99
A-2 Boxplots of integral absolute error (IAE) for 50 Monte Carlo runs for ARX and ARMAX with the same number of parameters for model A 101
A-3 Boxplots of input energy (InEn) for 50 Monte Carlo runs for ARX and ARMAX with the same number of parametersfor model A 103
A-4 Boxplots of integral squared error (ISE) for 50 Monte Carlo runs for ARX and ARMAX with the same number of parameters for model B 105
A-5 Boxplots of integral absolute error (IAE) for 50 Monte Carlo runs for ARX and ARMAX with the same number of parameters for model B 107
A-6 Boxplots of input energy (InEn) for 50 Monte Carlo runs for ARX and ARMAX with the same number of parameters for model B 109

List of Tables

3-1 Comparison of notation for ARX- and ARMAX-based SPC 36
3-2 Comparison of matrix dimensions for RLS (ARX) and ERLS (ARMAX) 37
3-3 Comparison of computational cost of RLS and ERLS 37

4-1 Absolute fit for dynamical and noise models with varying p 53
4-2 Relative fit for dynamical and noise models with varying p 53
4-3 Performance comparison of ARX and ARMAX for square-wave reference (0.5 Hz) 63
4-4 Performance comparison of ARX and ARMAX for sinusoidal reference (1 Hz) . . 63
4-5 Performance comparison of ARX and ARMAX for sinusoidal reference (2.61 Hz) 63
4-6 Performance comparison of ARX and ARMAX for sawtooth reference (0.5 Hz) . 63
4-7 Performance comparison of ARX and ARMAX for square-wave reference (0.5 Hz) with disturbance 69
4-8 Performance comparison of ARX and ARMAX for sinusoidal reference (1 Hz) with disturbance 69
4-9 Performance comparison of ARX and ARMAX for sinusoidal reference (2.61 Hz) with disturbance 69
4-10 Performance comparison of ARX and ARMAX for sawtooth reference (0.5 Hz) with disturbance 69

5-1 Parameters of the 1-DOF torsional system 71
5-2 VAF for various identification algorithms 77
5-3 Computational time statistics for ARX and ARMAX models across different (p, N_p) pairs 82
5-4 Performance metrics for ARX and ARMAX models across different (p, N_p) pairs . 84

Acknowledgements

This thesis was written as part of the Individual Double Degree Programme at Delft University of Technology in partial fulfilment of the requirements for the double master's in Applied Mathematics and Systems and Control.

I would like to thank my supervisors, Jan-Willem van Wingerden and Martin van Gijzen for their continuous guidance, many interesting meetings, and for providing critical yet insightful feedback and encouragement throughout this project. Their expertise in data-driven control, numerical methods, and linear algebra has been invaluable to the development of this work. I am also grateful for their advice and support beyond this thesis. I also wish to thank Johan Dubbeldam and Fritz Engeln for taking part in the assessment committee.

Finally, I would like to thank my family for their long-lasting support throughout my studies and my friends and peers for the academic discussions and study sessions that made this journey both enjoyable and inspiring.

Delft, University of Technology R.W. Van Weelden
November 4, 2025

Chapter 1

Introduction

Model Predictive Control (MPC) is a widely used technique for optimal control in engineering applications [37]. It predicts future system behaviour over a finite time horizon and solves an optimisation problem at each time step to compute the optimal control input. However, its performance strongly depends on the availability of an accurate model. For complex systems, deriving such a model from first principles can be difficult and time consuming [8], especially when the system exhibits non-linear behaviour, time-varying dynamics, or is affected by disturbances that are hard to model analytically [56].

In recent years, the growing availability of data in modern engineering systems has led to an increased interest in data-driven control methodologies. These approaches use data to learn the system dynamics and compute control actions, reducing the need for detailed analytical models [17]. Data-driven methods offer a flexible alternative to traditional model-based control, especially in complex or uncertain environments [8].

Data-driven control methods are generally classified into indirect and direct approaches [17]. Indirect methods first identify a predictive model from data, after which the model can be used in modern control techniques such as MPC, LQR, or state feedback. Direct methods bypass model identification and compute control policies directly from data.

An example of a direct method is Data-enabled Predictive Control (DeePC), the DeePC algorithm combines the system realisation and control step in a single optimisation algorithm using only data-matrices without formulating an explicit model [49]. This method is grounded in Willems Fundamental Lemma, which states that all future input-output trajectories of a linear system can be constructed from a sufficiently excited past input-output trajectory [49]. DeePC has been applied to, for example, quadcopter control [10] and battery charging management [63].

Subspace Predictive Control (SPC) is another data-driven control method. It combines subspace identification, which estimates the Markov parameters from input-output data, with the MPC framework into a single robust algorithm [11]. It has been applied in contexts such as nuclear reactor control [45] and data centre cooling [27]. Under noise-free conditions or with instrumental variables, SPC is mathematically equivalent to DeePC [9, 49]. Hybrid methods that combine elements of DeePC and SPC have also been developed, such as γ-DDPC [3] and GDPC [26]. A comparison of these four methods is presented in [55].

However, direct data-driven control methods methods often scale with the dimension of the dataset, making real-time implementation only feasible for small datasets [55]. An advantage of SPC lies in its scalability, as it is based on the identification of prediction matrices obtained by solving a least-squares (LS) problem directly from I/O data. This approach is easily scalable through a recursive LS implementation, and the computation of the control input requires significantly fewer optimisation variables than DeePC. As a result, SPC seems more suitable for real-time control for large-scale systems or embedded systems. That is why this study focuses on the SPC algorithm.

Despite this benefit, SPC algorithms typically assumed an underlying ARX model structure. Although ARX models are easy to estimate and implement, they have a limited noise model related to the system dynamics, which reduces their ability to accurately represent disturbance dynamics. Studies like [19, 44, 34] have shown that more complex models, such as ARMAX models, can achieve better identification performance while keeping the model order low.

Therefore, this study investigates whether incorporating a more complex polynomial model within the Subspace Predictive Control (SPC) framework can enhance control performance, particularly for systems affected by coloured noise and exhibiting anti-resonance behaviour. The objective is to evaluate whether the inclusion of an explicit noise model leads to more accurate predictions and improved closed-loop performance compared to the traditional ARX-based SPC formulation. Specifically, the research addresses two key questions: from a theoretical perspective, how ARMAX models can be integrated into the SPC framework to achieve improved noise representation; and from a practical perspective, how ARMAX-based SPC can be applied to a real-life system exhibiting an anti-resonance.

To address these questions in a structured manner, chapter 2 introduces the theoretical foundations of Subspace Predictive Control, including state-space and polynomial model formulations, their connection to transfer functions, and recursive least squares methods for identifying the Markov parameters. Chapter 3 extends the classical ARX-based SPC framework to include ARMAX models, presenting the necessary assumptions, modified data and predictor equations, and the corresponding numerical implementation. Subsequently, chapter 4 compares the ARX- and ARMAX-based SPC algorithms through numerical simulations, while chapter 5 evaluates their real-time performance in laboratory experiments on an inertia-spring-damper system.

Chapter 2

Theoretical Background

This chapter presents the theoretical foundations of data-driven control, with a focus on the formulation and derivation of Subspace Predictive Control (SPC). By exploring the underlying models, methods, and algorithms, we develop a comprehensive understanding of SPC that enables the extension of the current algorithm with the integration of alternative model structures.

The SPC framework, summarised in Algorithm 1, consists of three main components: (i) a system identification step to find the Markov Parameters via least squares estimation, (ii) the formulation of the data equations, and (iii) the application of Model Predictive Control (MPC). These steps enable the design of an optimal control scheme based on input-output data without the need of an explicitly known model.

To derive and understand the SPC algorithm, we first introduce the state-space modelling framework in section 2-1. Since noise is inherent in real-world data, we then discuss several polynomial models that explicitly account for both noise and system dynamics in section 2-2.

Next, we address the identification of these models from measured data. This leads to the field of System Identification, covered in section 2-3, which provides an overview of techniques for estimating mathematical models of dynamic systems from input-output measurements, with an emphasis on subspace identification methods.

With these modelling and identification tools, we derive the SPC algorithm using AutoRegressive models with eXogenous inputs (ARX). The derivation, presented in section 2-4, uses a least squares formulation to estimate the Markov parameters. Subsequently, the relationship between the Markov Parameters, polynomial models, and their corresponding transfer functions is established in section 2-5

Finally, section 2-6 discusses the integration of Model Predictive Control (MPC) within SPC, while section 2-7 examines computational strategies for efficiently solving least squares problems. Building on these foundations, a detailed research plan for the remainder of the thesis is presented in section 2-8.

2-1 Introduction to State-Space Modelling

The introduction of the parametric state-space model by Kalman in 1960, together with optimal control, marked the beginning of modern control theory, also known as model-based control (MBC). In MBC applications, the first step is to model the system from first principles or to identify it directly from data, after which a controller is designed based on the obtained model. Since controller performance depends on the models ability to accurately represent the real system, modelling and identification are fundamental to MBC for obtaining a reliable state-space representation [17].

This state-space representation provides a mathematical model for describing continuous dynamical systems. The most general state-space model is given by [24]:

$$\begin{aligned} \dot{x}(t) &= f(t, x(t), u(t), w(t)), \\ y(t) &= h(t, x(t), u(t), v(t)). \end{aligned} \tag{2-1}$$

For the Subspace Predictive Control (SPC) framework, a linear, discrete-time model is considered, which may include slowly time-varying dynamics. However, to develop the theory and simplify notation, a linear time-invariant (LTI) system is assumed. In addition, process and measurement noise are typically present in practical applications. Therefore, the following discrete, noisy LTI model is considered [46]:

$$\begin{aligned} x_{k+1} &= Ax_k + Bu_k + w_k, \\ y_k &= Cx_k + Du_k + v_k. \end{aligned} \tag{2-2}$$

Here $x_k \in \mathbb{R}^{n\times 1}$ is the state vector, $u_k \in \mathbb{R}^{m\times 1}$ the input vector, $w_k \in \mathbb{R}^{n\times 1}$ the process noise with zero mean and covariance $Q \geq 0$, $y_k \in \mathbb{R}^{l\times 1}$ the output vector, and $v_k \in \mathbb{R}^{l\times 1}$ the measurement noise with zero mean and covariance $R > 0$. The matrices $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times m}$, $C \in \mathbb{R}^{l\times n}$, and $D \in \mathbb{R}^{l\times m}$ are the state, input, output, and feedthrough matrices, respectively. The stochastic properties are summarised as [23]:

$$\mathbb{E}(w_k) = 0, \quad \mathbb{E}(v_k) = 0, \quad \forall k, \quad \text{(Zero-Mean)} \tag{2-3}$$

$$\mathbb{E}(w_k w_s^T) = 0, \quad \mathbb{E}(v_k v_s^T) = 0, \quad \text{if } k \neq s, \quad \text{(Independent random variables)} \tag{2-4}$$

$$\mathbb{E}(w_k w_k^T) = Q, \quad \mathbb{E}(v_k v_k^T) = R, \quad \forall k, \quad \text{(Covariance)} \tag{2-5}$$

$$\mathbb{E}(w_k v_k^T) = S, \quad \mathbb{E}(v_k w_k^T) = S^T, \quad \forall k. \quad \text{(Cross-Covariance)} \tag{2-6}$$

The noisy LTI model in Equation 2-2 can be expressed in innovation form by introducing the Kalman Gain $K \in \mathbb{R}^{n\times l}$ [51]:

$$\begin{aligned} x_{k+1} &= Ax_k + Bu_k + Ke_k, \\ y_k &= Cx_k + Du_k + e_k, \end{aligned} \tag{2-7}$$

where e_k represents the innovation, which is the component of y_k that cannot be predicted from past data. As shown in [28], e_k is a white sequence with covariance $\Sigma = CPC^T + R$, where P is the unique symmetric positive-definite solution to the Algebraic Riccati Equation (ARE):

$$P = APA^T + Q - \left(APC^T + S\right)\left(CPC^T + R\right)^{-1}\left(APC^T + S\right)^T. \tag{2-8}$$

The Kalman gain K can then be computed as:

$$K = \left(APC^T + S\right)\left(CPC^T + R\right)^{-1} = \left(APC^T + S\right)\Sigma^{-1}. \tag{2-9}$$

An equivalent predictor form can be obtained by substituting $e_k = y_k - Cx_k - Du_k$ into the state equation [46]:

$$\begin{aligned} x_{k+1} &= \tilde{A}x_k + \tilde{B}u_k + Ky_k, \\ y_k &= Cx_k + Du_k + e_k, \end{aligned} \tag{2-10}$$

where $\tilde{A} = A - KC$ and $\tilde{B} = B - KD$. Any LTI state-space model can be written in this predictor form, where K denotes the steady-state Kalman gain. Provided that the pair (A, C) is detectable and $(A, Q^{1/2})$ is stabilisable, the resulting predictor dynamics $\tilde{A} = A - KC$ are asymptotically stable [1, 7, 40]. This representation forms the basis for deriving the Subspace Predictive Control algorithm in section 2-4 for our application of a discrete-time, noisy, and slowly time-varying LTI system.

While the state-space representation provides a compact and physically interpretable description of system dynamics, data-driven methods typically operate directly on measured inputs and outputs. To relate these measurable quantities, the state-space equations in Equation 2-2 can be rewritten as difference equations involving past inputs, outputs, and disturbances. This leads to polynomial model structures, which express the system dynamics as functions of the backward shift operator q^{-1}.

2-2 Introduction to Polynomial Model Structures

This section introduces a set of polynomial models commonly used in system identification and predictive control: the AR (AutoRegressive), OE (Output Error), ARX (AutoRegressive with exogenous terms), ARMAX (AutoRegressive-moving average with exogenous terms), and Box-Jenkins model. All these models are simplifications of the general linear model [28]:

$$A(q)y(k) = \frac{B(q)}{F(q)}u(k) + \frac{C(q)}{D(q)}e(k), \tag{2-11}$$

where $y(k)$ is the output, $u(k)$ the input, and $e(k)$ a disturbance. The polynomials $A(q)$, $B(q)$, $C(q)$, $D(q)$, and $F(q)$ are defined in terms of the backward shift operator q^{-1}, defined as $q^{-p}\psi(k) = \psi(k-p)$ with ψ a placeholder, and take the form:

$$A(q) = 1 + a_1q^{-1} + a_2q^{-2} + \ldots + a_{k_a}q^{-k_a}, \tag{2-12}$$

$$B(q) = b_0 + b_1q^{-1} + b_2q^{-2} + \ldots + b_{k_b}q^{-k_b}, \tag{2-13}$$

$$C(q) = 1 + c_1q^{-1} + c_2q^{-2} + \ldots + c_{k_c}q^{-k_c}, \tag{2-14}$$

$$D(q) = 1 + d_1q^{-1} + d_2q^{-2} + \ldots + d_{k_d}q^{-k_d}, \tag{2-15}$$

$$F(q) = 1 + f_1q^{-1} + f_2q^{-2} + \ldots + f_{k_f}q^{-k_f}. \tag{2-16}$$

The parameters k_a, k_b, k_c, k_d and k_f denote the polynomial orders. By selecting certain polynomials as unity or zero, the following model structures are obtained:

$$\begin{aligned}
&\text{Box-Jenkins} \quad & y(k) &= \frac{B(q)}{F(q)}u(k) + \frac{C(q)}{D(q)}e(k), & \text{(2-17a)}\\
&\text{ARMAX} \quad & y(k) &= \frac{B(q)}{A(q)}u(k) + \frac{C(q)}{A(q)}e(k), & \text{(2-17b)}\\
&\text{ARX} \quad & y(k) &= \frac{B(q)}{A(q)}u(k) + \frac{1}{A(q)}e(k), & \text{(2-17c)}\\
&\text{OE} \quad & y(k) &= \frac{B(q)}{F(q)}u(k) + e(k), & \text{(2-17d)}\\
&\text{AR} \quad & y(k) &= \frac{1}{A(q)}e(k). & \text{(2-17e)}
\end{aligned}$$

Each of these models can also be represented in the time domain. Among them, the ARMAX and simpler models have particularly interpretable time-domain forms. The ARMAX model, for instance, expresses the current output $y(k)$ as a linear combination of previous outputs, current and past inputs, and current and past disturbances:

$$\begin{aligned}
&\underbrace{y(k) + a_1 y(k-1) + \ldots + a_{k_a} y(k-k_a)}_{\text{Autoregressive (AR)}}\\
&= \underbrace{b_0 u(k) + b_1 u(k-1) + \ldots + b_{k_b} u(k-k_b)}_{\text{eXogeneous (X)}} + \underbrace{e(k) + c_1 e(k-1) + \ldots c_{k_c} e(k-k_c)}_{\text{Moving Average (MA)}}. \qquad \text{(2-18)}
\end{aligned}$$

Identifying an ARMAX model from data is generally more challenging than identifying an ARX model due to the presence of past noise terms $e(k-i)$ for $i = 1 \ldots k_c$, which are not directly measurable. However, if these past disturbances are known or can be estimated, the identification of an ARMAX model is closely related to the identification of an ARX model.

Within the Subspace Preditive Control (SPC) framework, the ARX model is frequently used, since a sufficiently high-order ARX model is capable of approximating any linear system arbitrarily well [28, 336], [30]. Therefore, the identification of an ARX model is treated in depth in section 2-4 and 2-5, while the ARMAX extension is covered in chapter 3. Once the model parameters are identified, the models frequency response or Bode diagram provides insight into its dynamic and noise characteristics.

2-3 Introduction to System Identification

To apply the polynomial models introduced in section 2-2 within the Subspace Predictive Control (SPC) framework, it is essential to identify their parametrisations from data. Numerous identification methods exist in the literature that can broadly be classified into three categories: White-Box, Gray-Box, and Black-Box models. These classifications differ in the extent to which physical insight is incorporated into the modelling process [31]. Specifically:

- White-Box models rely entirely on first-principles and physical laws.

- Gray-Box models combine partial physical knowledge with data-driven techniques.
- Black-Box models are purely data-driven and do not require any prior physical understanding of the system.

In this thesis, we focus on Black-Box modelling, as data-driven methods are frequently applied to higher-order systems where deriving physical models may be infeasible or impractical. The goal of system identification is to obtain a model that accurately captures the behaviour of the system based on observed input-output data. Common identification approaches include Prediction Error Methods (PEM), Instrumental Variable Methods (IVM), and Subspace Identification Methods (SIM) [29, 51]. The idea of each method is briefly discussed below.

Prediction Error Method (PEM)

PEM treats the model as a predictor of the system output. Given a parameter vector θ, the one-step-ahead prediction error is defined as:

$$e_k(\theta) = y_k - \hat{y}_k(\theta). \tag{2-19}$$

To assess the overall model performance on the dataset of size N, a cost function is defined as:

$$V_N(\theta) = \sum_{k=1}^{N} l(e_k(\theta)), \tag{2-20}$$

where $l(\cdot)$ is a norm function, often the squared error. The optimal parameter estimate $\hat{\theta}_N$ for this dataset minimises the cost function such that:

$$\hat{\theta}_N = \arg\min_{\theta} V_N(\theta). \tag{2-21}$$

In the case of a quadratic cost function, the solution is commonly found through gradient-based optimisation methods.

Instrumental Variable Methods (IVM)

IVM ensures that prediction errors are uncorrelated with the available past information. That is, a desirable model parameter θ satisfies:

$$\sum_{k=1}^{N} e_k(\theta)\zeta_k(\theta) = 0, \tag{2-22}$$

where ζ_k is called an instrumental variable that is not correlated with noise. ζ_k can be chosen freely such that the above equation holds and is typically constructed from past input and output data. This method is particularly useful in cases with measurement noise or when the regressors are correlated with the disturbances.

Subspace Identification Methods (SIM)

Subspace Identification Methods identify state-space models using structured data matrices created from input-output data and solving a least-squares problem. Through these methods, the original system matrices A, B, C, D, and K can be estimated. These approaches often rely on linear algebraic operations, such as singular value decomposition (SVD) and QR factorisation, rather than iterative optimisation. Consequently, SIMs are often more computationally efficient compared to PEM or maximum likelihood methods. Subspace methods are also especially attractive for multivariable systems and large datasets, as they scale well and yield consistent estimates without requiring an initial guess for model parameters [21]. An overview of SIM, where methods such as PBSID [5], MOESP [53], and N4SID [47, 48] are covered, are summarised in [50].

Recent developments in system identification have also explored the use of machine learning techniques for Black-Box modelling. These include neural networks and Gaussian processes that can approximate nonlinear dynamics [41]. Furthermore, hybrid approaches such as physics-informed machine learning or physics-informed DeePC [59] have gained interest, where knowledge about the system is partially integrated.

Having introduced the principles of subspace identification, the next step is to examine how these methods can be integrated directly into the Subspace Predictive Control (SPC) framework.

2-4 Least Squares Formulation for the Identification of ARX Models

Subspace System Identification (SIM) was developed in the late 1980s and proved to be a highly efficient method for modelling high-order systems with multiple inputs and outputs [11]. Traditionally, system identification and control design were treated as separate problems. In 1990, Favoreel [11] introduced Subspace Predictive Control (SPC), a method that integrates the identification and predictive control design of linear systems into a single procedure.

The main result was that the steps of system identification and controller design could be replaced by a QR decomposition followed by a singular value decomposition (SVD) of matrices containing input and output data of the unknown linear time-invariant (LTI) system, with the assumption that the system could be written in innovation form, as introduced in Equation 2-7. This approach demonstrated that subspace identification and model predictive control could be combined into a numerically robust identification and control algorithm.

A fundamental assumption in the original SPC formulation is that the input u_k is independent of the disturbance affecting the output, implying the absence of feedback from y_k to u_k [64]. In practice, however, the data is collected under closed-loop conditions, as new input-output measurements are incorporated into the next iterations. As shown in [51], this violates the independence assumption, leading to biased least-squares estimates and reduced model accuracy, particularly at low and middle frequencies [64]. This bias affects the controllers performance, particularly when the predictor is used within a model predictive control (MPC) framework [6]. To mitigate this issue, an adapted structure for Closed-Loop SPC algorithm is discussed next.

Adapted SPC structure for closed-loop bias

This section derives the Subspace Predictive Control (SPC) algorithm, following the approach outlined in [46]. A thorough understanding of the underlying linear algebra and derivation steps is essential for the development and application of the SPC algorithm.

We begin with the predictor form introduced in Equation 2-10 at time step k:

$$\begin{aligned} x_{k+1} &= \tilde{A}x_k + \tilde{B}u_k + Ky_k, \\ y_k &= Cx_k + Du_k + e_k. \end{aligned} \tag{2-23}$$

A similar expression holds for the next time step $k+1$:

$$\begin{aligned} x_{k+2} &= \tilde{A}x_{k+1} + \tilde{B}u_{k+1} + Ky_{k+1}, \\ y_{k+1} &= Cx_{k+1} + Du_{k+1} + e_{k+1}. \end{aligned} \tag{2-24}$$

Substituting x_{k+1} of Equation 2-23 into Equation 2-24 gives:

$$\begin{aligned} x_{k+2} &= \tilde{A}\left(\tilde{A}x_k + \tilde{B}u_k + Ky_k\right) + \tilde{B}u_{k+1} + Ky_{k+1} \\ &= \tilde{A}^2x_k + \tilde{A}\tilde{B}u_k + \tilde{A}Ky_k + \tilde{B}u_{k+1} + Ky_{k+1} \\ &= \tilde{A}^2x_k + \left[\begin{array}{cc:cc} \tilde{A}K & K & \tilde{A}\tilde{B} & \tilde{B} \end{array}\right] \left[\begin{array}{c} y_k \\ y_{k+1} \\ \hdashline u_k \\ u_{k+1} \end{array}\right], \end{aligned} \tag{2-25}$$

$$\begin{aligned} y_{k+1} &= C\left(\tilde{A}x_k + \tilde{B}u_k + Ky_k\right) + Du_{k+1} + e_{k+1} \\ &= C\tilde{A}x_k + C\tilde{B}u_k + CKy_k + Du_{k+1} + e_{k+1} \\ &= C\tilde{A}x_k + \left[\begin{array}{c:c} CK & C\tilde{B} \end{array}\right] \left[\begin{array}{c} y_k \\ \hdashline u_k \end{array}\right] + Du_{k+1} + e_{k+1}. \end{aligned} \tag{2-26}$$

Following the same approach, these relations can be extended to any general time step $k+p$.

To simplify the notation, define the extended controllability matrix $\mathcal{K} \in \mathbb{R}^{n \times lp+mp}$ as:

$$\mathcal{K} = \left[\begin{array}{cccc:cccc} \tilde{A}^{p-1}K & \cdots & \tilde{A}K & K & \tilde{A}^{p-1}\tilde{B} & \cdots & \tilde{A}\tilde{B} & \tilde{B} \end{array}\right], \tag{2-27}$$

and a stacked data vector $z_k^{(p)} \in \mathbb{R}^{lp+mp \times 1}$:

$$z_k^{(p)} = \left[\begin{array}{cccc:cccc} y_k^T & y_{k+1}^T & \cdots & y_{k+p-1}^T & u_k^T & u_{k+1}^T & \cdots & u_{k+p-1}^T \end{array}\right]^T. \tag{2-28}$$

Then, for any $p \in \mathbb{N}$, x_{k+p} and y_{k+p} can be written as:

$$x_{k+p} = \tilde{A}^p x_k + \mathcal{K}z_k^{(p)}, \tag{2-29}$$

$$y_{k+p} = C\tilde{A}^p x_k + C\mathcal{K}z_k^{(p)} + Du_{k+p} + e_{k+p}. \tag{2-30}$$

Here $C\mathcal{K} \in \mathbb{R}^{l \times lp+mp}$ is given by:

$$C\mathcal{K} = \left[\begin{array}{cccc:cccc} C\tilde{A}^{p-1}K & \cdots & C\tilde{A}K & CK & C\tilde{A}^{p-1}\tilde{B} & \cdots & C\tilde{A}\tilde{B} & C\tilde{B} \end{array}\right]. \tag{2-31}$$

If $\tilde{A}$ is stable, and therefore has all its eigenvalues inside the unit circle, then $||\tilde{A}^p||_2 \approx 0$ for sufficiently large p. This means that the past horizon is chosen large enough to neglect the effect of the initial state, and the term can therefore be omitted in both equations.

For a dataset containing $N+p$ input-output data samples, the expressions for the output can be repeated to obtain y_{k+p} up to $y_{k+p+N-1}$.

$$\begin{aligned} y_{k+p} &= C\tilde{A}^p x_k + C\mathcal{K}z_k^{(p)} + Du_{k+p} + e_{k+p}, \\ y_{k+p+1} &= C\tilde{A}^p x_{k+1} + C\mathcal{K}z_{k+1}^{(p)} + Du_{k+p+1} + e_{k+p+1}, \\ &\vdots \\ y_{k+p+N-1} &= C\tilde{A}^p x_{k+N-1} + C\mathcal{K}z_{k+N-1}^{(p)} + Du_{k+p+N-1} + e_{k+p+N-1}. \end{aligned} \tag{2-32}$$

Each of these terms are computed similarly to Equation 2-30, where the first term $C\tilde{A}^p x_k$ is neglected. These expressions are then collected into a "row vector" of size $l \times N$ in the form $Y_f = \begin{bmatrix} y_{k+p} & y_{k+p+1} & \cdots & y_{k+p+N-1} \end{bmatrix}$, so $Y_f \in \mathbb{R}^{l\times N}$ can be written as:

$$Y_f = C\mathcal{K}Z_p + DU_f + E_f. \tag{2-33}$$

In this equation Z_p, U_f and E_f are defined similarly through horizontal stacking:

$$Z_p = \begin{bmatrix} z_k^{(p)} & z_{k+1}^{(p)} & \cdots & z_{k+N-1}^{(p)} \end{bmatrix} \in \mathbb{R}^{lp+mp\times N}, \tag{2-34}$$

$$U_f = \begin{bmatrix} u_{k+p} & u_{k+p+1} & \cdots & u_{k+p+N-1} \end{bmatrix} \in \mathbb{R}^{m\times N}, \tag{2-35}$$

$$E_f = \begin{bmatrix} e_{k+p} & e_{k+p+1} & \cdots & e_{k+p+N-1} \end{bmatrix} \in \mathbb{R}^{l\times N}. \tag{2-36}$$

Since the innovation sequence e_k is a zero-mean white noise, the expression for Y_f can be used to estimate the terms $C\mathcal{K}$ and D in a least-square sense [46]:

$$\widehat{[C\mathcal{K}, D]} = \underset{[C\mathcal{K},D]}{\arg\min} \left\| Y_f - [C\mathcal{K}, D] \begin{pmatrix} Z_p \\ U_f \end{pmatrix} \right\|_F^2. \tag{2-37}$$

Defining $\Phi_k = \begin{pmatrix} Z_p \\ U_f \end{pmatrix}$, then in case $\Phi_k \Phi_K^T$ is invertible, a unique solution to the least-squares problem in Equation 2-37 exists, and is given by:

$$\widehat{[C\mathcal{K}, D]} = Y_f \Phi_k^T (\Phi_k \Phi_k^T)^{-1}, \tag{2-38}$$

where $\Phi_k^T(\Phi_k\Phi_k^T)^{-1} = \Phi_k^\dagger$ is the Moore-Penrose Inverse, also known as the pseudoinverse denoted by $(\cdot)^\dagger$. There are various aspects that require attention when solving Equation 2-37, which are discussed in more detail in section 2-7.

Note that the solution of the Least Squares problems contains the Markov parameters in $C\mathcal{K}$, which can now be used for control purposes through an MPC scheme. The application of these terms for controller synthesis and the computation of an optimal control law within a Model Predictive Control (MPC) framework are discussed in detail in section 2-6. It is now first shown how the obtained Markov Parameters are related to the ARX model from section 2-2.

2-5 Relation Between Transfer Functions and ARX Models

While the state-space formulation provides a compact description of system dynamics, it might not yet be clear how the estimated Markov parameters obtained in the least-squares problem relate to the ARX transfer function introduced in Equation 2-17c. These parameters, which correspond to the impulse response of the system, form the bridge between the state-space and transfer function representations.

To illustrate this connection, consider Equation 2-32, where the output at time $k+p$ is given by:

$$y_{k+p} = C\mathcal{K}z_k^{(p)} + Du_{k+p} + e_{k+p}. \tag{2-39}$$

Expanding the right-hand side by substituting the definitions for $C\mathcal{K}$ and $z_k^{(p)}$ yields:

$$\begin{aligned} y_{k+p} =& C\tilde{A}^{p-1}Ky_k + \ldots + C\tilde{A}Ky_{k+p-2} + CKy_{k+p-1} \\ &+ C\tilde{A}^{p-1}\tilde{B}u_k + \ldots + C\tilde{A}\tilde{B}u_{k+p-2} + C\tilde{B}u_{k+p-1} + Du_{k+p} + e_{k+p}. \end{aligned} \tag{2-40}$$

The ARX model from Equation 2-17c can similarly be expressed in the time domain as:

$$A(q)y(k) = B(q)u(k) + e(k), \tag{2-41}$$

$$y_{k+p} + a_1y_{k+p-1} + \ldots + a_py_k = b_0u_{k+p} + b_1u_{k+p-1} + \ldots + b_pu_k + e_k. \tag{2-42}$$

Comparing the two expressions shows that the ARX coefficients a_i and b_i can be parameterised directly in terms of the Markov parameters as:

$$a_i = -C\tilde{A}^{i-1}K, \qquad \text{for } i = 1\ldots p, \tag{2-43}$$

$$b_i = C\tilde{A}^{i-1}\tilde{B}, \qquad \text{for } i = 1\ldots p, \tag{2-44}$$

$$b_0 = D. \tag{2-45}$$

Although the method allows the direct feedthrough term D, well-posed problems might occur if this term is included [46]. Instead of comparing individual coefficients, the following subsections provide a more general and theoretical derivation using the transfer functions of the innovation and predictor forms, leading to an equivalent parametrisation.

2-5-1 Transfer Function of the Innovation Form

A general linear time-invariant (LTI) system can be expressed in transfer-function form as [29]:

$$y_k = G(q)u_k + H(q)e_k, \tag{2-46}$$

where e_k is a white noise sequence with zero mean independent of u_k. Here, $G(q)$ represents the deterministic dynamics from input to output, while $H(q)$ models the stochastic dynamics due to noise [51]. Depending on the choice of $H(q)$, several of which were discussed in Section 2-2, it can describe the disturbance with arbitrary spectrum.

The one-step-ahead predictor $\hat{y}_k$, which corresponds to the conditional mean of y_k given past data, of the general model in Equation 2-46 is given by [29]:

$$\begin{aligned} \hat{y}_{k|\Theta} &= G(q)u_k + \big(I - H^{-1}(q)\big)\big(y_k - G(q)u_k\big) && \text{(2-47)} \\ &= H^{-1}(q)G(q)u_k + \big(I - H^{-1}(q)\big)y_k. && \text{(2-48)} \end{aligned}$$

This result for the one-step-ahead predictor can be obtained as follows. Rewriting Equation 2-46 gives:

$$y_k = G(q)u_k + (H(q) - I)e_k + e_k. \tag{2-49}$$

Since e_k is a zero-mean innovation, its conditional expectation is zero, leading to:

$$\hat{y}_{k|\Theta} = \mathbb{E}(y_{k|\Theta}) = G(q)u_k + (H(q) - I)e_k. \tag{2-50}$$

Assuming $H(q)$ is invertible, we can express e_k in terms of known quantities:

$$\begin{aligned} H(q)e_k &= y_k - G(q)u_k, \\ e_k &= H^{-1}(y_k - G(q)u_k). \end{aligned} \tag{2-51}$$

Substituting this into Equation 2-50 yields:

$$\begin{aligned} \hat{y}_{k|\Theta} &= G(q)u_k + (H(q) - I)H^{-1}(q)(y_k - G(q)u_k) && (2\text{-}52) \\ &= G(q)u_k + (I - H^{-1}(q))(y_k - G(q)u_k) && (2\text{-}53) \\ &= H^{-1}(q)G(q)u_k + (I - H^{-1}(q))y_k, && (2\text{-}54) \end{aligned}$$

which matches the predictor form in Equation 2-48.

The transfer functions $H(q)$ and $G(q)$ for the innovation form are parametrised by:

$$\begin{aligned} H(q) &= I + C(qI - A)^{-1}K, && (2\text{-}55) \\ G(q) &= D + C(qI - A)^{-1}B. && (2\text{-}56) \end{aligned}$$

For the predictor to be stable, $H(q)$ must have no zeros on or outside the unit circle, ensuring that its inverse $H^{-1}(q)$ is also stable.

Derivation of the Transfer Functions via the z-Transform

The transfer functions $H(q)$ and $G(q)$ can also be derived directly from the state-space innovation form in Equation 2-7 by taking the z-transform[1] under zero initial conditions:

$$\begin{aligned} qX_q &= AX_q + BU_q + KE_q, \\ (qI - A)X_q &= BU_q + KE_q, \\ X_q &= (qI - A)^{-1}BU_q + (qI - A)^{-1}KE_q. \end{aligned} \tag{2-57}$$

Substituting the latter expression for X_q into the z-transform of y_k:

$$\begin{aligned} Y_q &= CX_q + Du_q + E_q, \\ Y_q &= C(qI - A)^{-1}BU_q + C(qI - A)^{-1}KE_q + Du_q + E_q, \\ Y_q &= \underbrace{\left(D + C(qI - A)^{-1}B\right)}_{G(q)} U_q + \underbrace{\left(I + C(qI - A)^{-1}K\right)}_{H(q)} E_q. \end{aligned} \tag{2-58}$$

These are precisely the same expressions for $H(q)$ and $G(q)$ as defined in Equation 2-55 and 2-56.

[1]In this thesis, the backward shift operator q^{-1} and the z-transform variable z^{-1} are used interchangeably, as they are algebraically equivalent.

2-5-2 Transfer Function of the Predictor Form

A similar derivation can be carried out for the one-step-ahead predictor form introduced in Equation 2-10. Taking the z-transform under zero initial conditions yields:

$$X_q = (qI - \tilde{A})^{-1}\tilde{B}U_q + (qI - \tilde{A})^{-1}KY_q, \tag{2-59}$$

$$Y_q = \left(I - C(qI - \tilde{A})^{-1}K\right)^{-1}(D + C(qI - \tilde{A})^{-1}\tilde{B})U_q + \left(I - C(qI - \tilde{A})^{-1}K\right)^{-1}E_q. \tag{2-60}$$

Applying Woodbury's Inversion Lemma, also called the matrix inversion identity [14],

$$(A + BCD)^{-1} = A^{-1} - A^{-1}B(C^{-1} + DA^{-1}B)^{-1}DA^{-1}, \tag{2-61}$$

and noting that

$$-\tilde{A} - KC = -A + KC - KC = -A, \tag{2-62}$$

gives

$$\left(I - C(qI - \tilde{A})^{-1}K\right)^{-1} = I + C(qI - \tilde{A} - KC)^{-1}K \tag{2-63}$$

$$= I + C(qI - A)^{-1}K. \tag{2-64}$$

Substituting this result into Equation 2-60 yields:

$$Y_q = \underbrace{\left(I + C(qI - A)^{-1}K\right)\left(D + C(qI - \tilde{A})^{-1}\tilde{B}\right)}_{\bar{G}(q)} U_q + \underbrace{\left(I + C(qI - A)^{-1}K\right)}_{\bar{H}(q)} E_q. \tag{2-65}$$

Thus, in the standard form,

$$y_k = \bar{G}(q)u_k + \bar{H}(q)e_k, \tag{2-66}$$

where the predictor transfer functions $\bar{G}(q)$ and $\bar{H}(q)$ are parametrised by:

$$\bar{H}(q) = \left(I + C(qI - A)^{-1}K\right) = H(q), \tag{2-67}$$

$$\bar{G}(q) = \left(I + C(qI - A)^{-1}K\right)\left(D + C(qI - \tilde{A})^{-1}\tilde{B}\right) = H(q)\tilde{G}(q). \tag{2-68}$$

The corresponding one-step-ahead predictor $\hat{y}_k$ can then be written as:

$$\hat{y}_k = \tilde{G}u_k + \tilde{H}y_k, \tag{2-69}$$

where $\tilde{G}(q) = D + C(qI - \tilde{A})^{-1}\tilde{B}$ and $\tilde{H}(q) = C(qI - \tilde{A})^{-1}K$ [46]. Since the resolvent $(qI - \tilde{A})^{-1}$ can be expanded as:

$$(qI - \tilde{A})^{-1} = \sum_{k=0}^{\infty} \frac{\tilde{A}^k}{q^{k+1}}, \tag{2-70}$$

the terms $C(qI - \tilde{A})^{-1}\tilde{B}$ and $C(qI - \tilde{A})^{-1}K$ correspond exactly to the Markov Parameters contained in $C\mathcal{K}$ in Equation 2-31.

The analysis above shows that the Markov parameters obtained through the SPC framework provide a direct link between the state-space, ARX, and transfer-function representations.

2-6 Model Predictive Control for ARX Models

With the prediction model established, the next step is to formulate a Model Predictive Controller (MPC) based on the identified Markov parameters. MPC is an optimisation-based control strategy that computes an optimal input sequence over a finite control horizon N_c by predicting the future outputs of the system over a finite prediction horizon N_p. The control objective is to minimise a cost function that penalises deviations from a reference trajectory and excessive control effort, while satisfying system constraints.

In the context of Subspace Predictive Control (SPC), the previously estimated predictor Markov parameters from the least-squares problem can be used to construct a linear output predictor based on Equation 2-32:

$$y_{k+p} = \widehat{C\mathcal{K}} z_k^{(p)} + \widehat{D} u_{k+p}. \tag{2-71}$$

Prediction Model Formulation

Let the sequences of predicted outputs $\tilde{y}_{\text{pred}}$ and future inputs $\tilde{u}$ over the prediction horizon N_p be defined as:

$$\tilde{y}_{\text{pred}} = \begin{bmatrix} y_{k+1} \\ y_{k+2} \\ \vdots \\ y_{k+N_p} \end{bmatrix}, \qquad \tilde{u} = \begin{bmatrix} u_{k+1} \\ u_{k+2} \\ \vdots \\ u_{k+N_p} \end{bmatrix}. \tag{2-72}$$

Then, the predicted output sequence $\tilde{y}_{\text{pred}} \in \mathbb{R}^{lN_p \times 1}$ can be expressed as:

$$\tilde{y}_{\text{pred}} = \tilde{\Gamma} z_k^{(p)} + \tilde{H} \tilde{u}_k + \tilde{G} \tilde{y}_{\text{pred}}. \tag{2-73}$$

Here, the matrix $\tilde{\Gamma}$ represents the effect of past known inputs and outputs, while the lower triangular block Toeplitz matrices $\tilde{H}$ and $\tilde{G}$ capture the influence of future inputs and predicted outputs, respectively. For $N_p \leq p$, these matrices are given by:

$$\tilde{\Gamma} = \begin{bmatrix}\Gamma_K & \Gamma_{\tilde{B}}\end{bmatrix}, \tag{2-74}$$

$$\text{with } \Gamma_X = \begin{bmatrix} C\tilde{A}^{p-1}X & C\tilde{A}^{p-2}X & \cdots & CX \\ & \ddots & & \vdots \\ & & C\tilde{A}^{p-1}X \quad \cdots & C\tilde{A}^{N_p-1}X \end{bmatrix}, \quad \text{for } X \in \{K, \tilde{B}\}, \tag{2-75}$$

$$\tilde{H} = \begin{bmatrix} D & & & & \\ C\tilde{B} & D & & & \\ C\tilde{A}\tilde{B} & C\tilde{B} & D & & \\ \vdots & & \ddots & \ddots & \\ C\tilde{A}^{N_p-2}\tilde{B} & C\tilde{A}^{N_p-3}\tilde{B} & \dots & C\tilde{B} & D \end{bmatrix}, \tag{2-76}$$

$$\tilde{G} = \begin{bmatrix} 0 & & & & \\ CK & 0 & & & \\ C\tilde{A}K & CK & 0 & & \\ \vdots & & \ddots & \ddots & \\ C\tilde{A}^{N_p-2}K & C\tilde{A}^{N_p-3}K & \dots & CK & 0 \end{bmatrix}. \tag{2-77}$$

For $N_p > p$, the matrix entries are capped by the maximum estimated power $\tilde{A}^{p-1}$, with higher powers replaced by $\mathbf{0}$ of matching dimensions. Specifically, the matrices $\tilde{H}$ and $\tilde{G}$ are lower triangular Toeplitz matrices whose first columns contain the following vectors $v_{\tilde{H}}$ and $v_{\tilde{G}}$, respectively:

$$v_{\tilde{H}} = \begin{bmatrix} D & C\tilde{B} & C\tilde{A}\tilde{B} & \dots & C\tilde{A}^{p-1}\tilde{B} & \mathbf{0}_{l\times m(N_p-p)} \end{bmatrix}^T, \tag{2-78}$$

$$v_{\tilde{G}} = \begin{bmatrix} 0 & CK & C\tilde{A}K & \dots & C\tilde{A}^{p-1}K & \mathbf{0}_{l\times l(N_p-p)} \end{bmatrix}^T. \tag{2-79}$$

While $\tilde{\Gamma}$

$$\tilde{\Gamma} = \begin{bmatrix}\Gamma_K & \Gamma_{\tilde{B}}\end{bmatrix},$$

$$\text{with } \Gamma_X = \begin{bmatrix} C\tilde{A}^{p-1}X & \dots & CX \\ & \ddots & \\ & & C\tilde{A}^{p-1}X \\ & 0 & \end{bmatrix}, \quad \text{for } X \in \{K, \tilde{B}\}. \tag{2-80}$$

The definition of the matrices $\tilde{\Gamma}$, $\tilde{H}$, and $\tilde{G}$ is the main difference between the original SPC algorithm of Favoreel [11] and the adapted "closed-loop" formulation of van der Veen [46]. The latter uses the predictor matrices that introduce $\tilde{y}_{\text{pred}}$ on both sides of the equation, whereas Favoreel used the extended observability matrix in place of $\tilde{\Gamma}$.

Open-Loop Predictor

The open-loop predictor is obtained by solving Equation 2-73 for $\tilde{y}_{\text{pred}}$:

$$\begin{aligned} (I - \tilde{G})\tilde{y}_{\text{pred}} &= \tilde{\Gamma} z_k^{(p)} + \tilde{H}\tilde{u}_k, \\ \tilde{y}_{\text{pred}} &= (I - \tilde{G})^{-1}\left(\tilde{\Gamma} z_k^{(p)} + \tilde{H}\tilde{u}_k\right). \end{aligned} \tag{2-81}$$

This can be expressed as:

$$\tilde{y}_{\text{pred}} = \Gamma z_k^{(p)} + H\tilde{u}_k, \tag{2-82}$$

where $\Gamma = (I - \tilde{G})^{-1}\tilde{\Gamma}$ and $H = (I - \tilde{G})^{-1}\tilde{H}$. These can be rewritten in terms of the innovation system parameters (A, B, C, D, K) [46]. Within Γ, the last columns corresponding to y and u respectively contain the block matrices:

$$\begin{bmatrix} CK & CAK & CA^2K & \dots & CA^{p-1}K \end{bmatrix}^T,$$

while the last columns related to u contains the block matrices

$$\begin{bmatrix} CB & CAB & CA^2B & \dots & CA^{p-1}B \end{bmatrix}^T.$$

MPC Optimisation Problem

The MPC optimisation problem aims to find an optimal input sequence for the control horizon N_c that minimises the deviation from a reference trajectory r_{k+i} over the prediction horizon N_p, subject to the dynamic constraints from Equation 2-73:

$$\min_{u_1 \dots u_{N_c}} \sum_{i=1}^{N_p} (\tilde{y}_{\text{pred},k+i} - r_{k+i})^T q_{y,i} (\tilde{y}_{\text{pred},k+i} - r_{k+i}) + \sum_{i=1}^{N_c} u_{k+i}^T r_{u,i} u_{k+i} + \Delta u_{k+i}^T r_{\Delta u,i} \Delta u_{k+i} \tag{2-83}$$

The MPC framework allows the inclusion of additional constraints such as actuator limits, rate limits, or safety-related output constraints [4]. The objective function in Equation 2-83 can be expressed in a quadratic form including constraints leading to the following general optimisation problem [37]:

$$\begin{aligned}
\min_{u_k} \quad & (\tilde{y}_{\text{pred}} - r)^T Q_y (\tilde{y}_{\text{pred}} - r) + u_k^T R_u u_k + \Delta u_k^T R_{\Delta u} \Delta u_k && \text{(2-84a)} \\
\text{s.t.} \quad & \tilde{y}_{\text{pred}} = \tilde{\Gamma} z_k^{(p)} + \tilde{H}\tilde{u}_k + \tilde{G}\tilde{y}_{\text{pred}}, && \text{(2-84b)} \\
& u_{k+i|k} \in \mathcal{U} \quad i = 1, \dots, N_c, && \text{(2-84c)} \\
& \Delta u_{k+i|k} \in \mathcal{U}_\Delta \quad i = 1, \dots, N_c, && \text{(2-84d)} \\
& y_{k+i|k} \in \mathcal{Y} \quad i = 1, \dots, N_p. && \text{(2-84e)}
\end{aligned}$$

Here, $\mathcal{U}$, $\mathcal{U}_\Delta$, and $\mathcal{Y}$ denote the sets for the control input, input rate, and output, respectively. These sets are typically defined as lower and upper bounds:

$$\begin{aligned}
\mathcal{U} &= \{u \mid u_{\min} \le u \le u_{\max}\}, \\
\mathcal{U}_\Delta &= \{\Delta u \mid \Delta u_{\min} \le \Delta u \le \Delta u_{\max}\}, \\
\mathcal{Y} &= \{y \mid y_{\min} \le y \le y_{\max}\}.
\end{aligned} \tag{2-85}$$

In this formulation, Q_y is a semi-positive definite matrix, and R_u and $R_{\Delta u}$ are positive definite weighting matrices [37]. The desired reference trajectory is denoted by r, and $\tilde{y}_{\text{pred}}$, defined in Equation 2-81, is a function of the input sequence u_k. Although u_k is defined over the prediction horizon N_p, optimisation is only performed over the control horizon N_c, while the final $N_p - N_c$ inputs are held constant to reduce computational cost.

2-6-1 Unconstrained Optimisation Solution

When the weighting matrices Q_y, R_u, and $R_{\Delta u}$ are chosen appropriately, it may be sufficient to solve the MPC problem without constraints. In this case, the optimisation problem becomes an unconstrained quadratic optimisation problem, which can be solved analytically over the entire prediction horizon N_p by taking the derivative of Equation 2-84 with respect to u_k and equating it to zero.

Let the following terms be defined:

$$M = (I - \tilde{G})^{-1}\tilde{H}, \tag{2-86}$$

$$F = (I - \tilde{G})^{-1}\tilde{\Gamma}z_k^{(p)}, \tag{2-87}$$

then, the optimal control sequence for the unconstrained problem is computed as [54]:

$$u_k = -\left(M^T Q_y M + R_u + A^T R_{\Delta u} A\right)^{-1}\left(M^T Q_y (F - r) + A^T R_{\Delta u} b\right). \tag{2-88}$$

Although the vector u_k contains the optimal control inputs over the entire control horizon N_c, only the first control input is applied to the system at each time step. After this, a new input-output measurement is collected, and the optimisation is repeated. This receding-horizon approach allows controller feedback and enables the controller to react to changes in system behaviour and disturbances.

In the infinite-horizon setting, it has been shown that Subspace Predictive Control is equivalent to Linear Quadratic Gaussian (LQG) control [7]. When there is a steady-state offset, an offset-free MPC formulation can be implemented using a rate-based control strategy, as described in [55, 54].

Based on the formulations in section 2-4 and section 2-6, the complete Subspace Predictive Control (SPC) algorithm for (V)ARX models is summarised in Algorithm 1.

Algorithm 1 Subspace Predictive Control Algorithm for ARX models

```
1: Estimate the Markov parameters $C\mathcal{K}$ and $D$ in Equation 2-38 by solving the LS problem
   from Equation 2-37.
2: Construct data matrices $\tilde{\Gamma}$, $\tilde{H}$, and $\tilde{G}$ from Equation 2-75, 2-76, and 2-77, respectively.
3: Solve the open-loop predictor for $\tilde{y}_{\text{pred}}$ given by Equation 2-81.
4: Construct and solve the quadratic problem in Equation 2-84 to obtain the optimal control
   sequence $u_k$.
5: Implement the first optimal control input $u_k(1 : m)$.
6: Update the matrices $Y_f$, $Z_p$, and $U_f$, found in Equation 2-33, 2-34, and 2-35, with the
   new input-output pair.
7: Repeat step 1-6
```

2-7 Computation of the Least Squares Solution for ARX models

Now that the SPC algorithm for ARX models is known, we return to the computation of the solution to the least-squares problem introduced in Equation 2-37. This section discusses the computational aspects and various least-squares methods, including standard LS, batch-wise LS (BLS), and recursive LS (RLS) methods.

Standard Least Squares Formulation

The standard LS solution for estimating the parameters $[\widehat{C\mathcal{K}, D}]$ is given by:

$$[\widehat{C\mathcal{K}, D}] = Y_f \Phi_k^T (\Phi_k \Phi_k^T)^+, \tag{2-89}$$

where a unique solution exists if $\Phi_k \Phi_k^T$ is full rank. To ensure full rank, N must be at least as large as the number of rows in Φ_k. According to Favoreel [11], the number of columns N is typically at least 100 times larger than the number of block rows in Φ_k. Omitting the direct feedthrough term D, the number of rows is related to the past window parameter p with a dimension of $2p(m+l)$. Since this matrix contains the inputs in the last pm rows, the input sequence U must be persistently exciting of order pm.

Persistency of Excitation is defined as follows: a sequence $\{u_k\}_{k=i}^{i+N+p-2}$ is persistently exciting of order p if the block-Hankel matrix $\mathcal{H}_{i,p,N}(u_k)$ has full row rank [60]. Intuitively, this means that the input signal is diverse and excites the system in different directions.

The selection of the past window parameter p involves a bias-variance trade-off. A larger p reduces bias by capturing longer dynamics but increases variance due to a greater number of estimated parameters [32].

Numerical Conditioning and Stability

When Φ_k is rank deficient, i.e. $\text{rank}(\Phi_k) < 2p(m+l)$, the least-squares problem is ill-posed, meaning that a unique solution does not exist. In contrast, when Φ_k is of full rank but has singular values spanning several orders of magnitude, the problem is ill-conditioned. The condition number of Φ_k is defined as:

$$\text{cond}(\Phi_k) = \frac{\sigma_{\max}}{\sigma_{\min}}, \tag{2-90}$$

where $\sigma_{\max}$ and $\sigma_{\min}$ are the largest and smallest singular values of Φ_k, respectively. Intuitively, a large condition number implies that small errors or noise in the data can lead to large deviations in the estimated parameters.

To handle ill-conditioned cases, several strategies are available:

- QR factorisation provides improved numerical stability compared to solving the normal equations directly, since it avoids forming $(\Phi_k^T \Phi_k)$ whose condition number is squared relative to Φ_k. Orthogonal transformations used in QR decomposition preserve numerical precision and can be computed efficiently via back substitution. [57]

- Truncated SVD removes small singular values below a certain threshold, to filter out the contribution of smaller singular values to the solution [15].
- Regularisation introduces a penalty term in the objective function to influence the structure or robustness of the solution. The effect depends on the form of the penalty term.

Regularisation Techniques

For the latter, common regularisation terms include [32]:

- The ℓ_1 norm, also known as least absolute shrinkage and selection operator (LASSO), promotes a sparse solution where coefficients become 0. This leads to a convex optimisation problem that encourages simpler, more interpretable models. [43, 25]
- The ℓ_2 norm, also known as Tikhonov regularisation, penalises terms with higher magnitude to prevent overfitting. While, it introduces bias, it reduces the variance and mean square error of the solution [32]. The introduction of this term still allows an analytical solution given by $\widehat{[C\mathcal{K}, D]} = Y_f \Phi_k^T (\Phi_k \Phi_k^T + \gamma I_p)^+$.
- Elastic net, is a linear combination of ℓ_1 and ℓ_2 regularisation. It is especially useful when a sparse solution is desired, while the data matrix is correlated. [42]

These regularisation strategies correspond to the following optimisation problems:

$$\widehat{[C\mathcal{K}, D]} = \underset{[C\mathcal{K},D]}{\arg\min} \left\| Y_f - [C\mathcal{K}, D] \begin{pmatrix} Z_p \\ U_f \end{pmatrix} \right\|_2^2 + \gamma \left\| \theta \right\|_1, \tag{2-91}$$

$$\widehat{[C\mathcal{K}, D]} = \underset{[C\mathcal{K},D]}{\arg\min} \left\| Y_f - [C\mathcal{K}, D] \begin{pmatrix} Z_p \\ U_f \end{pmatrix} \right\|_2^2 + \gamma \left\| \theta \right\|_2^2, \tag{2-92}$$

$$\widehat{[C\mathcal{K}, D]} = \underset{[C\mathcal{K},D]}{\arg\min} \left\| Y_f - [C\mathcal{K}, D] \begin{pmatrix} Z_p \\ U_f \end{pmatrix} \right\|_2^2 + \gamma\alpha \left\| \theta \right\|_1 + \gamma(1-\alpha) \left\| \theta \right\|_2^2. \tag{2-93}$$

More advanced regularisation techniques exist as well, which are often referred to as kernel design in literature. These methods embed prior knowledge about the system dynamics in the estimator [22].

One notable example is the Diagonal/Correlated (DC) kernel, which enforces exponential decay of higher-order Markov parameters [33]. This assumption is consistent with the SPC framework, which assumes $|\tilde{A}^p|^2 \approx 0$ for sufficiently large p, implying that the terms $C\tilde{A}^i K$ and $C\tilde{A}^i B$ decay exponentially with i. The DC kernel captures this property by introducing correlations among neighbouring parameters that reflect the underlying system dynamics through multiplication of $\tilde{A}$.

Computational Efficiency and Online Implementation

Besides estimation accuracy, computational efficiency is crucial for real-time applications. The computational load depends heavily on how the data update step (Step 6 in Algorithm 1) is handled. In the standard LS approach, all data from the start up to the current time step are used, leading to large matrix operations that become impractical in real time.

A more feasible approach is the batch-wise LS (BLS) method, which shifts the data window N along time. Only the most recent $N+p$ samples are used to construct Φ_k and Y_f, reducing matrix size and improving adaptability to time-varying systems by removing outdated data.

Recursive Least Squares (RLS)

While BLS mitigates memory growth and enhances adaptability, it still requires solving a large LS problem at each update. This motivates the Recursive Least Squares (RLS) algorithm for real-time implementation. The RLS algorithm updates the parameter estimate with a rank-one update as new data arrive, using the previous estimate $[C\mathcal{K}, D]$. The estimate is for simplicity denoted by θ_k, then the RLS algorithm is given by [38]:

$$\hat{\Theta}_{k+1} = \Theta_k + (y_{k+1} - \Theta_k \phi_{k+1}) \frac{\phi_{k+1}^T P_k}{\lambda + \phi_{k+1}^T P_k \phi_{k+1}}, \tag{2-94}$$

$$P_{k+1} = \frac{1}{\lambda} \left(P_k - \frac{P_k \phi_{k+1} \phi_{k+1}^T P_k}{\lambda + \phi_{k+1}^T P_k \phi_{k+1}} \right). \tag{2-95}$$

$$\tag{2-96}$$

Although RLS is an “exact” recursive solution to the LS problem and is more computationally efficient than LS or BLS, several numerical challenges arise:

1. Overflow of the covariance matrix P due to the accumulation of new observations.
2. Loss of positive definiteness of the covariance matrix $P_k = \left(\Phi_k \Phi_k^T\right)^{-1}$.
3. Quadratic Complexity, with computational cost scaling as $\mathcal{O}\left(p^2(l+m)^2\right)$.

Several strategies have been proposed in the literature to address these issues. Covariance growth is controlled by introducing a forgetting factor λ, which exponentially decays the influence of older data. A practical choice for λ links it to the data window N:

$$\lambda = 1 - \frac{1}{N}. \tag{2-97}$$

This parameter defines the algorithms memory length and improves adaptivity to slow system variations [38, 28]. Positive definiteness is preserved by the Square-Root RLS formulation [52], while computational complexity can be reduced further to linear order $\mathcal{O}(p(m+l))$ using the Fast-Array RLS algorithm [18].

2-7-1 Regularised Recursive Least Squares Algorithm

This subsection derives the general form of the regularised Recursive Least Squares (RLS) algorithm with exponential forgetting. The method minimises a weighted least-squares criterion that penalises large parameter values through a regularisation term $\Pi > 0$ and assigns exponentially decreasing weights to older data through a forgetting factor $0 < \lambda \leq 1$.

Without loss of generality, consider the following exponentially weighted least-squares criterion, where older samples are discounted via the diagonal forgetting matrix Λ_N:

$$\Lambda_N = \mathrm{diag}(\lambda^N, \lambda^{N-1}, \dots \lambda, 1), \tag{2-98}$$

and the following sizes $Y_N \in \mathbb{R}^{1\times N}$, $\Theta_N \in \mathbb{R}^{1\times p}$, $\Phi_N \in \mathbb{R}^{p\times N}$ and $\Pi \in \mathbb{R}^{p\times p}$. The regularised exponentially weighted least-squares problem is then formulated as:

$$\begin{aligned}&\arg\min_{\Theta} \|Y_N - \Theta_N\Phi_N\|^2_{\Lambda_N} + \lambda^{N+1}\|0 - \Theta_N\|^2_{\Pi} = \\ &\arg\min_{\Theta} (Y_N - \Theta_N\Phi_N)\Lambda_N(Y_n - \Theta_N\Phi_N)^T + \lambda^{N+1}\Theta_N\Pi\Theta_N^T.\end{aligned} \tag{2-99}$$

The optimal solution satisfies:

$$\begin{aligned}\Theta_N\left(\lambda^{N+1}\Pi + \Phi_N\Lambda_N\Phi_N^T\right) &= Y_N\Lambda_N\Phi_N^T \\ \Theta_N &= Y_N\Lambda_N\Phi_N^T\left(\lambda^{N+1}\Pi + \Phi_N\Lambda_N\Phi_N^T\right)^{-1}.\end{aligned} \tag{2-100}$$

Since $\Pi > 0$ is positive definite and $\Phi_N\Lambda_N\Phi_N^T \geq 0$ is semi-positive definite, their sum is strictly positive definite, which ensures that the inverse exists.

To derive the recursive update, define the scaled variables that absorb the forgetting weights:

$$\tilde{\Phi}_N = \Phi_N\sqrt{\Lambda}_N, \qquad \tilde{\Phi}_{N-1} = \Phi_{N-1}\sqrt{\Lambda}_{N-1}, \tag{2-101}$$

$$\tilde{Y}_N = Y_N\sqrt{\Lambda}_N, \qquad \tilde{Y}_{N-1} = Y_{N-1}\sqrt{\Lambda}_{N-1}, \tag{2-102}$$

and the corresponding covariance matrices P:

$$P_N = (\lambda^{N+1}\Pi + \tilde{\Phi}_N\tilde{\Phi}_N^T)^{-1}, \tag{2-103}$$

$$P_{N-1} = (\lambda^N\Pi + \tilde{\Phi}_{N-1}\tilde{\Phi}_{N-1}^T)^{-1}, \tag{2-104}$$

with initial condition $P_0 = \Pi^{-1}$. The solution Θ following Equation 2-100 at time $N-1$ and N can be simplified as:

$$\Theta_N = \tilde{Y}_N\tilde{\Phi}_N^T P_N, \tag{2-105}$$

$$\Theta_{N-1} = \tilde{Y}_{N-1}\tilde{\Phi}_{N-1}^T P_{N-1}. \tag{2-106}$$

The entries of $\tilde{\Phi}_N$ and $\tilde{Y}_N$ are closely related to the entries of $\tilde{\Phi}_{N-1}$ and $\tilde{Y}_{N-1}$. The data matrices can be related recursively by appending a column with the new sample (ϕ_N, y_N):

$$\tilde{\Phi}_N = \left[\sqrt{\lambda}\tilde{\Phi}_{N-1}, \phi_N\right], \tag{2-107}$$

$$\tilde{Y}_N = \left[\sqrt{\lambda}\tilde{Y}_{N-1}, y_N\right]. \tag{2-108}$$

Using these relations, P_N^{-1} can be computed recursively as:

$$\begin{aligned}P_N^{-1} &= \lambda^{N+1}\Pi + \tilde{\Phi}_N\tilde{\Phi}_N^T \\ &= \lambda \cdot \underbrace{(\lambda^N\Pi + \tilde{\Phi}_{N-1}\tilde{\Phi}_{N-1}^T)}_{P_{N-1}} + \phi_N\phi_N^T \\ &= \lambda P_{N-1}^{-1} + \phi_N\phi_N^T.\end{aligned} \tag{2-109}$$

Applying the matrix inversion lemma from Equation 2-61 with $A = \lambda P_{N-1}^{-1}$,$B = \phi_N$,$C = 1$, and $D = \phi_N^T$, gives the recursive update for P_N:

$$\begin{aligned} P_N &= (\lambda P_{N-1}^{-1} + \phi_N \phi_N^T)^{-1} \\ &= \frac{1}{\lambda}\left(P_{N-1} - \frac{P_{N-1}\phi_N\phi_N^T P_{N-1}}{\lambda + \phi_N^T P_{N-1}\phi_N}\right), \end{aligned} \tag{2-110}$$

which corresponds to the recursive covariance update in Equation 2-95.

Substituting P_N and the recursive form of $\tilde{Y}_N$ into Equation 2-105, the parameter update becomes:

$$\begin{aligned} \Theta_N &= \tilde{Y}_N \tilde{\Phi}_N^T P_N \\ &= \left(\lambda \tilde{Y}_{N-1}\tilde{\Phi}_{N-1}^T + y_N \phi_N^T\right) P_N \\ &= \frac{1}{\lambda}\left(\lambda \tilde{Y}_{N-1}\tilde{\Phi}_{N-1}^T + y_N \phi_N^T\right)\left(P_{N-1} - \frac{P_{N-1}\phi_N\phi_N^T P_{N-1}}{\lambda + \phi_N^T P_{N-1}\phi_N}\right) \\ &= \underbrace{\tilde{Y}_{N-1}\tilde{\Phi}_{N-1}^T P_{N-1}}_{\Theta_{N-1}} - \underbrace{\tilde{Y}_{N-1}\tilde{\Phi}_{N-1}^T P_{N-1}}_{\Theta_{N-1}} \frac{\phi_N\phi_N^T P_{N-1}}{\lambda + \phi_N^T P_{N-1}\phi_N} + \frac{1}{\lambda} y_N \phi_N^T P_{N-1} - \frac{1}{\lambda} y_N \frac{\phi_N^T P_{N-1}\phi_N}{\lambda + \phi_N^T P_{N-1}\phi_N}\phi_N^T P_{N-1} \\ &= \Theta_{N-1} - \Theta_{N-1}\phi_N \frac{\phi_N^T P_{N-1}}{\lambda + \phi_N^T P_{N-1}\phi_N} + \frac{1}{\lambda} y_N \underbrace{\left(1 - \frac{\phi_N^T P_{N-1}\phi_N}{\lambda + \phi_N^T P_{N-1}\phi_N}\right)}_{\frac{\lambda}{\lambda + \phi_N^T P_{N-1}\phi_N}} \phi_N^T P_{N-1} \\ &= \Theta_{N-1} + (y_N - \Theta_{N-1}\phi_N)\frac{\phi_N^T P_{N-1}}{\lambda + \phi_N^T P_{N-1}\phi_N}. \end{aligned} \tag{2-111}$$

This is equivalent to the model update found in Equation 2-94, now including a regularisation matrix Π.

2-8 Future Directions and Research Plan

This review has explored the foundations and recent developments in data-driven control, with a focus on Subspace Predictive Control (SPC). The key theoretical components underlying SPC have been discussed in detail. This included state-space modelling, polynomial models, subspace identification techniques, least-squares estimation, and model predictive control.

SPC offers a scalable and computationally efficient framework through the use of recursive least squares and is extendable to multi-input multi-output (MIMO) systems. These properties make it attractive for real-time applications. However, existing SPC implementations are typically based on AutoRegressive with eXogenous input (ARX) models, which are limited in their capacity to represent coloured disturbances or complex stochastic dynamics. This limitation reduces their model identification accuracy.

From this literature review, several open areas for future research have been identified. These include:

- Hybrid SPC algorithms that can incorporate partial physical models in line with the recent hybrid DeePC implementation [59].
- Extensions of ARX-based SPC to nonlinear systems.
- The exploration of SPC algorithms which include a more complex model structure that enhance noise and resonance modelling.

This thesis focuses on the third option. Previous studies, such as [19], have shown that AutoRegressive-Moving Average with eXogenous input (ARMAX) models outperform ARX models for smaller past horizons p. Whether this structural extension leads to practical benefits in control performance remains an open question, and it forms the central topic of this thesis.

To study this extension, the following main research questions and sub-questions are formulated:

1. Is it possible to integrate an ARMAX model into the SPC framework to achieve improved noise modelling?
    - What system assumptions are required for the SPC algorithm to function with the ARMAX model?
    - How does the ARMAX model change the identification of the Markov parameters?
    - How does the ARMAX model change the formulation of the data equations?
    - How does the ARMAX model change the predictor equations required for Model Predictive Control?
2. How can ARMAX-based SPC be applied to a real-life system exhibiting an anti-resonance?
    - How can the ARMAX-based SPC algorithm be implemented efficiently within a real-time control framework?

- Which numerical techniques can enhance computational efficiency, accuracy, and robustness?
- How does the control performance and computational cost compare between ARX- and ARMAX-based SPC implementations?
- How does the increased number of parameters in the ARMAX model influence identification accuracy and real-time feasibility?

This research plan provides both a theoretical investigation of the ARMAX-based SPC formulation and a practical evaluation through simulation and laboratory validation.

2-8-1 Methodology

This section explains the methodological approach adopted in this thesis, including the research design, data collection and processing procedures, and the analysis metrics employed. It also justifies the selected techniques and reflects on their limitations.

Research design and approach

The research integrates theoretical development, numerical simulations, and laboratory validation. The aim is to investigate whether ARMAX-based SPC provides measurable advantages over the conventional ARX-based formulation. The theoretical component involves the derivation of the required modifications to the SPC framework, including the least-squares identification, the formulation of data equations, and predictor structure. These adaptations were first evaluated through MATLAB simulations across different model configurations. Subsequently, the algorithm was validated experimentally on a 1-DOF torsional setup to assess its feasibility for real-time implementation. This approach ensures a balance between theoretical analysis and practical verification.

Data collection and experimental setup

Experimental data for model identification and validation were obtained from the Quanser torsional setup, introduced in section 5-2. The identification experiments used a chirp excitation signal, while the control performance was evaluated using a sequence of square, sinusoidal, and sawtooth reference trajectories. The sampling time was set to $T_s = 0.02$s. Noise analysis during idle operation revealed negligible stochastic disturbances but noticeable output quantisation. To ensure comparability, all experiments were performed under identical parameter settings, initial conditions and actuator constraints.

Data analysis and identification methods

The SPC framework was implemented using both ARX and ARMAX models. Identification of the Markov parameters was tested for several estimation methods. For online feasibility, Recursive Least Squares (RLS) and Extended RLS (ERLS) algorithms were implemented, including a forgetting factor to enable adaptation to time-varying dynamics.

The predictor and control law were formulated within the SPC framework and solved using Model Predictive Control (MPC) via MATLABs `quadprog` solver.

The performance was quantified in three ways:

- Model fit: The variance Accounted For (VAF) and Bode plot comparison against known dynamics in case of simulations.
- Control performance: Testing the reference tracking capabilities through the metrics Integral Squared Error (ISE), Integral Absolute Error (IAE), and Input Energy (InEn).
- Computational performance: Testing the real-time implementation of the algorithm through the mean, standard deviation, and maximum computation time for both the identification step and MPC step, compared against the system sampling time.

Justification and limitations

The chosen methodology uses identical datasets, reference trajectories, and controller parameters for both ARX and ARMAX models, enabling a fair comparison of performance. Recursive identification algorithms were implemented to support real-time estimation and adaptation. The selected performance metrics capture not only the identification accuracy but also the resulting control behaviour, as an effective control does not necessarily require a perfectly identified model.

Nevertheless, several limitations were encountered. The experimental setup exhibited some non-linearities, including abrupt output transitions, which cannot be fully captured by linear ARX or ARMAX models. This nonlinearity affected the reliability of the noise estimation in the ARMAX model, particularly since the actual stochastic disturbances were minimal. Furthermore, the metrics ISE and IAE do not account for phase delay, meaning that a system that remains static can appear to perform better than one that tracks the reference with a small delay. The control design for ARMAX assumes deterministic operation under the certainty-equivalence principle, implying that robustness to future unknown noise was not explicitly evaluated.

In addition, the study was restricted to SISO setups. As a result, the scalability of the proposed approach to VARMAX systems remains to be verified. The inclusion of noise-related regressors in the ARMAX formulation also increased parameter correlation, which can affect numerical conditioning and identifiability, particularly for higher model orders. This study did not explicitly address this issue, and more advanced identification techniques would be required to mitigate such correlation effects. Finally, the evaluation was limited to similar periodic reference trajectories, other types of trajectories or frequency ranges may yield different performance metrics.

Overall, this methodology provides a structured framework for extending Subspace Predictive Control from ARX to ARMAX models, combining theoretical analysis, numerical simulations, and experimental evaluation to assess their practical applicability, robustness, and real-time feasibility.

Chapter 3

Subspace Predictive Control for ARMAX models

This chapter extends the Subspace Predictive Control (SPC) framework to systems described by AutoRegressive-Moving Average with eXogenous input (ARMAX) models. While the general structure of SPC for ARMAX is closely related to SPC for ARX models, a key difference lies in the presence of the $C(q)$ polynomial, which introduces a dependency of the current output on past noise terms. Since these noise terms are unmeasured in practice, additional challenges arise in both model identification and prediction.

The objective of this chapter is to derive a complete SPC formulation for ARMAX models, covering the underlying assumptions, the derivation of the data and predictor equations, the setup of the Model Predictive Control (MPC) optimisation problem, and the numerical techniques required for efficient online identification. Each section addresses one of the following research questions:

3-1 Model assumptions: What model assumptions are required for the SPC algorithm to operate with an ARMAX model?

3-2 Algorithm derivation: How can the influence of past noise terms on the current output be modelled within the SPC framework?

3-3 Prediction and control: How can the ARMAX model be integrated into an MPC scheme, given that future noise terms are unknown? How is the future output $\tilde{y}_{\text{pred}}$ computed under these conditions?

3-4 Identification: How can the ARMAX model parameters be estimated when the noise sequence is unobserved? Which numerical techniques ensure efficient, robust, and accurate identification?

3-5 ARX and ARMAX comparison: What are the main differences between the ARX- and ARMAX-based SPC algorithms in terms of notation, computational load, identification, and MPC?

3-1 Introduction to ARMAX Models

We now assume that the system can be described by an AutoRegressive-Moving Average model with eXogenous input (ARMAX). The general discrete-time representation of the ARMAX model is described by:

$$y(k) = \frac{B(q)}{A(q)}u(k) + \frac{C(q)}{A(q)}e(k), \tag{3-1}$$

where q^{-1} is the backward shift operator, $u(k)$ denotes the control input, $y(k)$ the measured output, and $e(k)$ a zero mean white noise disturbance with standard deviation σ_e. The model polynomials are defined as

$$\begin{aligned} A(q) &= 1 + a_1 q^{-1} + \cdots + a_{k_a} q^{-k_a}, \\ B(q) &= b_0 + b_1 q^{-1} + \cdots + b_{k_b} q^{-k_b}, \\ C(q) &= 1 + c_1 q^{-1} + \cdots + c_{k_c} q^{-k_c}. \end{aligned}$$

In the time domain, this is equivalent to an intuitive expression:

$$\begin{aligned} \underbrace{y(k) + a_1 y(k-1) + \cdots + a_{k_a} y(k-k_a)}_{\text{Autoregressive (AR)}} &= \underbrace{b_0 u(k) + b_1 u(k-1) + \cdots + b_{k_b} u(k-k_b)}_{\text{Exogenous input (X)}} \\ &\quad + \underbrace{e(k) + c_1 e(k-1) + \cdots + c_{k_c} e(k-k_c)}_{\text{Moving average (MA)}}. \end{aligned} \tag{3-2}$$

This formulation shows that the current output depends on the past outputs, the current and past inputs, and the current and previous noise terms.

The ARMAX structure extends the simpler ARX model by including the moving average term $C(q)$, which explicitly models the influence of past noise on the current output. While this generally improves the accuracy and robustness of the identified model, it also introduces new challenges, since the noise sequence $e(k)$ is normally unmeasured in practice. This complicates the model identification and its integration into the SPC algorithm.

For the derivations in the following sections, the following assumptions are made regarding the ARMAX model:

- The innovation $e(k)$ is a zero mean, independent, identically distributed (i.i.d.) white noise with variance σ_e^2, such that the resulting disturbance $C(q)/A(q)$ is potentially coloured.
- The roots of $A(q)$, corresponding to the system poles, lie inside the unit circle.
- The roots of $C(q)$, corresponding to the noise model zeros, lie inside the unit circle to guarantee invertibility and stability of the noise model.

It is important to note the distinction between the innovation sequence $e(k)$ and the resulting disturbance acting on the system output. While $e(k)$ is assumed to be a zero mean, i.i.d. white noise process, the moving-average polynomial $C(q)$ shapes this sequence, producing a coloured disturbance term $C(q)/A(q)e(k)$ that allows temporal correlation. This coloured noise representation enables the ARMAX model to capture systems with more complex stochastic dynamics than those modelled by the ARX structure, in which the disturbance directly equals the innovation, and should therefore be purely white.

3-2 Least-Squares Formulation for the Identification of ARMAX Models

This section introduces a modified form of the Kalman innovation model (Equation 2-7) to derive the least-squares formulation for ARMAX-based SPC. Recall the original innovation form:

$$\begin{aligned} x_{k+1} &= Ax_k + Bu_k + Ke_k, \\ y_k &= Cx_k + Du_k + e_k. \end{aligned} \tag{3-3}$$

By adding and subtracting My_k, the model can be rewritten as:

$$\begin{aligned} x_{k+1} &= Ax_k + Bu_k - My_k + My_k + Ke_k, \\ y_k &= Cx_k + Du_k + e_k. \end{aligned} \tag{3-4}$$

Substituting y_k into the first equation yields:

$$x_{k+1} = Ax_k + Bu_k - M(Cx_k + Du_k + e_k) + My_k + Ke_k \tag{3-5}$$

$$= (A - MC)x_k + (B - MD)u_k + My_k + (K - M)e_k. \tag{3-6}$$

This representation is known as the deadbeat/Kalman predictor form [19]:

$$x_{k+1} = \tilde{A}x_k + \tilde{B}u_k + My_k + \tilde{K}e_k, \tag{3-7}$$

$$y_k = Cx_k + Du_k + e_k, \tag{3-8}$$

where $\tilde{A} = A - MC$, $\tilde{B} = B - MD$, and $\tilde{K} = K - M$. The matrix M can be selected to dampen the dynamics of $\tilde{A}$ compared to those of A. In the case of a deadbeat observer, M is often chosen such that all eigenvalues are located at the origin. This implies that $\tilde{A}^n = 0$, where n is the true model order, and consequently, for $p > n$, $\|\tilde{A}^p\|_2 = 0$. This property is well known in the Observer/Kalman Identification (OKID) method.

To derive the least-squares formulation, consider the next time step:

$$x_{k+2} = \tilde{A}x_{k+1} + \tilde{B}u_{k+1} + My_{k+1} + \tilde{K}e_{k+1}, \tag{3-9}$$

$$y_{k+2} = Cx_{k+2} + Du_{k+2} + e_{k+2}. \tag{3-10}$$

Substituting Equation 3-7 into Equation 3-9, and then into Equation 3-10, gives:

$$\begin{aligned}
x_{k+2} &= \tilde{A}\left(\tilde{A}x_k + \tilde{B}u_k + My_k + Ke_k\right) + \tilde{B}u_{k+1} + My_{k+1} + \tilde{K}e_{k+1} \\
&= \tilde{A}^2x_k + \tilde{A}\tilde{B}u_k + \tilde{A}My_k + \tilde{A}Ke_k + \tilde{B}u_{k+1} + My_{k+1} + \tilde{K}e_{k+1} \\
&= \tilde{A}^2x_k + \left[\begin{array}{cc|cc|cc} \tilde{A}M & M & \tilde{A}\tilde{B} & \tilde{B} & \tilde{A}\tilde{K} & \tilde{K} \end{array}\right] \left[\begin{array}{c} y_k \\ y_{k+1} \\ \hline u_k \\ u_{k+1} \\ \hline e_k \\ e_{k+1} \end{array}\right], \qquad (3\text{-}11)\\
y_{k+2} &= Cx_{k+2} + Du_{k+2} + e_{k+2} \\
&= C\tilde{A}^2x_k + \left[\begin{array}{cc|cc|cc} C\tilde{A}M & CM & C\tilde{A}\tilde{B} & C\tilde{B} & C\tilde{A}\tilde{K} & C\tilde{K} \end{array}\right] \left[\begin{array}{c} y_k \\ y_{k+1} \\ \hline u_k \\ u_{k+1} \\ \hline e_k \\ e_{k+1} \end{array}\right] + Du_{k+2} + e_{k+2}.
\end{aligned} \qquad (3\text{-}12)$$

Following the same approach, similar expressions can be derived for general time steps $k+p$. Defining the matrix $\mathcal{K}$ for ARMAX models as:

$$\mathcal{K} = \left[\begin{array}{cccc|cccc|cccc} \tilde{A}^{p-1}M & \cdots & \tilde{A}M & M & \tilde{A}^{p-1}\tilde{B} & \cdots & \tilde{A}\tilde{B} & \tilde{B} & \tilde{A}^{p-1}\tilde{K} & \cdots & \tilde{A}\tilde{K} & \tilde{K} \end{array}\right], \qquad (3\text{-}13)$$

and a stacked sample data of outputs, inputs, and estimated noise terms, denoted by $z_k^{(p)} \in \mathbb{R}^{lp+mp+lp\times 1}$:

$$z_k^{(p)} = \left[\begin{array}{cccc|cccc|cccc} y_k^T & y_{k+1}^T & \cdots & y_{k+p-1}^T & u_k^T & u_{k+1}^T & \cdots & u_{k+p-1}^T & e_k^T & e_{k+1}^T & \cdots & e_{k+p-1}^T \end{array}\right]^T. \qquad (3\text{-}14)$$

Then, for general $p \in \mathbb{N}$, x_{k+p} and y_{k+p} can be written as:

$$x_{k+p} = \tilde{A}^p x_k + \mathcal{K}z_k^{(p)}, \qquad (3\text{-}15)$$

$$y_{k+p} = C\tilde{A}^p x_k + C\mathcal{K}z_k^{(p)} + Du_{k+p} + e_{k+p}, \qquad (3\text{-}16)$$

where $C\mathcal{K}$ is given by:

$$C\mathcal{K} = \left[\begin{array}{ccc|ccc|ccc} C\tilde{A}^{p-1}M & \cdots & CM & C\tilde{A}^{p-1}\tilde{B} & \cdots & C\tilde{B} & C\tilde{A}^{p-1}\tilde{K} & \cdots & C\tilde{K} \end{array}\right]. \qquad (3\text{-}17)$$

Assuming that all eigenvalues of $\tilde{A} = A - MC$ lie strictly inside the unit circle, the state-dependent term decays asymptotically, leading to:

$$x_{k+p} = \mathcal{K}z_k^{(p)}, \qquad (3\text{-}18)$$

$$y_{k+p} = C\mathcal{K}z_k^{(p)} + Du_{k+p} + e_{k+p}. \qquad (3\text{-}19)$$

These equations are similar to the ARX case in equations (2-29)–(2-36), with the key distinction that $z_k^{(p)}$ now includes the additional noise terms $e_k, \dots, e_{k+p-1}$. For completeness, the complete data equation is expressed as:

$$Y_f = C\mathcal{K}Z_p + DU_f + E_f, \tag{3-20}$$

where Y_f, Z_p, U_f and E_f are defined as:

$$Y_f = \begin{bmatrix} y_{k+p} & y_{k+p+1} & \cdots & y_{k+p+N-1} \end{bmatrix} \in \mathbb{R}^{lp\times N}, \tag{3-21}$$

$$Z_p = \begin{bmatrix} z_k^{(p)} & z_{k+1}^{(p)} & \cdots & z_{k+N-1}^{(p)} \end{bmatrix} \in \mathbb{R}^{(l+m+l)p\times N}, \tag{3-22}$$

$$U_f = \begin{bmatrix} u_{k+p} & u_{k+p+1} & \cdots & u_{k+p+N-1} \end{bmatrix} \in \mathbb{R}^{m\times N}, \tag{3-23}$$

$$E_f = \begin{bmatrix} e_{k+p} & e_{k+p+1} & \cdots & e_{k+p+N-1} \end{bmatrix} \in \mathbb{R}^{l\times N}. \tag{3-24}$$

Finally, the least-squares estimation problem is then formulated as:

$$\widehat{[C\mathcal{K}, D]} = \underset{[C\mathcal{K},D]}{\arg\min} \left\| Y_f - [C\mathcal{K}, D] \begin{pmatrix} Z_p \\ U_f \end{pmatrix} \right\|_F^2 . \tag{3-25}$$

This optimisation problem is nonlinear due to the dependence on the unmeasured noise sequence $e_k, \dots, e_{k+p-1}$. The estimation of these noise terms, and the numerical computation of the least-squares solution, are discussed in section 3-4. The next section introduces the formulation of the Model Predictive Control algorithm based on the identified ARMAX model.

3-3 Model Predictive Control for ARMAX Models

The predictor Markov parameters obtained from the least-squares problem are used to construct an output predictor based on Equation 3-20:

$$y_{k+p} = \widehat{C\mathcal{K}} z_k^{(p)} + \widehat{D} u_{k+p} + e_{k+p}. \tag{3-26}$$

Let us define the sequence of inputs and outputs over the prediction horizon N_p as follows:

$$\tilde{y}_{\text{pred}} = \begin{bmatrix} y_{k+1} \\ y_{k+2} \\ \vdots \\ y_{k+N_p} \end{bmatrix}, \qquad \tilde{u}_k = \begin{bmatrix} u_{k+1} \\ u_{k+2} \\ \vdots \\ u_{k+N_p} \end{bmatrix}, \qquad \text{and} \qquad \tilde{e}_k = \begin{bmatrix} e_{k+1} \\ e_{k+2} \\ \vdots \\ e_{k+N_p} \end{bmatrix}. \tag{3-27}$$

With this notation, the predicted output sequence $\tilde{y}_{\text{pred}} \in \mathbb{R}^{lN_p \times 1}$ can be expressed as:

$$\tilde{y}_{\text{pred}} = \tilde{\Gamma} z_k^{(p)} + \tilde{G}\tilde{y}_{\text{pred}} + \tilde{H}\tilde{u}_k + \tilde{F}\tilde{e}_k. \tag{3-28}$$

Here, $\tilde{\Gamma}$ contains the effects of previously known inputs, outputs, and estimated residuals, while $\tilde{H}$, $\tilde{G}$, and $\tilde{F}$ are lower block-triangular Toeplitz matrices representing the effects of future unknown inputs, outputs, and noise terms, respectively. For $N_p \leq p$, these matrices are given by:

$$\tilde{\Gamma} = \begin{bmatrix} \Gamma_M & \Gamma_B & \Gamma_K \end{bmatrix}, \tag{3-29}$$

$$\text{with } \Gamma_X = \begin{bmatrix} C\tilde{A}^{p-1}\tilde{X} & C\tilde{A}^{p-2}\tilde{X} & \cdots & C\tilde{X} \\ & \ddots & & \vdots \\ & C\tilde{A}^{p-1}\tilde{X} & \cdots & C\tilde{A}^{N_p-1}\tilde{X} \end{bmatrix}, \quad \text{for } X \in \{M, B, K\}, \tag{3-30}$$

$$\tilde{G} = \begin{bmatrix} 0 & & & & \\ C\tilde{M} & 0 & & & \\ C\tilde{A}\tilde{M} & C\tilde{M} & 0 & & \\ \vdots & & \ddots & \ddots & \\ C\tilde{A}^{N_p-2}\tilde{M} & C\tilde{A}^{N_p-3}\tilde{M} & \dots & C\tilde{M} & 0 \end{bmatrix}, \tag{3-31}$$

$$\tilde{H} = \begin{bmatrix} D & & & & \\ C\tilde{B} & D & & & \\ C\tilde{A}\tilde{B} & C\tilde{B} & D & & \\ \vdots & & \ddots & \ddots & \\ C\tilde{A}^{N_p-2}\tilde{B} & C\tilde{A}^{N_p-3}\tilde{B} & \dots & C\tilde{B} & D \end{bmatrix}, \tag{3-32}$$

$$\tilde{F} = \begin{bmatrix} 1 & & & & \\ C\tilde{K} & 1 & & & \\ C\tilde{A}\tilde{K} & C\tilde{K} & 1 & & \\ \vdots & & \ddots & \ddots & \\ C\tilde{A}^{N_p-2}\tilde{K} & C\tilde{A}^{N_p-3}\tilde{K} & \dots & C\tilde{K} & 1 \end{bmatrix}. \tag{3-33}$$

When $N_p > p$, the matrix entries are capped by the maximum identified power $\tilde{A}^{p-1}$, with higher-order terms replaced by zero matrices of appropriate size. In this case, $\tilde{H}$, $\tilde{G}$, and $\tilde{F}$ remain lower block-triangular Toeplitz matrices, whose first columns contain the following vectors $v_{\tilde{H}}$, $v_{\tilde{G}}$, and $v_{\tilde{F}}$ respectively:

$$v_{\tilde{G}} = \begin{bmatrix} 0 & C\tilde{M} & C\tilde{A}\tilde{M} & \dots & C\tilde{A}^{p-1}\tilde{M} & \mathbf{0}_{l\times l(N_p-p)} \end{bmatrix}^T, \tag{3-34}$$

$$v_{\tilde{H}} = \begin{bmatrix} D & C\tilde{B} & C\tilde{A}\tilde{B} & \dots & C\tilde{A}^{p-1}\tilde{B} & \mathbf{0}_{l\times m(N_p-p)} \end{bmatrix}^T, \tag{3-35}$$

$$v_{\tilde{F}} = \begin{bmatrix} 1 & C\tilde{K} & C\tilde{A}\tilde{K} & \dots & C\tilde{A}^{p-1}\tilde{K} & \mathbf{0}_{l\times l(N_p-p)} \end{bmatrix}^T. \tag{3-36}$$

The corresponding matrix $\tilde{\Gamma}$ for this case follows the same structure, capped at $\tilde{A}^{p-1}$:

$$\tilde{\Gamma} = \begin{bmatrix}\Gamma_M & \Gamma_B & \Gamma_K\end{bmatrix},$$

$$\text{with } \Gamma_X = \begin{bmatrix} C\tilde{A}^{p-1}\tilde{X} & \dots & C\tilde{X} \\ & \ddots & \\ & & C\tilde{A}^{p-1}\tilde{X} \\ & 0 & \end{bmatrix}, \quad \text{for } X \in \{M, B, K\}. \tag{3-37}$$

It is important to note that the future error terms in Equation 3-28 are unknown. A common approach in Robust Model Predictive Control (RMPC) [36] or Stochastic Model Predictive Control (SMPC) [16] is to explicitly incorporate this uncertainty into the optimisation problem.

However, the deterministic MPC formulation used in this work applies the certainty equivalence principle [39], which assumes that the expected value of future disturbances is zero. This approach, common in Linear Quadratic Gaussian (LQG) control and dynamic programming, implies that the optimal control policy is unaffected by replacing the random disturbance with its expected value. Consequently, the term $\tilde{F}\tilde{e}_k$ in Equation 3-28 is neglected, leading to the deterministic predictor:

$$\tilde{y}_{\text{pred}} = \tilde{\Gamma} z_k^{(p)} + \tilde{G}\tilde{y}_{\text{pred}} + \tilde{H}\tilde{u}_k. \tag{3-38}$$

3-4 Computation of the Least Squares Solution for ARMAX Models

Several numerical techniques have been proposed in the literature to solve the ARMAX least-squares problem. These include Generalised Least Squares (GLS), Recursive Extended Least Squares (RELS), Approximate Maximum Likelihood (AML), Recursive Maximum Likelihood (RML), Modified Extended Recursive Least Squares (MERLS) [62], and Latest-Estimation Based Hierarchical Recursive Extended Least Squares (LE-HRELS) [61].

The last two methods (MERLS and LE-HRELS) decompose the identification problem into two stages: one for the parameters associated with the deterministic input-output dynamics, and another for the parameters associated with the stochastic noise model. This decoupling allows the algorithms to use the most recent information for each subsystem and slightly reduces the computational cost associated with matrix-vector multiplications. However, MERLS has not demonstrated significantly improved identification performance compared to the standard approaches. LE-HRELS provides more accurate noise estimation by updating the past p residuals with the latest model parameters at each iteration, similar to iterative batch ARMAX implementations. The main limitation of LE-HRELS is its higher memory requirement in real-time applications, as two past data windows must be stored to predict y_k at each iteration, adding approximately $p-1$ vector-vector multiplications per time step.

In this thesis, the focus is placed on the Extended Recursive Least Squares (ERLS) algorithm. This method provides a direct extension of the Recursive Least Squares (RLS) algorithm described in section 2-7. The extension incorporates the estimated noise terms into the regression vector ϕ_k. Two common variants of ERLS exist, depending on how the innovation sequence is estimated:

- The a-priori error estimation: the noise $\hat{e}_k$ is computed using the parameter estimate from the previous iteration, $\hat{\theta}_{k-1}$.
- The a-posteriori error estimation: the noise $\hat{e}_k$ is computed based on the residual obtained through the most recently updated parameter estimate, $\hat{\theta}_k$. This corresponds to the AML method.

The a-posteriori method shows better convergence properties compared to the a-priori method, as it uses the most recent parameter update when computing the residual. When the prediction error converges to the residual error, both approaches yield equivalent results. Therefore, in the rest of the thesis, the ERLS algorithm is implemented in its a-posteriori form.

It is noted that convergence can be affected if the following positive real condition is not satisfied [28, p. 211]:

$$\mathfrak{Re}\left(H(e^{i\omega})^{-1} - \tfrac{1}{2}\right) > 0, \quad \forall \omega. \tag{3-39}$$

The Extended Recursive Least Squares algorithm for the identification of an ARMAX model is summarised in Algorithm 2.

Algorithm 2 Extended Recursive Least Squares (ERLS) for ARMAX models

```
1: Initialise parameter vector $\hat{\theta}_0$, covariance matrix $P_0 > 0$, and noise estimates $\hat{e}_0 = 0$.
2: for $k = 1, 2, \dots$ do
3:     Form the data vector $\phi_k$ from past inputs $u$, outputs $y$, and estimated noise terms $\hat{e}$.
4:     Compute the prediction $\hat{y}_k = \hat{\theta}_{k-1}\phi_k$.
5:     Compute the prediction error $\hat{e}_k = y_k - \hat{\theta}_{k-1}\phi_k$.          (a-priori)
6:     Compute the Kalman gain:
           $$K_k = \frac{P_{k-1}\phi_k}{\lambda + \phi_k^T P_{k-1}\phi_k}$$
7:     Update parameter row vector:
           $$\hat{\theta}_k = \hat{\theta}_{k-1} + e_k K_k^T$$
8:     Update covariance:
           $$P_k = \frac{1}{\lambda}\Big(P_{k-1} - K_k\phi_k^T P_{k-1}\Big)$$
9:     Compute the residual $r_k = y_k - \theta_k\phi_k$.
10:    Update the noise estimate: $\hat{e}_k \leftarrow r_k$.          (a-posteriori)
11: end for
```

3-5 Comparison Between ARX- and ARMAX-based SPC

This section provides a summary of the differences between ARX- and ARMAX-based formulations of the Subspace Predictive Control (SPC) algorithm. The goal is to highlight differences in notation, identification requirements, Model Predictive Control (MPC) implementation, and computational cost.

Notation

In the ARX formulation, the system is expressed in terms of AutoRegressive dynamics with direct dependence on past inputs and outputs. The ARMAX formulation extends this structure by explicitly including a moving-average noise component. This requires additional parameters to capture the influence of past disturbances on the current output. ARX assumes an innovation form, while ARMAX is based on the deadbeat predictor form. These forms result in slightly different definitions summarised in Table 3-1.

Table 3-1: Comparison of ARX- and ARMAX-based Subspace Predictive Control (SPC) in terms of notation.

	ARX	**ARMAX**
# Parameters	$(l+m)p$	$(2l+m)p$
Data vector $z_k^{(p)}$	$\begin{bmatrix} y_k^T & u_k^T \end{bmatrix}^T$	$\begin{bmatrix} y_k^T & u_k^T & e_k^T \end{bmatrix}^T$
$\mathcal{K}$	$\left[\ \tilde{A}^{p-1}K \ \dots \ K \ \vdots \ \tilde{A}^{p-1}\tilde{B} \ \dots \ \tilde{B}\ \right]$	$\left[\ \tilde{A}^{p-1}M \ \dots \ M \ \vdots \ \tilde{A}^{p-1}\tilde{B} \ \dots \ \tilde{B} \ \vdots \ \tilde{A}^{p-1}\tilde{K} \ \dots \ \tilde{K}\ \right]$
$\tilde{A}$	$A - KC$	$A - MC$
$\tilde{B}$	$B - KD$	$B - MD$
$\tilde{K}$	K	$K - M$

Identification

For ARX models, identification reduces to estimating the Markov Parameters using least-squares methods. This is a linear problem and can be solved efficiently with batch or recursive approaches such as RLS. In contrast, the identification of ARMAX models is nonlinear, since the noise terms are unknown and need to be estimated. This requires more elaborate methods such as multi-stage least squares, iterative schemes, or recursive approaches like ERLS. The latter is adopted in this thesis because of its suitability for real-time implementation.

Compared to ARX, ERLS introduces an additional lp parameters to capture past noise estimates, which increases both the dimensionality of the parameter vector and the size of the associated covariance matrix. As a result, the computational cost per update is higher, however both methods have equal, and quadratic complexity in the number of parameters. The differences in matrix dimensions are summarised in Table 3-2, while the corresponding computational costs are listed in Table 3-3. ERLS also requires an additional residual computation step, which is not present in RLS.

For a single-input single-output (SISO) case with $l = m = 1$, we have $n = 2p$ and $\tilde{n} = 3p$, meaning that ERLS requires approximately $(3/2)^2 = 2.25$ times more computations per update than RLS. While this increases the runtime of the algorithm, the added noise terms in ERLS improve the identification of the dynamic model and disturbance model significantly.

Table 3-2: Comparison of matrix dimensions for RLS (ARX) and ERLS (ARMAX).

	RLS (ARX)	**ERLS (ARMAX)**
Data vector ϕ_k	$(l+m)p \times 1$	$(2l+m)p \times 1$
Parameter vector $\hat{\theta}_k$	$(l+m)p \times 1$	$(2l+m)p \times 1$
Covariance P_k	$((l+m)p) \times ((l+m)p)$	$((2l+m)p) \times ((2l+m)p)$
Kalman gain K_k	$(l+m)p \times 1$	$(2l+m)p \times 1$
Noise terms included	No	Yes ($\hat{e}_{k-i},\ i = 1, \dots, p$)

Table 3-3: Approximate computational cost of each step in Recursive Least Squares (RLS, ARX) and Extended Recursive Least Squares (ERLS, ARMAX). Here $n = (l+m)p$ and $\tilde{n} = (2l+m)p$ denote the number of parameters. The dominant terms are highlighted in bold.

Operation	**RLS (ARX)**	**ERLS (ARMAX)**
(3) Form data vector ϕ_k	$\mathcal{O}(n)$	$\mathcal{O}(\tilde{n})$
(4) Prediction $\hat{y}_k = \hat{\theta}_{k-1}^\top \phi_k$	$\mathcal{O}(n)$	$\mathcal{O}(\tilde{n})$
(5) Prediction error $\hat{e}_k = y_k - \hat{y}_k$	$\mathcal{O}(1)$	$\mathcal{O}(1)$
(6) Gain numerator $v = P_{k-1}\phi_k$	$\mathcal{O}(\mathbf{n^2})$	$\mathcal{O}(\mathbf{\tilde{n}^2})$
(6) Gain denominator $d = \phi_k^\top v$	$\mathcal{O}(n)$	$\mathcal{O}(\tilde{n})$
(6) Kalman gain $K_k = v/d$	$\mathcal{O}(n)$	$\mathcal{O}(\tilde{n})$
(7) Parameter update $\hat{\theta}_k = \hat{\theta}_{k-1} + K_k \hat{e}_k$	$\mathcal{O}(n)$	$\mathcal{O}(\tilde{n})$
(8) Covariance update $P_k = \lambda^{-1}(P_{k-1} - K_k v^\top)$	$\mathcal{O}(\mathbf{n^2})$	$\mathcal{O}(\mathbf{\tilde{n}^2})$
(9) Residual (a-post.) $r_k = y_k - \hat{\theta}_k^\top \phi_k$	Not required	$\mathcal{O}(\tilde{n})$
Total per update	$\mathcal{O}(\mathbf{n^2})$	$\mathcal{O}(\mathbf{\tilde{n}^2})$

MPC formulation

Both ARX- and ARMAX-based models can be included into the SPC framework to generate predictors for Model Predictive Control (MPC). For ARX models, the predictor equations are directly obtained from the identified Markov Parameters. In contrast, ARMAX models include an additional noise model, which modifies the predictor equations and alters the construction of the prediction matrices. Since future noise terms are unmeasured and therefore unknown, the standard assumption is to set them equal to zero. Under this assumption, the ARX and ARMAX formulations share the same predictor structure, differing only in the definitions of the matrices. The predicted output is given by

$$\tilde{y}_{\text{pred}} = \tilde{\Gamma} z_k^{(p)} + \tilde{G}\, \tilde{y}_{\text{pred}} + \tilde{H}\, \tilde{u}_k, \tag{3-40}$$

where $\tilde{\Gamma}$ is an upper-triangular Toeplitz matrix, and $\tilde{G}$ and $\tilde{H}$ are lower-triangular Toeplitz matrices defined below.

ARX formulation. For ARX models, the first row of $\tilde{\Gamma}$ is

$$v_{\tilde{\Gamma}} = \left[\begin{array}{ccc:ccc} C\tilde{A}^{p-1}K & \dots & CK & C\tilde{A}^{p-1}\tilde{B} & \dots & C\tilde{B} \end{array}\right], \tag{3-41}$$

while the first columns of $\tilde{G}$ and $\tilde{H}$ are given by

$$v_{\tilde{G}} = \left[\begin{array}{cccccc} 0 & CK & C\tilde{A}K & \dots & C\tilde{A}^{p-1}K & \mathbf{0}_{l\times l(N_p-p)} \end{array}\right]^T, \tag{3-42}$$

$$v_{\tilde{H}} = \left[\begin{array}{cccccc} D & C\tilde{B} & C\tilde{A}\tilde{B} & \dots & C\tilde{A}^{p-1}\tilde{B} & \mathbf{0}_{l\times m(N_p-p)} \end{array}\right]^T. \tag{3-43}$$

ARMAX formulation. For ARMAX models, the first row of $\tilde{\Gamma}$ is extended with additional terms arising from the noise model:

$$v_{\tilde{\Gamma}} = \left[\begin{array}{ccc:ccc:ccc} C\tilde{A}^{p-1}\tilde{M} & \dots & C\tilde{M} & C\tilde{A}^{p-1}\tilde{B} & \dots & C\tilde{B} & C\tilde{A}^{p-1}\tilde{K} & \dots & C\tilde{K} \end{array}\right]. \tag{3-44}$$

The corresponding first columns of $\tilde{G}$ and $\tilde{H}$ are

$$v_{\tilde{G}} = \left[\begin{array}{cccccc} 0 & C\tilde{M} & C\tilde{A}\tilde{M} & \dots & C\tilde{A}^{p-1}\tilde{M} & \mathbf{0}_{l\times l(N_p-p)} \end{array}\right]^T, \tag{3-45}$$

$$v_{\tilde{H}} = \left[\begin{array}{cccccc} D & C\tilde{B} & C\tilde{A}\tilde{B} & \dots & C\tilde{A}^{p-1}\tilde{B} & \mathbf{0}_{l\times m(N_p-p)} \end{array}\right]^T. \tag{3-46}$$

In summary, both ARX and ARMAX predictors share the same structural form but slightly differ in the definitions of their Toeplitz matrices.

Chapter 4

Simulation Results

This chapter presents the online simulation results for the models that will be introduced in section 4-1. For the inertia-spring-damping model, the identification accuracy of ARX and ARMAX structures is examined in section 4-2. The uncertainty of the identification process is then analysed using Monte Carlo simulations and the first-order variance approximation. Subsequently, higher-order ARX and ARMAX models are investigated in section 4-3. Finally, the complete Subspace Predictive Control (SPC) algorithm, which integrates the identification of the Markov parameters with the Model Predictive Control (MPC) framework, is evaluated in section 4-4.

4-1 Model Descriptions

The following systems are considered for simulation purposes:

A) The inertia-spring-damper system,

B) A 5th order SISO model, that represents a laboratory test setup with two circular plates and flexible shafts from [49].

System A is studied thoroughly in the subsequent subsections, while some identification and control results are added in the appendix for Model B. For all models, data is generated using the discrete-time state-space representation

$$x_{k+1} = Ax_k + Bu_k + Ke_k, \tag{4-1}$$

$$y_k = Cx_k + Du_k + e_k, \tag{4-2}$$

with zero initial condition $x_0 = 0$. The term e_k is measurement noise with standard deviation σ.

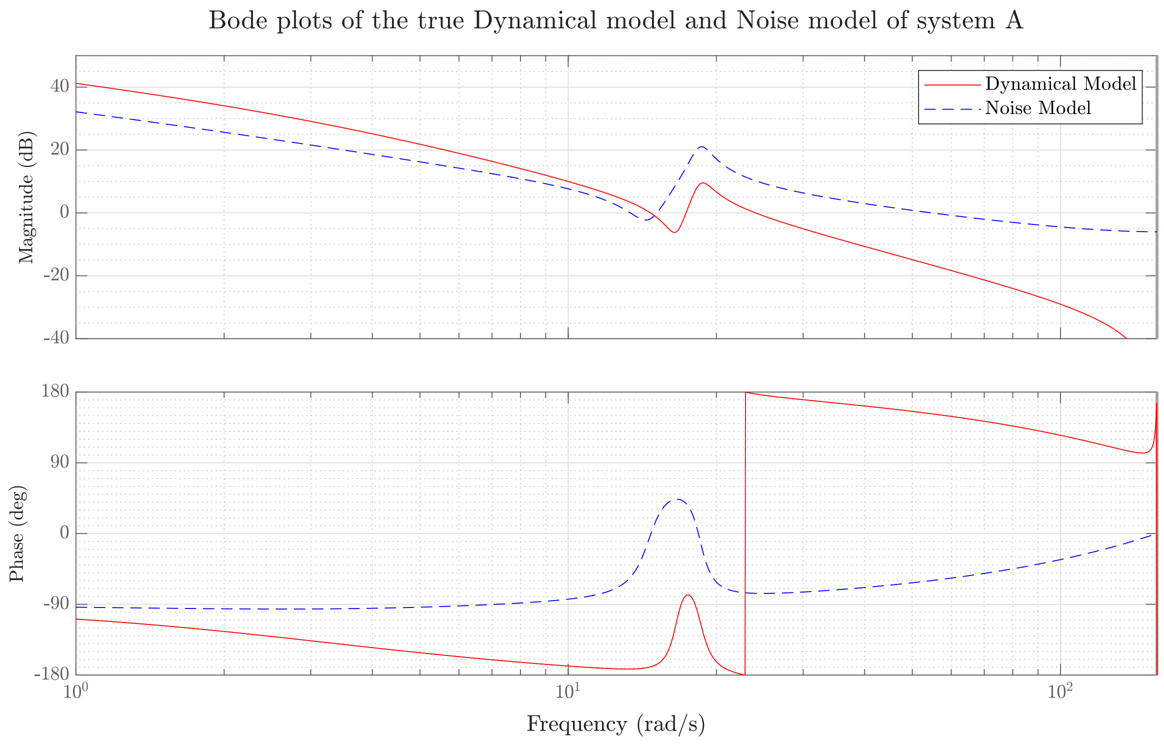


Figure 4-1: Bode plots of the dynamical and noise models for the Inertia-Spring-Damper system.

Inertia-spring-damper system

The first system represents a torsional setup with state variables $x = \begin{bmatrix} \theta_1 & \dot{\theta}_1 & \theta_2 & \dot{\theta}_2 \end{bmatrix}^T$, where the input u is a torque acting on inertia J_1. The parameters are

$$k = 0.5, \quad b_1 = 0.015, \quad b_2 = 0.0015, \quad J_1 = 2.18 \times 10^{-3}, \quad J_2 = 5.45 \times 10^{-4}$$

The corresponding state-space model, derived from first principles (see section 5-1), is

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -\frac{k}{J_1} & -\frac{b_1}{J_1} & \frac{k}{J_1} & 0 \\ 0 & 0 & 0 & 1 \\ \frac{k}{J_2} & 0 & -\frac{k}{J_2} & -\frac{b_2}{J_2} \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ \frac{1}{J_1} \\ 0 \\ 0 \end{bmatrix}, \quad C = \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix}, \quad D = 0.$$

The continuous-time system is discretised using the zero-order hold (ZOH) method with a sampling time of $t_s = 0.02$ s. The Kalman gain K is computed as described in Equation 2-9, using the `idare` command in MATLAB. The Bode plot of this model, shown in Figure 4-1, reveals both a resonance frequency and an anti-resonance frequency in the dynamic model as well as in the noise model, these resonances make the model an ideal candidate for testing an ARMAX model.

5th-order SISO model

The 5th-order single-input single-output (SISO) model represents a laboratory setup consisting of two circular plates connected by flexible shafts [49]. Its discrete-time system matrices are defined as:

$$A = \begin{bmatrix} 4.4 & 1 & 0 & 0 & 0 \\ -8.09 & 0 & 1 & 0 & 0 \\ 7.83 & 0 & 0 & 1 & 0 \\ -4 & 0 & 0 & 0 & 1 \\ 0.86 & 0 & 0 & 0 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 0.00098 \\ 0.01299 \\ 0.01859 \\ 0.0033 \\ -0.00002 \end{bmatrix},$$

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \end{bmatrix}, \qquad K = \begin{bmatrix} 2.3 \\ -6.64 \\ 7.515 \\ -4.0146 \\ 0.86336 \end{bmatrix}, \quad D = 0 \tag{4-3}$$

The corresponding Bode plots of the dynamical and noise models are shown in Figure 4-2. The system exhibits two resonance peaks in both the dynamical and noise models but does not display an anti-resonance in its noise dynamics. Although this model is not discussed in detail in the main text, its control results, similar to those presented in section 4-4, are included in section A-2 to demonstrate consistency with the findings from the inertia-spring-damper system.

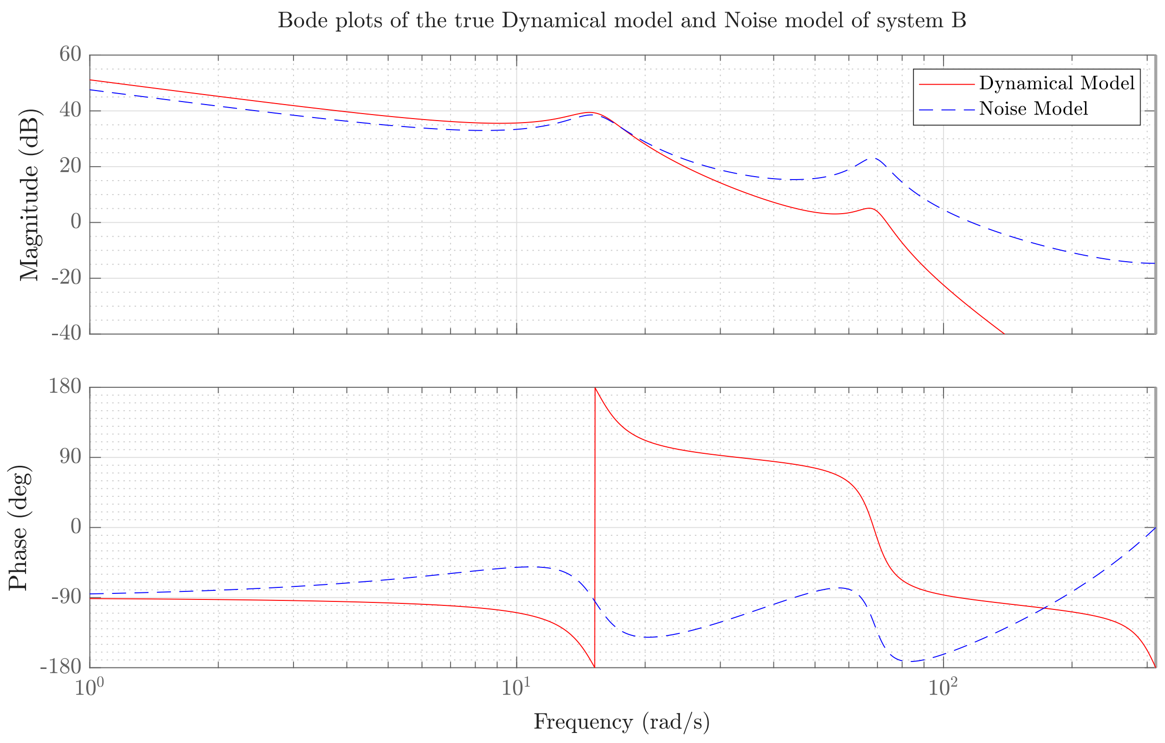


Figure 4-2: Bode plots of the dynamical and noise models for the 5th order SISO model.

4-2 Identification Results for ARX and ARMAX Models

This section evaluates the identification performance of ARX and ARMAX models using the recursive least-squares algorithms introduced in section 2-7 and section 3-4. To account for uncertainty, the simulations incorporate different noise realisations. The goal is to study how uncertainty in the system parameters propagates through the model. The robustness of the algorithms is therefore assessed through Monte Carlo simulations in subsection 4-2-1. As the number of simulations required for this approach is often computationally infeasible in practice, a first-order variance approximation in the frequency domain is also derived in subsection 4-2-2.

4-2-1 Monte Carlo Variance Estimation in the Frequency Domain

A common approach to evaluate the statistical properties of an estimator is the Monte Carlo method. In this procedure, a large number of independent datasets are generated from the same underlying system, each with a different noise realisation. The Least Squares identification algorithm is applied to each dataset, and the resulting parameter estimates are collected. For every estimated model, the frequency response is computed, and the empirical variance at each frequency is then evaluated. The procedure is summarised below in Algorithm 3:

Algorithm 3 Monte Carlo variance estimation

```
1: Select the system ($(A - C)$), model structure (ARX/ARMAX), dataset length $N$, past
   window size $p$, number of Monte Carlo runs $M$, and frequencies $\omega_k$
2: for $i = 1, \dots, M$ do
3:     Generate a noise realisation
4:     Simulate a dataset of length $N$ using Equation 2-7
5:     Estimate model parameters $\hat{\Theta}_i$ using a Least Squares method.
6:     Compute the frequency response $f_i = f(\hat{\Theta}_i)$
7: end for
8: Compute the sample mean response $\bar{f}(\omega_k) = \frac{1}{M}\sum_i f_i(\omega_k)$
9: Compute the variance $\widehat{\mathrm{var}}[f(\omega_k)] = \frac{1}{M-1}\sum_i \left(f_i(\omega_k) - \bar{f}(\omega_k)\right)^2$
```

Although conceptually straightforward, the Monte Carlo approach is computationally demanding. Each run requires the generation of the dataset, the estimation of the model, and the storage of the results. A reliable variance estimate typically requires a large number of runs, which is feasible in simulation studies but impractical in experimental settings, where repeating an experiment hundreds of times is unrealistic. To overcome this limitation, subsection 4-2-2 introduces an analytical first-order variance approximation. Before introducing the analytical approximation, the results of the Monte Carlo study are presented below to establish a benchmark for comparison.

Results and Discussion

Unless stated otherwise, the following parameters are used:

$$N = 50000, p = 20, \sigma = 1, \lambda = 1, \text{ and } M = 1000.$$

The following paragraphs examine the effect of four factors on the variance convergence behaviour: the noise realisation, the past window parameter p, the noise standard deviation σ, and the number of Monte Carlo runs M.

Effect of noise realisation. Figure 4-3 shows the convergence of the variance estimate for different noise realisations. The trace of the parameter variance, normalised by the number of parameters, remains consistent across all random seeds. This confirms that the convergence behaviour is independent of the specific noise sequence, and therefore a single noise realisation can be fixed for subsequent analyses.

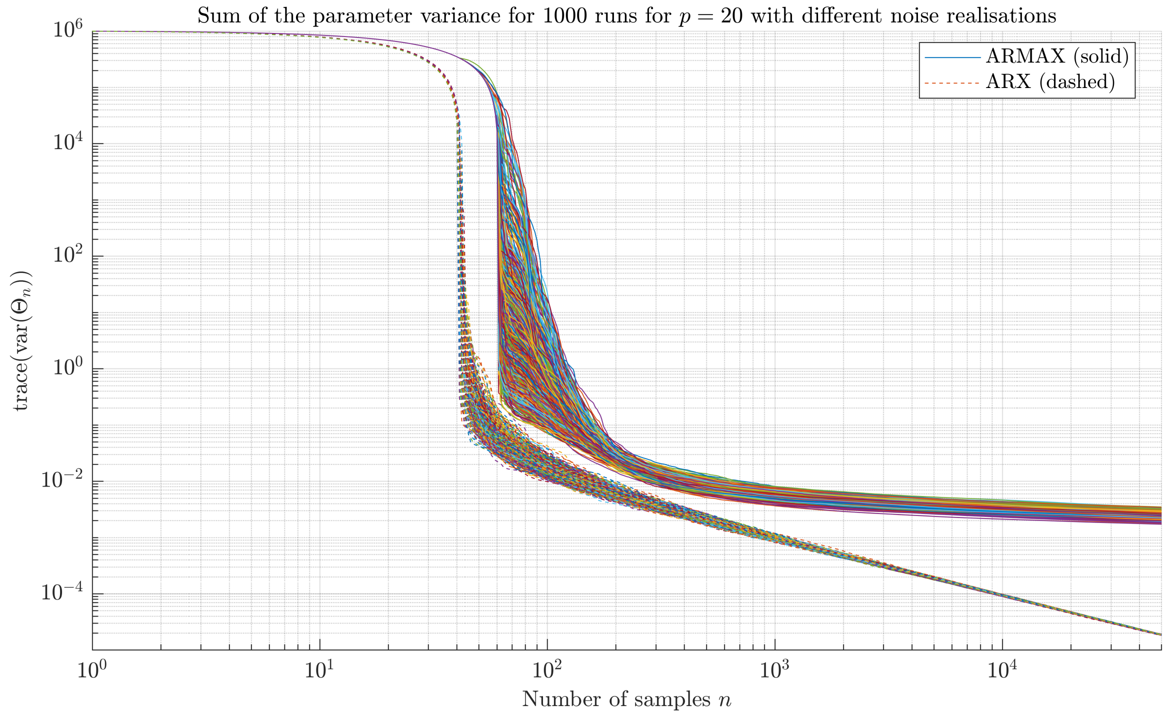
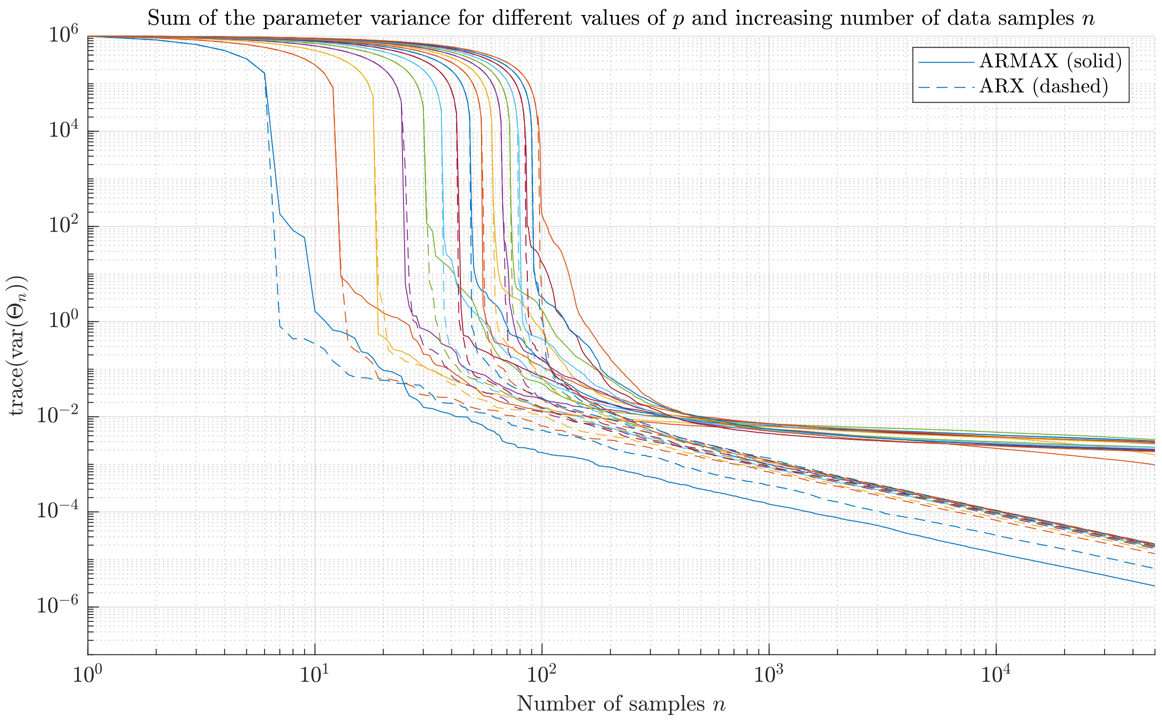
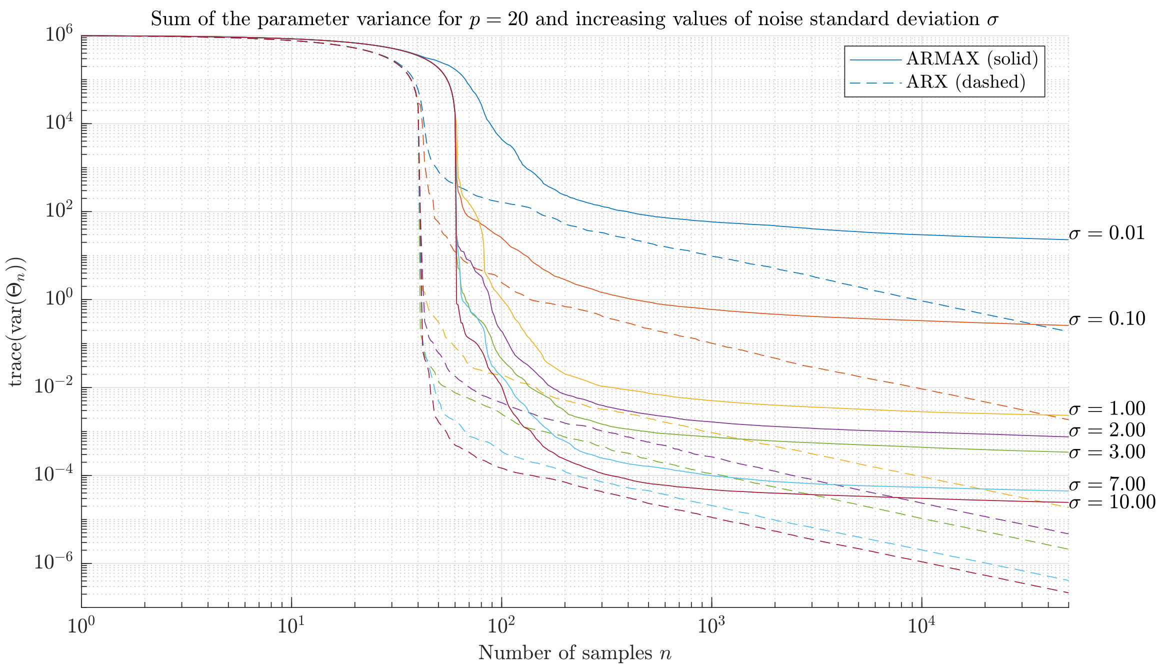


Figure 4-3: Effect of noise realisation on the variance convergence. The convergence behaviour is consistent for 1000 runs per model across different noise realisations for both ARX and ARMAX, so the convergence behaviour is independent of the noise realisation.

Effect of model order p. Figure 4-4 illustrates the influence of the past window size p and the noise standard deviation σ. In Figure 4-4a, the past window size is varied for both ARX and ARMAX. To allow a fair comparison, the total number of parameters is kept equal across models: ARX has $2p$ parameters and ARMAX has $3p$. Therefore, ARX is plotted in increments of three ($p \in \{3, 6, 9, \ldots, 48\}$) and ARMAX in increments of two ($p \in \{2, 4, 6, \ldots, 32\}$). Both models show a steep reduction in variance once the number of data samples exceeds the number of parameters ($N = 6, 12, 18, \ldots, 96$). ARX converges with the expected $1/N$ rate, whereas the ARMAX model follows this trend only when the past window size matches $p = 2$.



(a) Effect of varying the past window size p. The variance drops signifcantly as soon as the number of samples exceeds the number of parameters.



(b) Effect of varying the noise standard deviation σ. Higher σ reduces the variance, because additional noise improves the conditioning of the data matrix.

Figure 4-4: Influence of modelling parameters on the convergence of the variance estimate for ARX and ARMAX models. ARX follows the theoretical $^1/_N$ convergence, while ARMAX only does so when p equals the true order. The stagnation of the empirical variance for the ARMAX model can be attributed to the nonlinear estimation of the residual parameters.

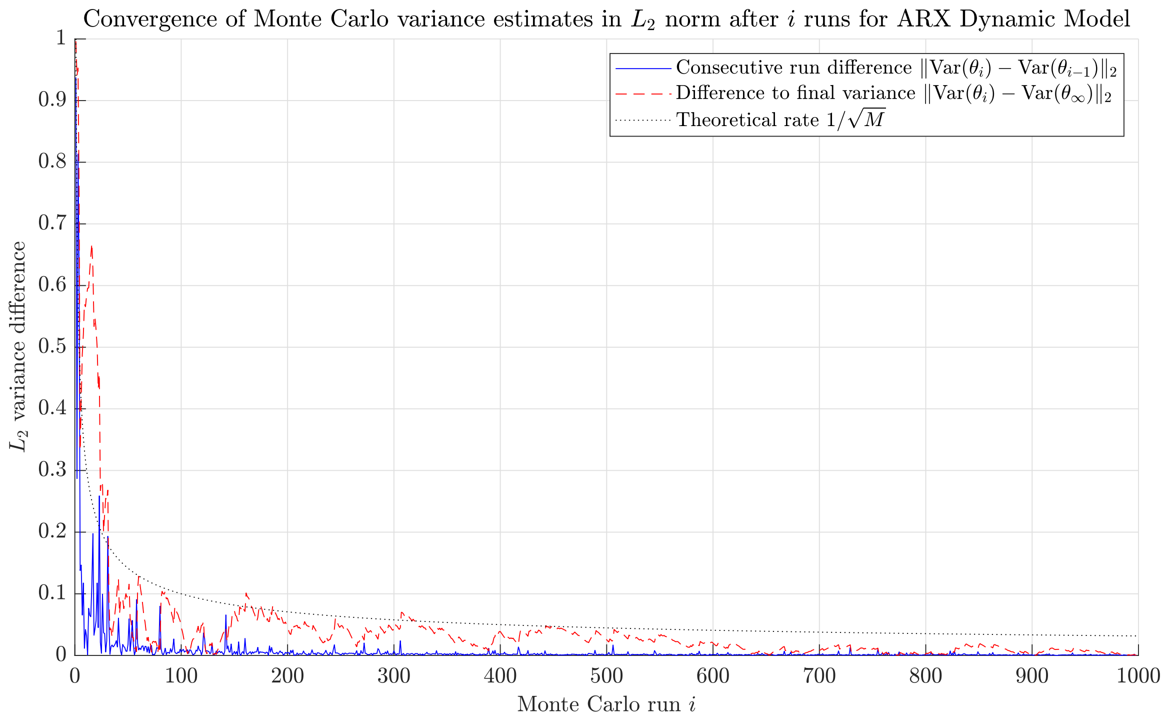
Effect of Noise Standard Deviation σ. Figure 4-4b shows the effect of varying the noise standard deviation $\sigma \in \{0.01, 0.1, 1, 2, 3, 7, 10\}$. Surprisingly, a larger σ results in a lower variance estimate. This can be explained by the parameter variance expression:

$$\text{var}(\Theta|\Phi_N) = \sigma^2(\Phi_N \Phi_N^\top)^{-1}. \tag{4-4}$$

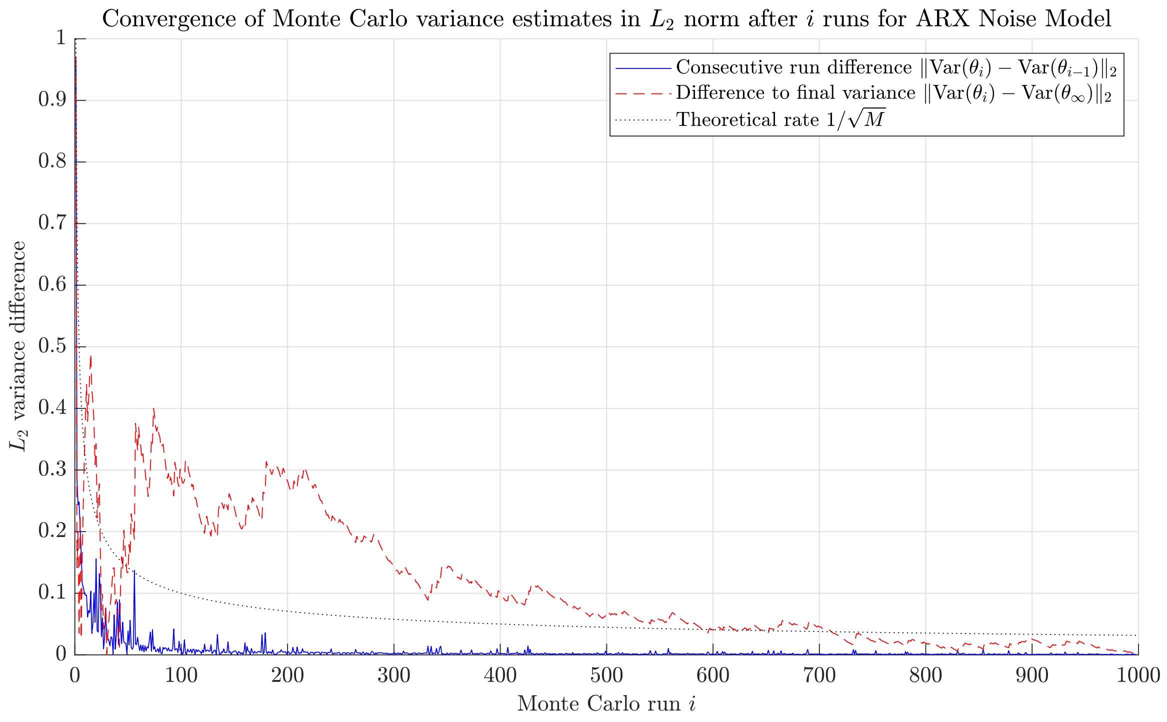
At low noise levels the data matrix becomes nearly collinear, producing an ill-conditioned Φ_N, which increases the variance. Adding noise improves the conditioning of Φ_N, and this effect outweighs the direct scaling with σ^2, leading to the observed decrease for ARX.

Moreover, both plots show that for higher-order ARMAX models the convergence of the variance stagnates. This is likely due to the correlation between the estimated residuals, and the model parameters, as the estimation of the residuals is a nonlinear problem in the parameters itself. A lower bound for this variance could potentially be calculated through the Cramér-Rao lower bound, and the Fisher information matrix.

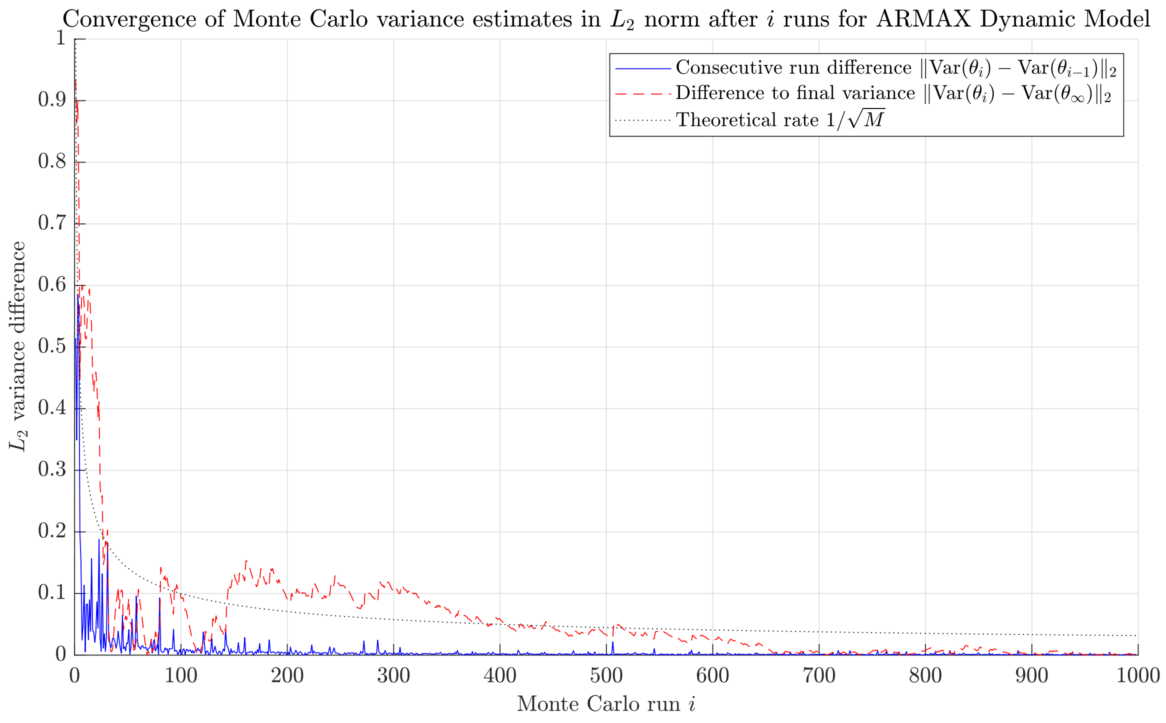
Effect of Monte Carlo Runs M. Finally, Figure 4-5 shows the convergence of the Monte Carlo variance estimate as the number of runs M increases. For small M, the estimate fluctuates with the noise realisation. As M increases, the estimate stabilises around a fixed value, with a convergence rate of approximately $\mathcal{O}(1/\sqrt{M})$. Around $M = 1000$ runs are sufficient for a reliable approximation.



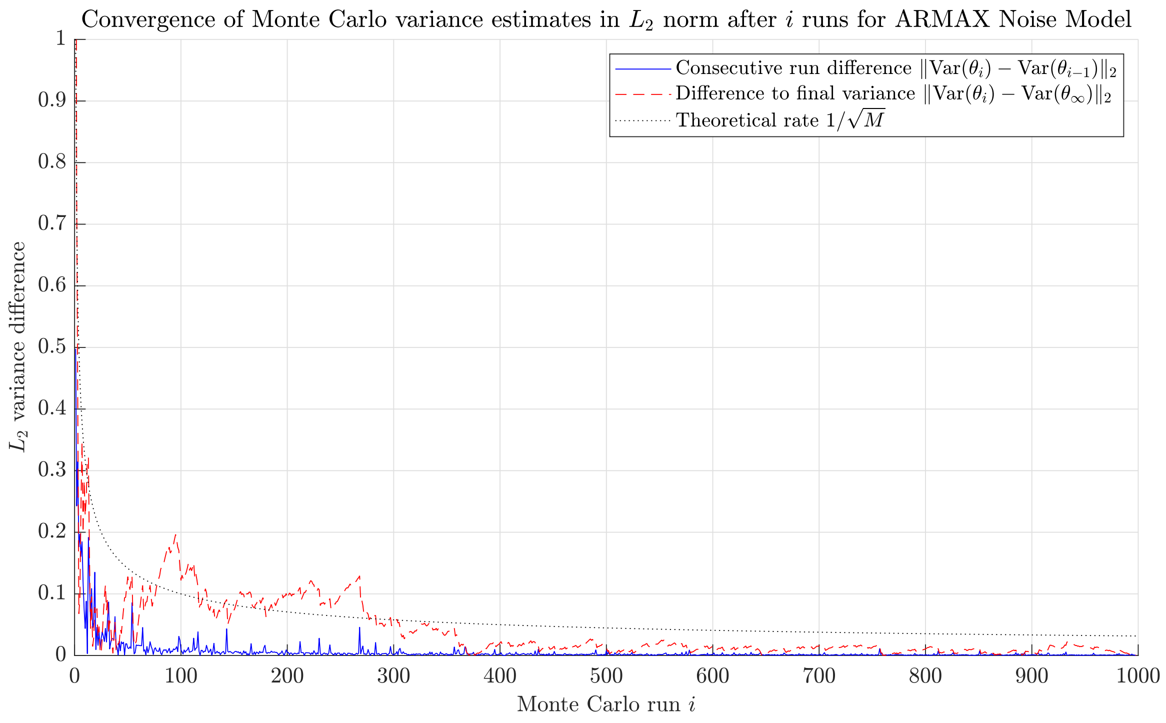
(a) ARX Dynamic Model



(b) ARX Noise Model



(c) ARMAX Dynamic Model



(d) ARMAX Noise Model

Figure 4-5: Influence of the amount of Monte Carlo runs M on the convergence of the Monte Carlo variance estimate for ARX and ARMAX dynamic and noise models.

4-2-2 First-Order Variance Approximation

Monte Carlo simulations provide reliable estimates of parameter uncertainty but are computationally demanding and infeasible in practice. An analytical alternative is the first-order variance approximation, which propagates uncertainty from estimated parameters into the functions of interest (e.g., transfer functions) through a linearisation. This method requires only a single dataset and avoids the need for multiple similar experiments.

The procedure consists of three main steps. First, the covariance of the least-squares parameter estimates is obtained. Second, this covariance is propagated through the transfer function using the Jacobian with respect to the model parameters. Finally, the resulting complex-valued uncertainty is converted into confidence bounds on the magnitude response in linear or decibel scale. The algorithm for the transfer function $G(q, \Theta)$ is summarised in Algorithm 4, the same steps apply to the noise model $H(q, \Theta)$.

Algorithm 4 First-Order Variance Approximation and Confidence Bounds for $G(e^{i\omega})$

```
1: Parameter estimation.
    1. Estimate parameters $\hat{\Theta}$ via least squares.
    2. Compute the residuals: $\hat{E} = Y_f - \hat{\Theta}\Phi$.
    3. Estimate the parameter covariance:
       $$P_\Theta \approx \frac{\hat{E}\hat{E}^T}{N - \dim(\Theta)}(\Phi\Phi^T)^{-1}$$
2: Covariance propagation.
    1. For transfer function $G(q^{-k}, \Theta)$, substitute $q = e^{i\omega}$, and compute Jacobian $J_G(e^{i\omega}, \hat{\Theta})$ w.r.t. the model parameters.
    2. Split into real and imaginary components:
       $$J_G(e^{i\omega}) = \begin{bmatrix} \mathfrak{Re}\{\nabla_\Theta G(e^{i\omega})\} \\ \mathfrak{Im}\{\nabla_\Theta G(e^{i\omega})\} \end{bmatrix} \in \mathbb{C}^{2\times\bar{p}}$$
       where $\bar{p}$ depends on the amount of parameters.
    3. Propagate covariance:
       $$\mathrm{Cov}(G(e^{i\omega})) \approx J_G(e^{i\omega}) P_\Theta J_G(e^{i\omega})^T \in \mathbb{C}^{2\times2}$$
3: Magnitude confidence bounds.
    1. Compute magnitude:
       $$G_{\mathrm{mag}}(\omega) = |G(e^{i\omega})|$$
    2. Derive variance of magnitude:
       $$\sigma^2_{|G|}(\omega) = \frac{\mathfrak{Re}(G)^2\,\mathrm{Cov}_{11} + 2\mathfrak{Re}(G)\mathfrak{Im}(G)\,\mathrm{Cov}_{12} + \mathfrak{Im}(G)^2\,\mathrm{Cov}_{22}}{|G(e^{i\omega})|^2}$$
    3. Standard deviation:
       $$\sigma_{|G|}(\omega) = \sqrt{\sigma^2_{|G|}(\omega)}$$
    4. Convert to decibel scale:
       $$G_{\mathrm{mag,dB}} = 20\log_{10}(G_{\mathrm{mag}})$$
    5. Construct upper/lower bounds of k standard deviations:
       $$20\log_{10}\left(G_{\mathrm{mag}} \pm k\sigma_{|G|}\right)$$
```

Application of First-Order Approximation for ARX and ARMAX Models in Frequency Domain

The derivation of the algorithm relies on the following assumptions:

(a) The estimator $\hat{\Theta}$ is unbiased and concentrated near the true parameter Θ_0.

(b) A first-order Taylor expansion around Θ_0 is valid, such that higher-order terms are negligible.

(c) The data matrix Φ has full row rank so that $(\Phi\Phi^T)^{-1}$ exists.

(d) The residuals are a reasonable estimate for the (unobservable) noise sequence, such that their variance can be used in place of the true noise variance.

So, consider now the least-squares estimator with data matrix Φ_N, future output Y_f, and parameter vector Θ. This estimate minimises:

$$\hat{\Theta} = \arg\min_{\Theta} \|Y_f - \Theta\Phi\|^2 . \tag{4-5}$$

However, the true noise terms are unobservable. The residual is the difference between the observed data and the model's prediction and is used as an estimate of the noise:

$$\hat{E} = Y_f - \hat{\Theta}\Phi. \tag{4-6}$$

Then an unbiased estimator for the parameter covariance is given by [13, 49]:

$$P_\Theta = \frac{\hat{E}\hat{E}^T}{N - dim(\Theta)}(\Phi_N\Phi_N^T)^{-1}, \tag{4-7}$$

where N is the number of samples.

To propagate this uncertainty, let $f(\Theta)$ be a function of interest, such as the dynamic model $G(q, \Theta)$ or noise model $H(q, \Theta)$. A first-order Taylor expansion around Θ_0 gives:

$$\begin{aligned} f(\hat{\Theta}) &\approx f(\Theta_0) + (\hat{\Theta} - \Theta_0)J(\Theta_0)^T, \\ f(\hat{\Theta}) - f(\Theta_0) &\approx (\hat{\Theta} - \Theta_0)J(\Theta_0)^T, \end{aligned} \tag{4-8}$$

where $J(\Theta) = \frac{\partial f}{\partial \Theta}$ is the Jacobian. Then, the covariance of this function can be computed as follows:

$$\begin{aligned} \mathrm{cov}(f(\Theta)) &= \mathrm{E}\left[\left(f(\hat{\Theta}) - \mathrm{E}(f(\hat{\Theta}))\right)^T \left(f(\hat{\Theta}) - \mathrm{E}(f(\hat{\Theta}))\right)\right] \\ &\approx \mathrm{E}\left[\left(f(\hat{\Theta}) - f(\Theta_0)\right)^T \left(f(\hat{\Theta}) - f(\Theta_0)\right)\right]. \end{aligned} \tag{4-9}$$

Plugging in Equation 4-8 results in the Gauss' approximation formula:

$$\mathrm{cov}(f(\Theta)) \approx \mathrm{E}\left[J(\Theta_0)(\hat{\Theta} - \Theta_0)^T(\hat{\Theta} - \Theta_0)J(\Theta_0)^T\right] \tag{4-10}$$

$$\mathrm{cov}(f(\Theta)) \approx J(\Theta_0)P_\Theta J(\Theta_0)^T. \tag{4-11}$$

Since the true parameters Θ_0 are unknown in practice, the Jacobian is evaluated at the estimated parameters $\hat{\Theta}$.

In the case of ARX and ARMAX models, an analytic Jacobian is available. Without direct feedthrough, let us define the parameter vectors as

$$\Theta_{ARX} = \begin{bmatrix} a_p & \cdots & a_1 & b_p & \cdots & b_1 \end{bmatrix}, \tag{4-12}$$

$$\Theta_{ARMAX} = \begin{bmatrix} a_p & \cdots & a_1 & b_p & \cdots & b_1 & c_p & \cdots & c_1 \end{bmatrix}, \tag{4-13}$$

where $\{a_k\}$, $\{b_k\}$, and $\{c_k\}$ are the coefficients of the autoregressive, exogeneous, and moving average, respectively. The corresponding transfer functions for ARX and ARMAX are given by Equation 2-17c and 2-17b

$$G(q, \Theta_{ARX}) = \frac{B(q)}{A(q)}, \qquad H(q, \Theta_{ARX}) = \frac{1}{A(q)}, \tag{4-14}$$

$$G(q, \Theta_{ARMAX}) = \frac{B(q)}{A(q)}, \qquad H(q, \Theta_{ARMAX}) = \frac{C(q)}{A(q)}. \tag{4-15}$$

For $\Theta_{ARX} \in \mathbb{R}^{1\times 2p}$, the gradient of the transfer function $G(q, \Theta_{ARX})$ and $H(q, \Theta_{ARX})$ with respect to the parameters is

$$\nabla_{\Theta_{ARX}} \frac{B(q)}{A(q)} = \begin{bmatrix} -\frac{B(q)}{A(q)^2} q^{-p} & \cdots & -\frac{B(q)}{A(q)^2} q^{-1} & \frac{1}{A(q)} q^{-p} & \cdots & \frac{1}{A(q)} q^{-1} \end{bmatrix}, \tag{4-16}$$

$$\nabla_{\Theta_{ARX}} \frac{1}{A(q)} = \begin{bmatrix} -\frac{1}{A(q)^2} q^{-p} & \cdots & -\frac{1}{A(q)^2} q^{-1} & 0 & \cdots & 0 \end{bmatrix}, \tag{4-17}$$

while for $\Theta_{ARMAX} \in \mathbb{R}^{1\times 3p}$, the gradients are

$$\nabla_{\Theta_{ARMAX}} \frac{B(q)}{A(q)} = \begin{bmatrix} -\frac{B(q)}{A(q)^2} q^{-p} & \cdots & -\frac{B(q)}{A(q)^2} q^{-1} & \frac{1}{A(q)} q^{-p} & \cdots & \frac{1}{A(q)} q^{-1} & 0 & \cdots & 0 \end{bmatrix}, \tag{4-18}$$

$$\nabla_{\Theta_{ARMAX}} \frac{C(q)}{A(q)} = \begin{bmatrix} -\frac{C(q)}{A(q)^2} q^{-p} & \cdots & -\frac{C(q)}{A(q)^2} q^{-1} & 0 & \cdots & 0 & \frac{1}{A(q)} q^{-p} & \cdots & \frac{1}{A(q)} q^{-1} \end{bmatrix}. \tag{4-19}$$

Substituting $q^{-k} = e^{-i\omega k}$ for $k = p, \ldots, 1$ yields frequency-dependent gradients. To handle real-valued covariance propagation, the complex gradient is split into its real and imaginary parts:

$$J_G(e^{i\omega}) = \begin{bmatrix} \mathfrak{Re}\{\nabla_\Theta G(e^{i\omega})\} \\ \mathfrak{Im}\{\nabla_\Theta G(e^{i\omega})\} \end{bmatrix} \in \mathbb{R}^{2\times \dim(\Theta)}. \tag{4-20}$$

Finally, the covariance of the frequency response is propagated as

$$\mathrm{Cov}(G(e^{i\omega})) \approx J_G(e^{i\omega}) P_\Theta J_G(e^{i\omega})^T \in \mathbb{R}^{2\times 2}, \tag{4-21}$$

where P_Θ is the parameter covariance matrix found in Equation 4-7. The derived Jacobians allow the propagation of parameter uncertainty into the frequency domain. By combining P_Θ with the sensitivity of the transfer function to the parameters, this formulation yields frequency-dependent confidence regions for ARX and ARMAX models.

These confidence bounds quantify the robustness of the identification under uncertainty and can be visualised as standard deviation bands in Bode plots. Such bands allow direct comparison between empirical Monte Carlo estimates and the analytical first-order approximation. The method therefore offers a computationally efficient alternative to repeated simulations.

In summary, the first-order variance method systematically translates parameter uncertainty into frequency-domain confidence intervals. This yields confidence bounds for ARX and ARMAX models, avoiding the need for a large amount of Monte Carlo simulations. Next, it is evaluated how well this first-order approximation agrees with the empirical Monte Carlo estimates.

Results and Discussion

The performance of the identified models is evaluated using two frequency-domain metrics: the absolute fit and the relative fit. The absolute fit (Equation 4-22) quantifies the squared deviation between the true and estimated frequency responses, normalised by the total energy of the true system response. It therefore measures the overall accuracy of the estimated model across all frequencies, giving more weight to frequency regions with high system gain.

The relative fit (Equation 4-23), on the other hand, normalises the error with respect to the local magnitude of the true system response. This ensures that low-magnitude regions such as anti-resonances contribute equally to the metric, instead of being overshadowed by the higher-gain dynamics.

$$Fit_{abs} = 1 - \frac{\sum_{\omega\in\Omega}(G_{\text{true}} - G_{\text{MC}})^2}{\sum G_{\text{true}}^2}, \tag{4-22}$$

$$Fit_{rel} = 1 - \frac{\sum_{\omega\in\Omega}\left(\frac{G_{\text{true}}-G_{\text{MC}}}{G_{\text{true}}}\right)^2}{N}. \tag{4-23}$$

The resulting fits for the dynamics and noise models are summarised in Table 4-1 and Table 4-2. Both the absolute and relative fit metrics clearly show that the ARMAX model outperforms the ARX model, particularly for lower model orders p. For example, at $p = 6$, the ARMAX dynamic model achieves an absolute fit of 0.97 compared to 0.77 for ARX, and a relative fit of 0.89 compared to 0.33. This shows the improved accuracy gained from explicitly modelling the noise dynamics. As the model order increases, the difference between ARX and ARMAX becomes smaller, indicating that for sufficiently high p, the ARX model can approximate the system dynamics equally well. However, for $p = 20$ a small bias remains in the dynamical model of ARX (Figure 4-7a) and a huge difference is noticeable in its noise model (Figure 4-7b), while the ARMAX model fits both models perfectly in Figure 4-7c and 4-7d respectively.

Table 4-1: Absolute fit for dynamics and noise models with varying p for $10 \leq \omega \leq 25$. Setup: Model A, $E = 1$, $N = 50000$, $\lambda = 1$, $MC = 1000$.

	ARX		ARMAX	
p	Dynamics	Noise	Dynamics	Noise
6	0.77411	-9.78006	0.96845	-3.34669
10	0.90712	-5.99210	0.99864	0.53148
15	0.99097	-0.37736	0.99999	0.99538
20	0.99909	0.97696	1.00000	1.00000

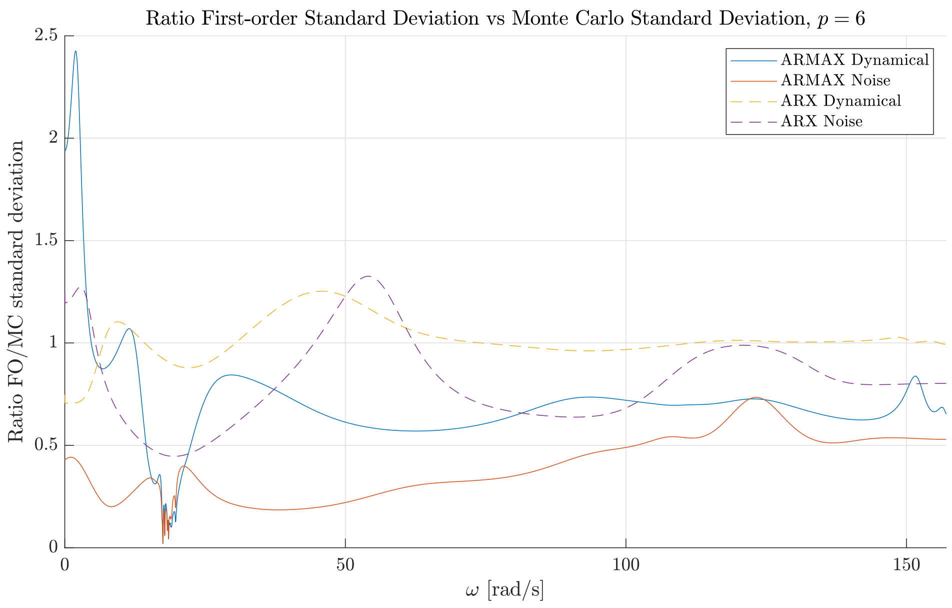
Table 4-2: Relative fit for dynamics and noise models with varying p for $10 \leq \omega \leq 25$. Setup: Model A, $E = 1$, $N = 50000$, $\lambda = 1$, $MC = 1000$.

	ARX		ARMAX	
p	Dynamics	Noise	Dynamics	Noise
6	0.33271	-106.12073	0.88541	-44.06002
10	0.81818	-71.59556	0.99509	-2.85455
15	0.99018	-11.46426	0.99999	0.97452
20	0.99913	0.73880	1.00000	1.00000

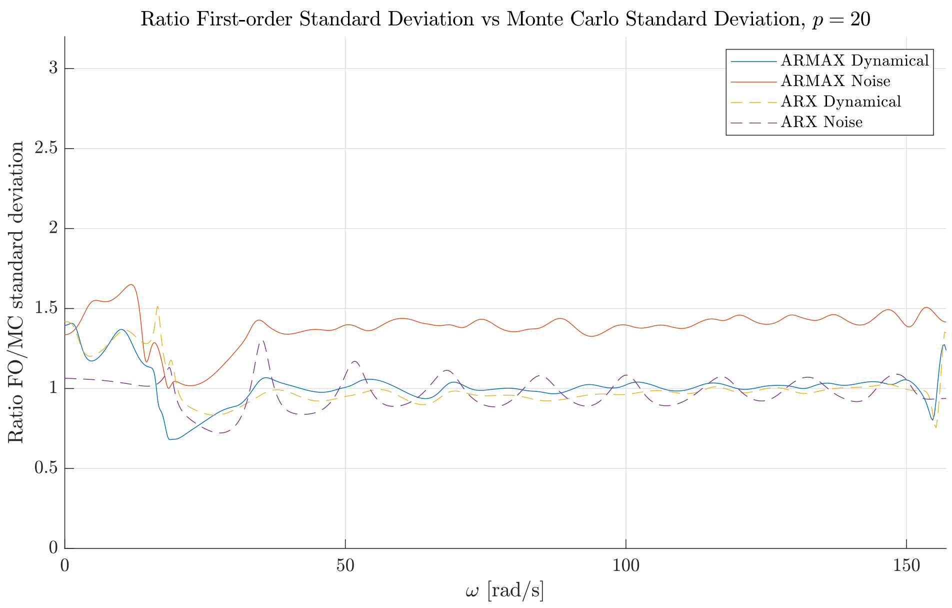
The quality of the variance approximation is further evaluated by comparing the First-Order (FO) variance estimate to a Monte Carlo (MC) based variance estimate. Figure 4-6a and Figure 4-6b illustrate the ratio between the FO and MC standard deviations for $p = 6$ and $p = 20$, respectively. For the lower-order model ($p = 6$), the ratio fluctuates between 0.1 and 2.5, indicating that the first-order approximation does not fully capture the parameter uncertainty, especially at frequencies around the system its resonant and anti-resonant modes. This could be related to the fact that the fit is $<< 1$, meaning that assumption (a) and assumption (d) on page 50 do not hold. In contrast, for the higher-order model ($p = 20$), the ratio remains within the narrower range of 0.6 to 1.6, and tends to fluctuate around unity at higher frequencies. This confirms that the first-order approximation becomes increasingly accurate as the model converges to the true model.

Notably, the ARMAX noise variance exhibits slightly larger deviations and a visible bias in the ratio plots, which suggests that the nonlinear coupling between the noise and dynamic components introduces higher-order effects that are not captured by the linearised variance model. Nevertheless, as shown in Figure 4-7, the visual difference between the FO and MC confidence bounds in the Bode plots is negligible, primarily due to the logarithmic (dB) scaling. This demonstrates that for practical purposes, the system behaviour and uncertainty bounds can still be accurately interpreted using the first-order variance approximation, particularly for models with a high fitting result.

In summary, the ARMAX structure provides a substantial improvement in model fit at lower-orders compared to ARX models. The first-order variance approximation remains a computationally efficient and sufficiently accurate tool for uncertainty assessment in models with a high fitting result.

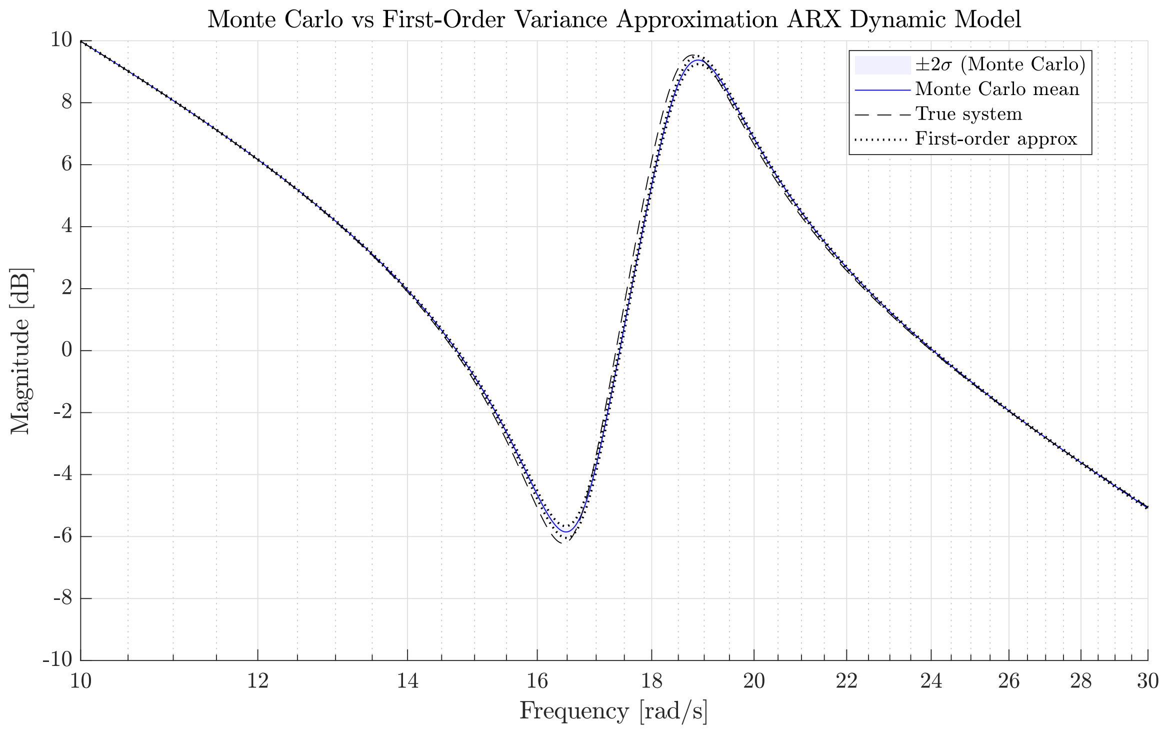


(a) $p = 6$

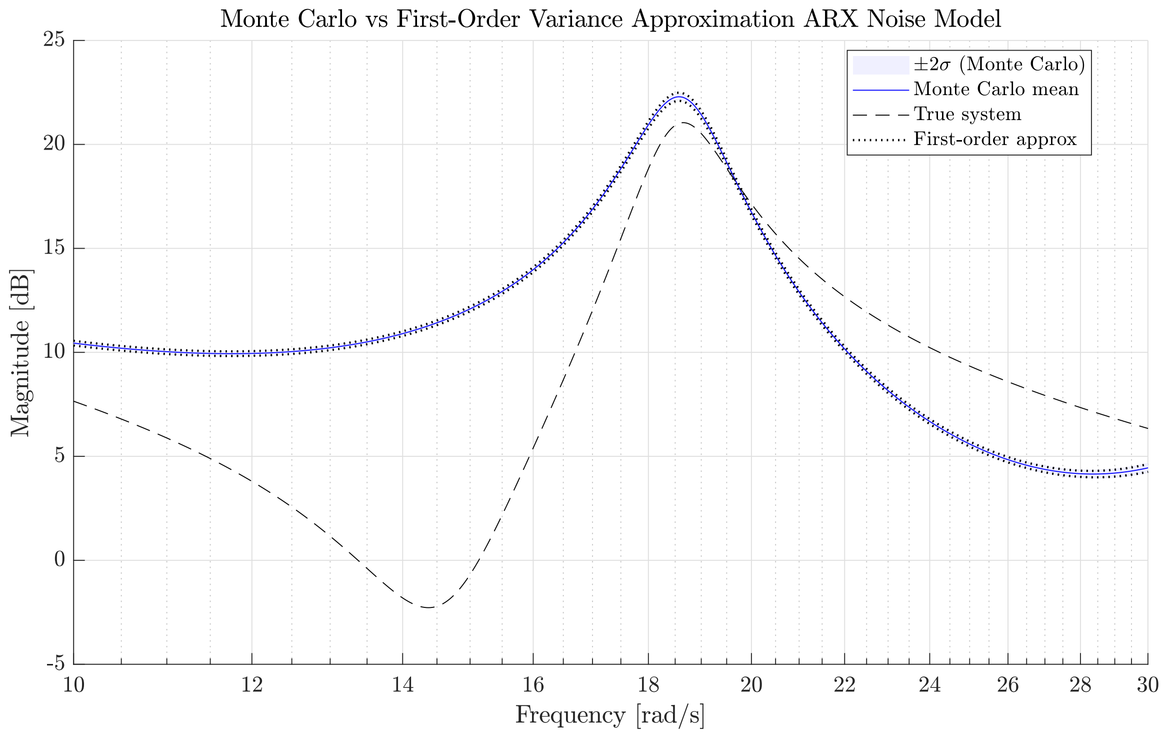


(b) $p = 20$

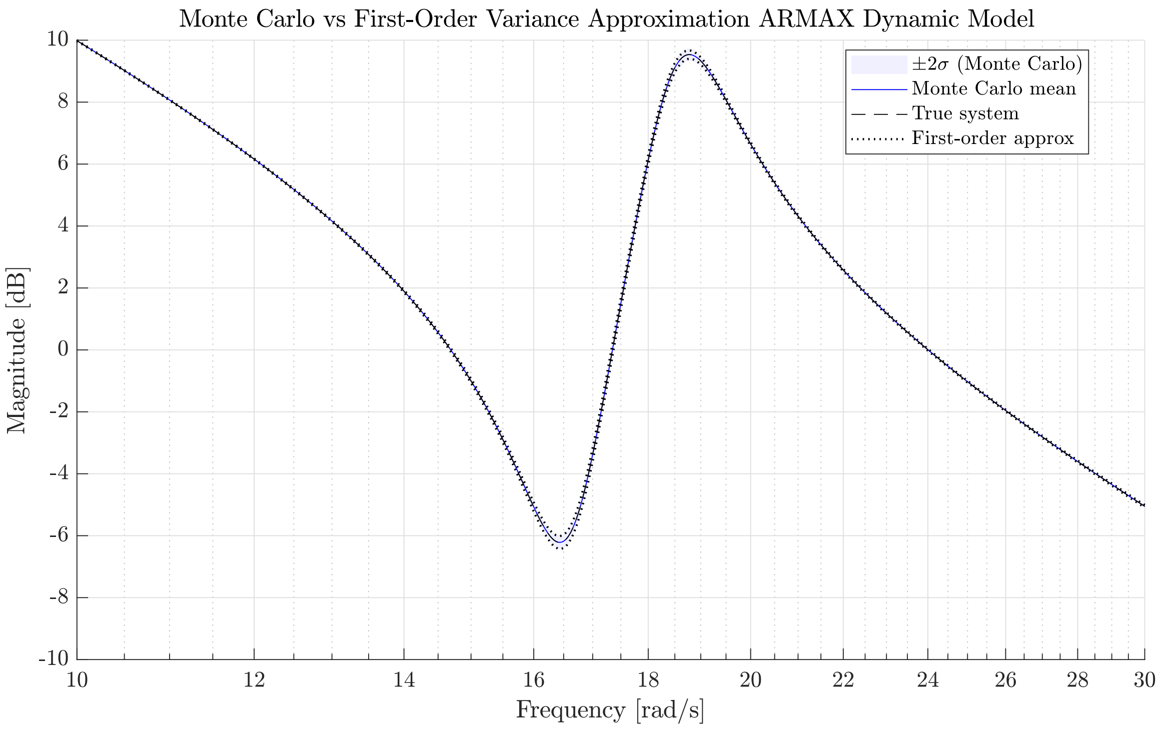
Figure 4-6: Ratio between the First-order variance approximation and the Monte Carlo variance approximation.



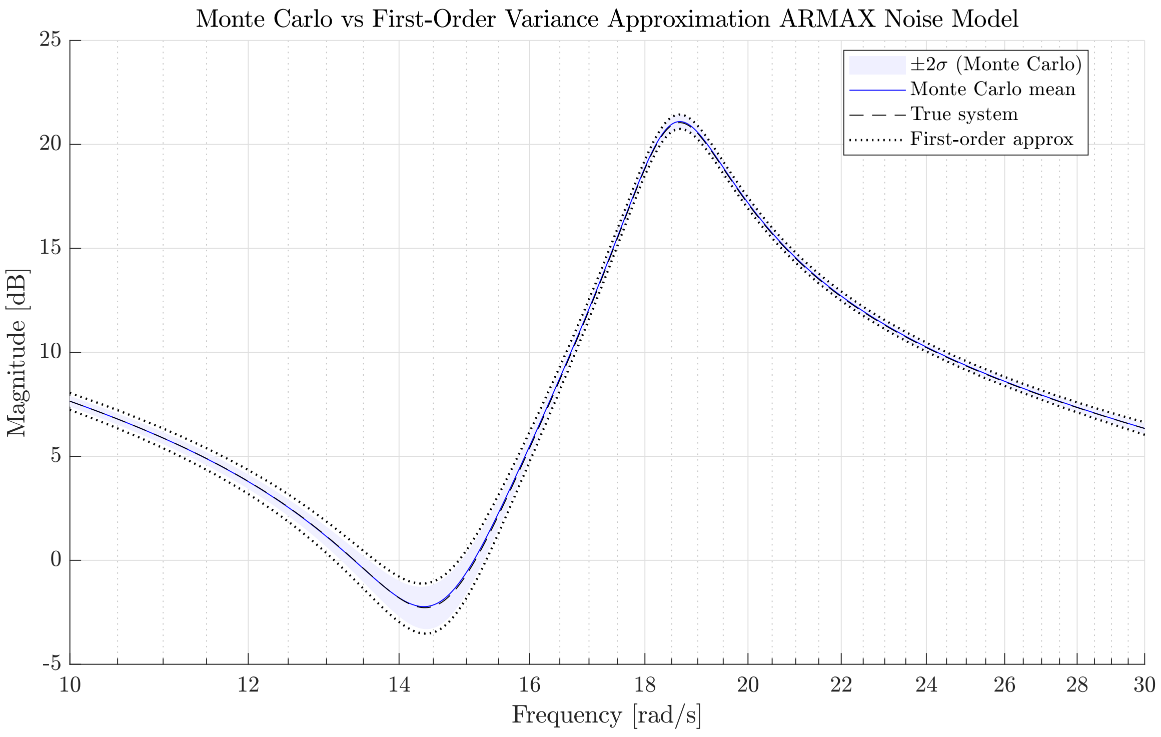
(a) ARX Dynamic Model



(b) ARX Noise Model



(c) ARMAX Dynamic Model



(d) ARMAX Noise Model

Figure 4-7: Bode plots of the Monte Carlo mean for $M = 1000$ for the dynamical and noise model for ARX and ARMAX. The first-order variance estimate and the Monte Carlo variance are visually close to each other.

4-3 Higher-Order ARX Models

Previous empirical results, such as the absolute and relative fits presented in Table 4-1 and Table 4-2, show that higher-order ARX models can reproduce both the system dynamics and the noise dynamic with high accuracy. This observation is somewhat counter-intuitive.

In control theory, poles correspond to resonance frequencies where the denominator of the transfer function approaches zero, leading to large output magnitudes, whereas zeros correspond to anti-resonances, where the numerator becomes zero and the output is attenuated. Since the ARX noise model includes only poles, it seems paradoxical that it can reproduce the anti-resonance behaviour, which is typically associated with zeros.

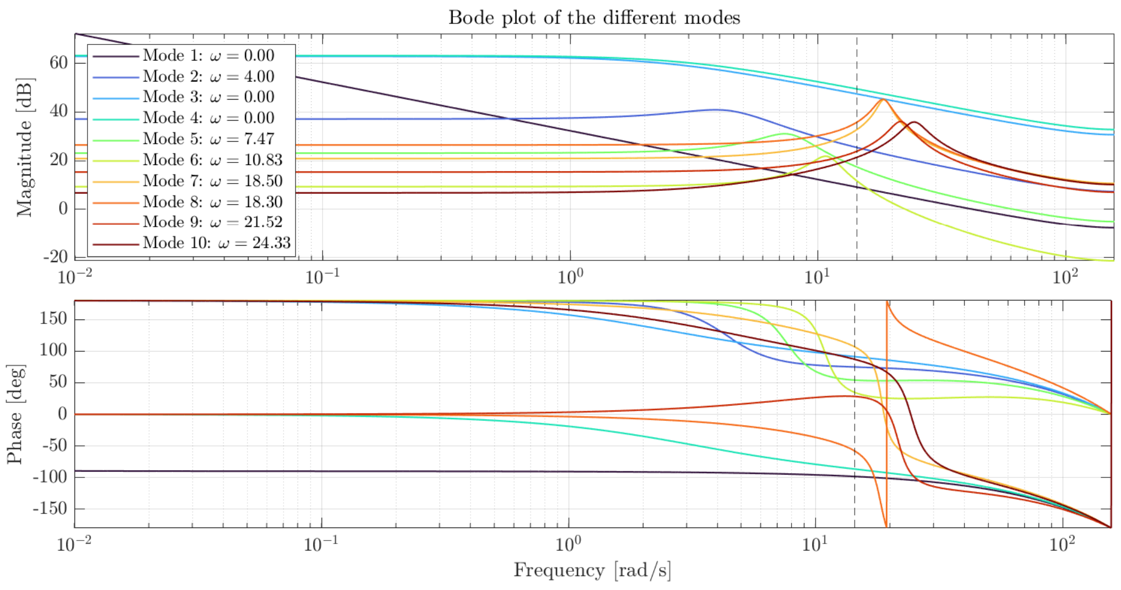
This phenomenon can be understood by considering that the frequency response of any linear system can be expressed as a sum of modal contributions:

$$G(e^{i\omega}) = \sum_{\text{modes}} G_{\text{mode}}(e^{i\omega}). \tag{4-24}$$

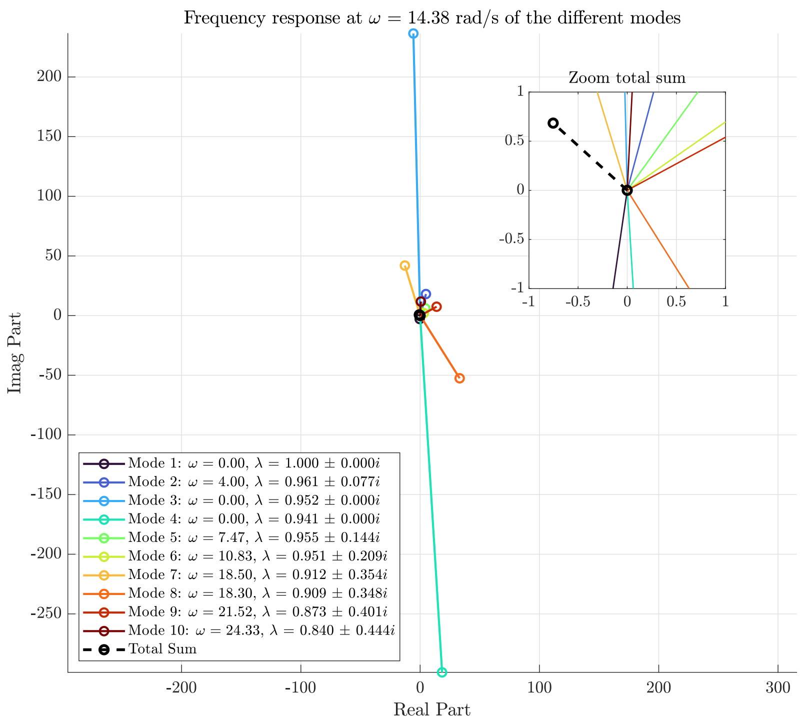
Each mode contributes a complex-valued vector in the frequency domain. At certain frequencies, particularly near anti-resonances, these modal vectors may have similar magnitudes but opposite directions. Their vector sum therefore partially cancels, resulting in a small overall response amplitude. In the magnitude Bode plot, where only $|G(e^{i\omega})|$ is shown, this destructive interference appears as an anti-resonance.

This principle can be visualised by decomposing a high-order ARX model into its modal responses. An ARX model with $p = 100$ was fitted to a dataset of 5×10^4 samples. Figure 4-8a shows the Bode magnitude plots of the first ten modes with $\omega < 25$ rad/s. Two pairs of modes (3,4) and (7,8) exhibit nearly identical magnitudes but a phase difference of approximately 180°. As a result, their contributions cancel each other in the total response, generating a local minimum in the magnitude plot.

To further illustrate this effect, Figure 4-8b shows the complex-plane representation of the modal vectors at the anti-resonance frequency $\omega = 14.38$ rad/s. Each mode contributes a complex-valued response vector whose direction and length vary with frequency. At the anti-resonance, vectors of similar magnitude but opposing directions nearly cancel, resulting in a small resultant vector (shown in black). This cancellation produces the dip in the magnitude response of the noise model, as observed in Figure 4-1. Hence, the anti-resonance behaviour that is typically attributed to the presence of zeros instead arises from cancellations among the modal components.



(a) Bode plots



(b) Frequency response vectors

Figure 4-8: Modal decomposition of a higher-order ARX model illustrating how anti-resonance behaviour arises from modal cancellations. (a) Bode magnitude plot of individual modal contributions, showing pairs of modes with similar magnitudes but opposite phases. (b) Complex-plane representation of modal vectors at the anti-resonance frequency $\omega = 14.38$ rad/s, where opposing modal vectors nearly cancel, resulting in a small resultant vector corresponding to the dip in the magnitude response.

Theoretical explanation

The empirical observations discussed above are well supported by theoretical results in the system identification literature. It has been established that sufficiently high-order ARX models can approximate both the deterministic system dynamics and the stochastic noise model with arbitrary accuracy. However, the practical use of such models depends strongly on their pole-zero configuration. If the estimated pole-zero map shows cancellations, this suggests that the dominant system behaviour can be represented by a lower-order model. When increasing the ARX order leads to additional poles that cancel zeros, these terms describe only the noise dynamics rather than the physical dynamics. In such situations, model structures that explicitly describe the noise, such as ARMAX, or Box-Jenkins models, are generally more suitable. [28]

The asymptotic behaviour of high-order ARX models has been analysed by Ljung and Wahlberg [30], who showed that least-squares estimates converge to the true dynamics and noise spectra as the model order increases. Subsequent work by Wahlberg [58] demonstrated that the asymptotic variance and distribution of these estimators remain well defined, thereby confirming the statistical validity of using high-order ARX structures. Similarly, Galrinho [12] proved that, as the model order grows, ARX models can approximate the more general Box-Jenkins representation to any desired degree of accuracy.

Time-series interpretation

Another, more intuitive, explanation can be obtained from time-series theory. Any stationary autoregressive process of order p can be expressed as an infinite moving-average process, and vice versa, provided that certain stability conditions are satisfied [20]. For example, consider the AR(1) process:

$$\begin{aligned} y_k &= a_1 y_{k-1} + e_k \\ &= a_1(a_1 y_{k-2} + e_{k-1}) + e_k \\ &= a_1^2 y_{k-3} + a_1 e_{k-1} + a_1^2 e_{k-2} + e_k \\ &= \dots . \end{aligned} \tag{4-25}$$

if $|a_1| < 1$ the powers of a_1 decay exponentially, yielding the following MA(∞) representation:

$$y_k = e_k + a_1 e_{k-1} + a_1^2 e_{k-2} + a_1^3 e_{k-3} + \dots . \tag{4-26}$$

A similar result holds for moving-average processes, which can be expressed as an infinite autoregressive representation under appropriate invertibility conditions. This relationship explains why high-order ARX models can approximate coloured noise dynamics as well.

In conclusion, despite the theoretical guarantees for high-order ARX models, they present several practical limitations. As the number of parameters increases, the estimation variance and computational load increases. Moreover, larger datasets are required to ensure reliable convergence, and the increased model complexity makes the estimation less adaptable to time-varying system dynamics due to a potentially overfitting model.

4-4 Comparison Between ARX- and ARMAX-based Subspace Predictive Control

The previous analysis demonstrated that while high-order ARX models can accurately reproduce both system dynamics and noise behaviour, lower-order ARMAX structures provide a more compact and interpretable representation by explicitly modelling the stochastic component. To examine how these modelling differences influence closed-loop behaviour, both model structures are now embedded within the Subspace Predictive Control (SPC) framework. This section evaluates their ability to track reference trajectories under stochastic disturbances and investigates how the underlying model structure affects overall control performance.

Control Metrics

The control performance is quantified using three standard performance criteria widely employed in control literature [26, 55]:

$$ISE = \frac{1}{N} \sum_{k=T_{ini}}^{t_{max}} \|y_k - r_k\|_2^2, \tag{4-27}$$

$$IAE = \frac{1}{N} \sum_{k=T_{ini}}^{t_{max}} \|y_k - r_k\|_1, \tag{4-28}$$

$$InEn = \frac{1}{N} \sum_{k=T_{ini}}^{t_{max}} \|u_k\|_2^2. \tag{4-29}$$

The integral squared error (ISE) penalises large deviations quadratically and is therefore particularly sensitive to overshoot and oscillatory behaviour. The integral absolute error (IAE) reflects the overall tracking quality, ans is less affected by short transients but representative of steady-state accuracy. Finally, the input energy ($InEn$) quantifies the total control effort required to achieve the desired tracking and is directly related to actuator activity and robustness against noise. These metrics provide insights into both tracking quality and control efficiency.

Parameter Settings and Reference Tracking

All simulations use the same controller structure, initial data, and tuning parameters to ensure a fair comparison between the models. The training dataset consists of 5000 samples, with the noise $E \sim \mathcal{N}(0,1)$ and the input signal $U \sim \mathcal{N}(0,20)$. The weighting matrices are set to $Q_y = 1000I$ and $R_u = I$, and the control input is constrained to ± 200 with no explicit output limits. A prediction horizon of $N_p = 25$ and a forgetting factor of $\lambda = 1$ are selected.

At each control step, the quadratic optimisation problem from Equation 2-84 is solved using MATLABs `quadprog` solver with the 'interior-point-convex' algorithm. To evaluate reference tracking performance across different types of waves and frequencies, the controller is tested on the following reference trajectories:

1. A square wave of frequency 0.5 Hz for time t: $0 \leq t < 10$
2. A sinusoidal wave of frequency 1.0 Hz for time t: $10 \leq t < 20$
3. A sinusoidal wave of frequency 2.61 Hz, close to the anti-resonance, for time t: $20 \leq t < 30$
4. A sawtooth wave of frequency 0.5 Hz for time t: $30 \leq t < 40$

Results and Discussion

The quantitative results for the different reference trajectories and model orders from a single run are summarised in Table 4-3 to 4-6. Comparable outcomes are obtained from a Monte Carlo simulation of 50 runs, shown in Figure 4-9, where the performance metrics were computed across the four reference signals for $p \in \{6, 10, 15, 20\}$. Across all tested cases, the ARMAX-based SPC controller consistently outperforms the ARX-based version for lower model orders ($p =$ 6-10). This improvement is most visible near the anti-resonance at 2.61 Hz, where the ARMAX controller achieves up to 50% lower ISE, IAE, and InEn values. The enhanced tracking accuracy is accompanied by a significant reduction in input energy, indicating that the improved noise modelling of the ARMAX structure results in more efficient and smoother actuator behaviour.

In contrast, the ARX-based controller tends to overreact to stochastic disturbances, resulting in higher control activity and less energy-efficient behaviour. This finding is supported by the power spectral density of the control input (Figure 4-10), which shows that the ARX model exhibits higher spectral power at high frequencies compared to the ARMAX model. Because the ARX structure lacks an explicit noise model, measurement disturbances directly influence the predicted output, causing the controller to interpret the stochastic fluctuations as dynamic behaviour. The ARMAX formulation, on the other hand, incorporates a noise filter through the $C(q)$ polynomial, which attenuates these effects and therefore produces smoother control actions for lower-order models.

As the model order increases ($p =$ 15-20), the performance gap between ARX and ARMAX models becomes negligible across all performance metrics. This observation aligns with the frequency-domain analysis presented earlier, where the absolute and relative fits (Table 4-1-4-2) indicated nearly identical model accuracy for high-order systems. At these orders, the deterministic component of the model already captures the dominant dynamics, resulting in comparable controller performance for both formulations.

Table 4-3: Performance comparison between ARX and ARMAX models for varying past horizon p for model A, with a square wave of frequency 0.5Hz for $0 \leq t < 10$. The right block shows the relative improvement factor > 1 indicating better performance of ARMAX.

	ARX			ARMAX			ARX/ARMAX		
p	ISE	IAE	InEn	ISE	IAE	InEn	ISE	IAE	InEn
6	12190	46.2	11192	10921	37.5	8441	1.12	1.23	1.33
10	10993	37.2	7816	10786	35.4	7037	1.02	1.05	1.11
15	10765	35.1	6966	10692	34.7	6913	1.01	1.01	1.01
20	10692	34.7	6907	10715	34.6	6927	1.00	1.00	1.00

Table 4-4: Performance comparison between ARX and ARMAX models for varying past horizon p for model A, with a sinusoidal wave of frequency 1Hz for $10 \leq t < 20$. The right block shows the relative improvement factor > 1 indicating better performance of ARMAX.

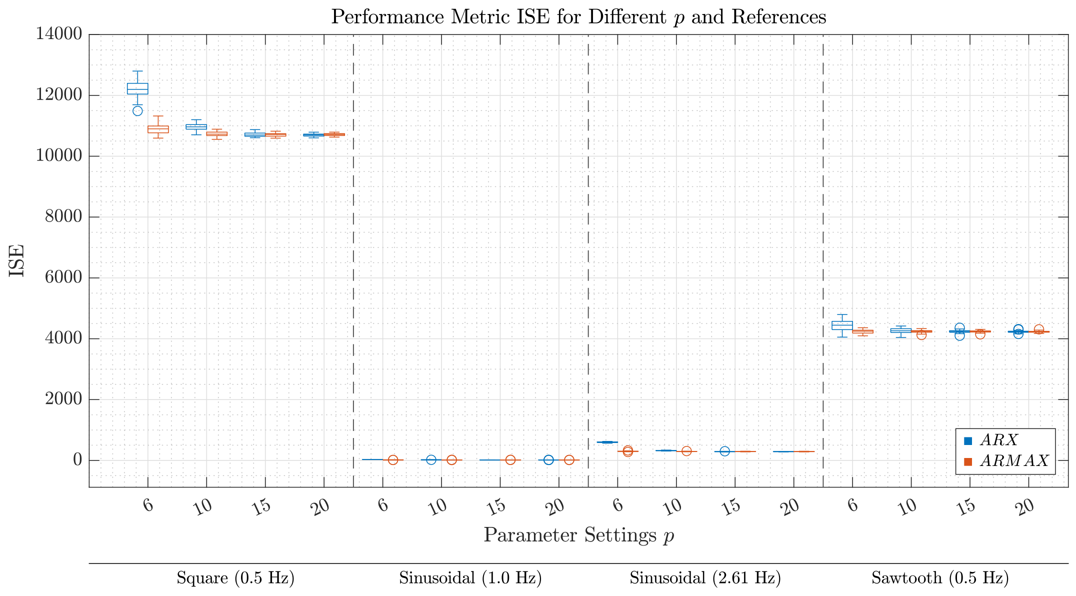
p	ARX			ARMAX			ARX/ARMAX		
	ISE	IAE	InEn	ISE	IAE	InEn	ISE	IAE	InEn
6	30.0	4.34	4652	18.5	3.65	1142	1.62	1.19	4.08
10	20.1	3.70	1264	16.3	3.52	315	1.23	1.05	4.02
15	16.9	3.57	418	15.3	3.43	196	1.11	1.04	2.13
20	15.6	3.44	215	15.1	3.41	155	1.03	1.01	1.39

Table 4-5: Performance comparison between ARX and ARMAX models for varying past horizon p for model A, with a sinusoidal wave of frequency 2.61Hz for $20 \leq t < 30$. The right block shows the relative improvement factor > 1 indicating better performance of ARMAX.

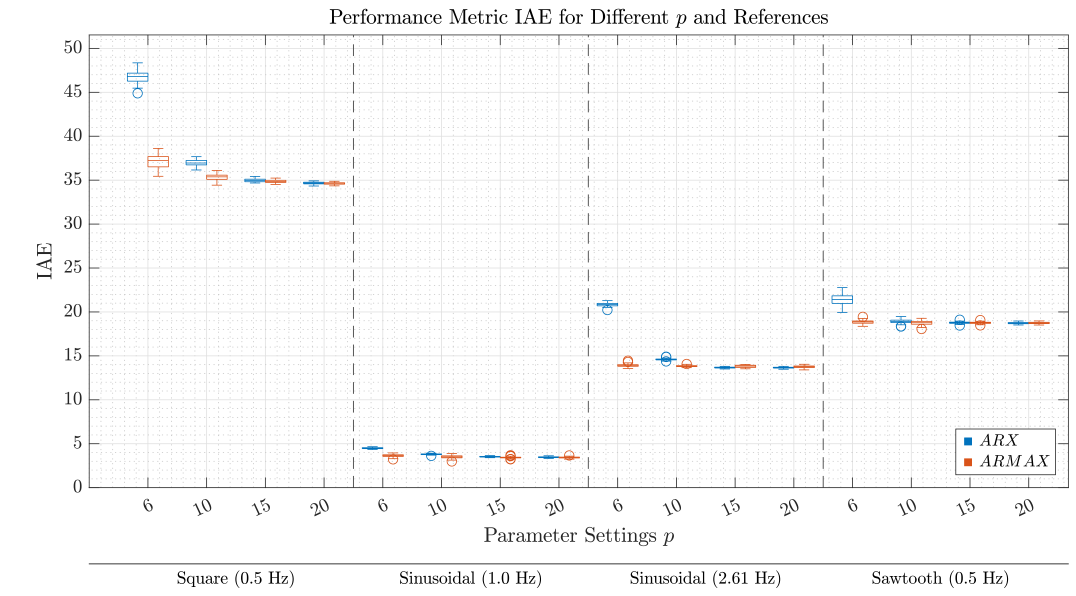
p	ARX			ARMAX			ARX/ARMAX		
	ISE	IAE	InEn	ISE	IAE	InEn	ISE	IAE	InEn
6	558	20.4	9516	286	13.8	5554	1.95	1.48	1.71
10	313	14.5	5491	282	13.7	4779	1.11	1.06	1.15
15	277	13.6	4793	285	13.7	4699	0.97	0.99	1.02
20	279	13.5	4671	285	13.7	4596	0.98	0.99	1.02

Table 4-6: Performance comparison between ARX and ARMAX models for varying past horizon p for model A, with a sawtooth wave of frequency 0.5Hz for $30 \leq t < 40$. The right block shows the relative improvement factor > 1 indicating better performance of ARMAX.

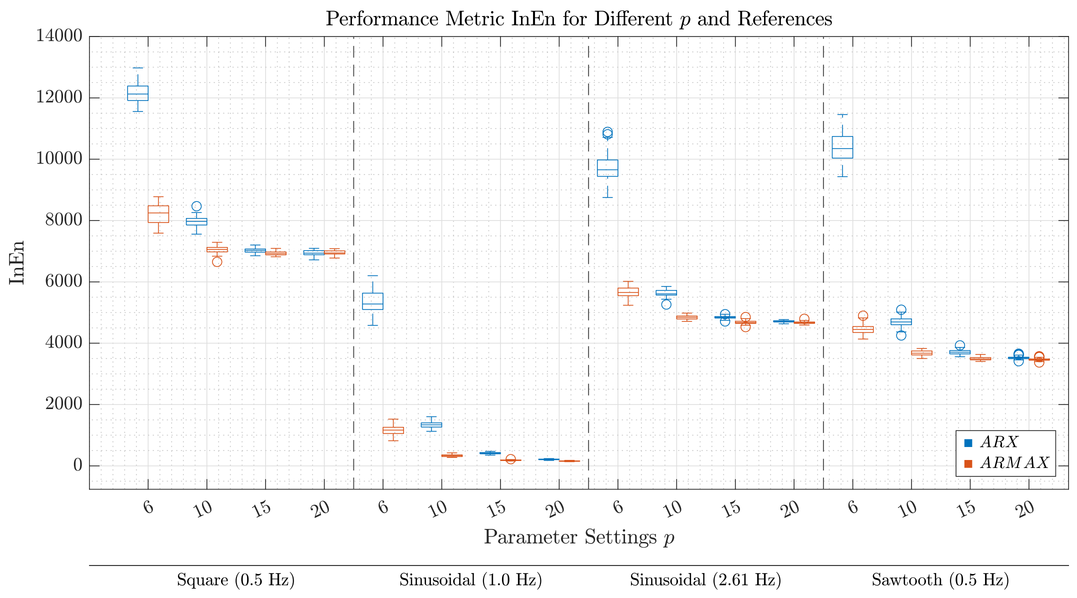
p	ARX			ARMAX			ARX/ARMAX		
	ISE	IAE	InEn	ISE	IAE	InEn	ISE	IAE	InEn
6	4413	21.4	10400	4254	18.8	4509	1.04	1.13	2.31
10	4271	19.0	4695	4240	19.1	3839	1.01	0.99	1.22
15	4262	18.9	3870	4236	18.9	3598	1.00	1.00	1.08
20	4245	18.9	3626	4237	18.9	3539	1.00	1.00	1.03


Performance Metric ISE for Different p and References
ISE
14000
12000
10000
8000
6000
4000
2000
0
6
10
15
20
Parameter Settings p
Square (0.5 Hz)
Sinusoidal (1.0 Hz)
Sinusoidal (2.61 Hz)
Sawtooth (0.5 Hz)
ARX
ARMAX

(a) ISE


Performance Metric IAE for Different p and References
IAE
50
45
40
35
30
25
20
15
10
5
0
6
10
15
20
Parameter Settings p
Square (0.5 Hz)
Sinusoidal (1.0 Hz)
Sinusoidal (2.61 Hz)
Sawtooth (0.5 Hz)
ARX
ARMAX

(b) IAE



(c) InEn

Figure 4-9: Boxplots performance metrics ISE, IAE and InEn for several values of p and different reference waves.

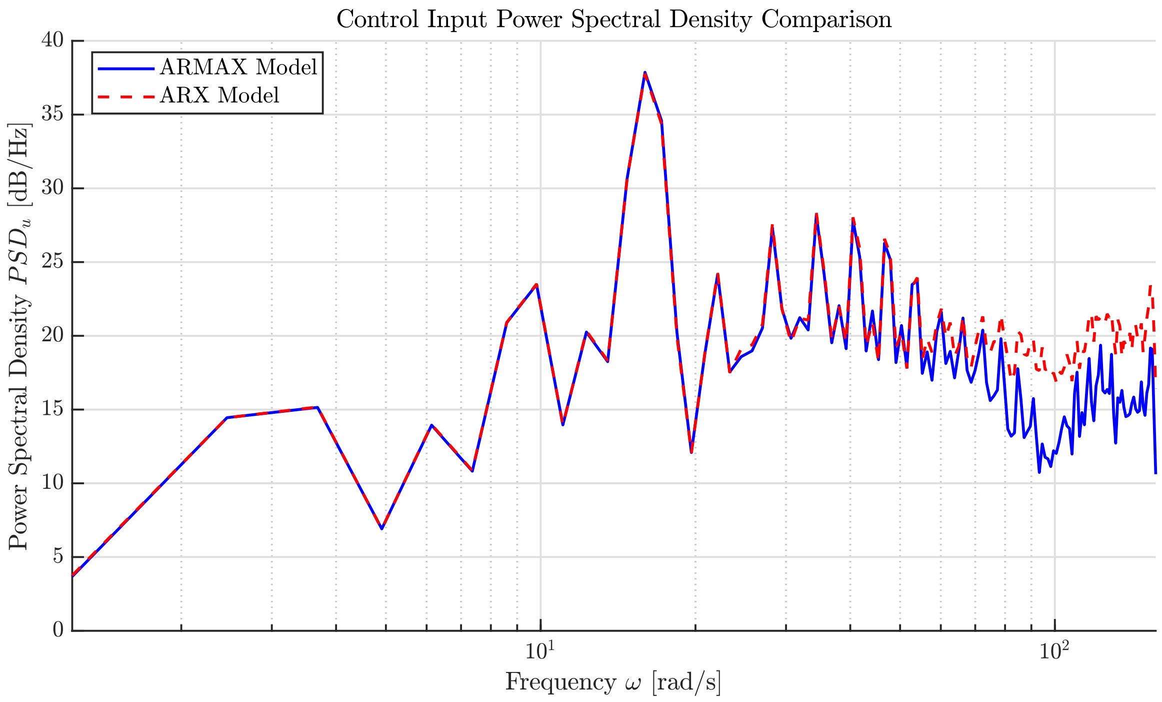
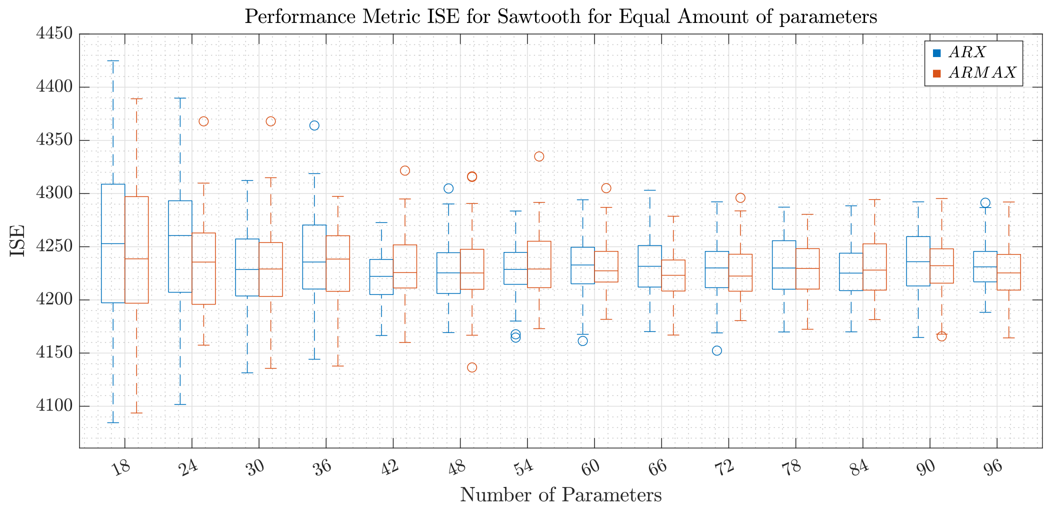


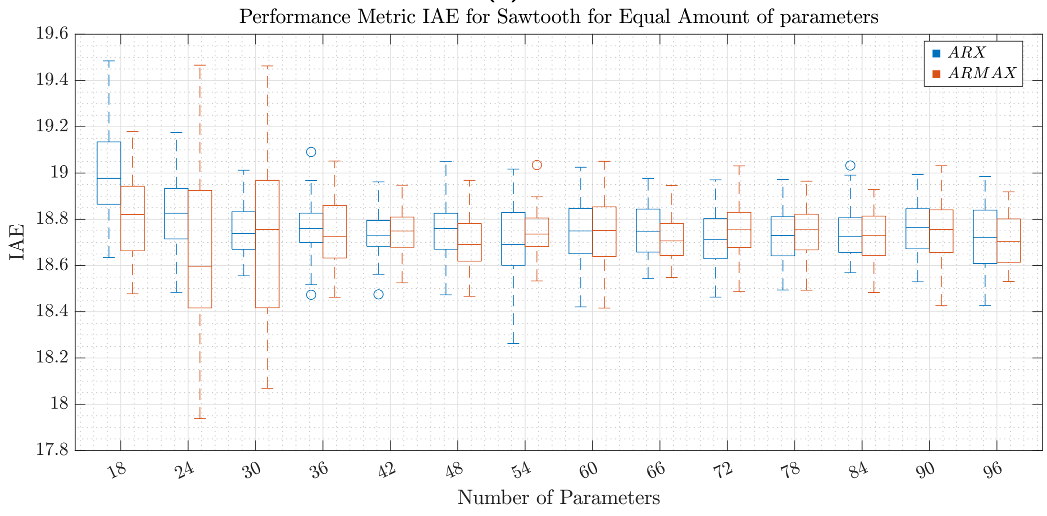
Figure 4-10: The power spectral density of the input signal for ARX and ARMAX for the parameters $p = 10$ and $N_p = 25$. For lower-orders p the two methods differ significantly for higher frequencies, while for higher-orders p, e.g. $p = 25$, the two lines coincide.

Interestingly, Table 4-3 to 4-6 and Figure 4-9 suggest that a slightly higher-order ARX model (e.g., $p = 10$) yields performance comparable to a lower-order ARMAX model ($p = 6$). This implies that control performance may depend primarily on the total number of parameters rather than the specific model structure. To investigate this, Figure 4-11 compares ARX and ARMAX models for the sawtooth reference, keeping the total number of parameters equal. For ARX, this corresponds to $p = {}^{\#\text{parameters}}/_{2}$, whereas for ARMAX, $p = {}^{\#\text{parameters}}/_{3}$. The figure shows that at a lower number of parameter, ARMAX still outperforms ARX, but their performance quickly converges as the number of parameters increases. Because the computational load of the identification step scales quadratically with the number of parameters for both ARX and ARMAX, there is no significant computational advantage in choosing one over the other. Similar trends were observed for the other reference signals, these are available in the Appendix A-1.

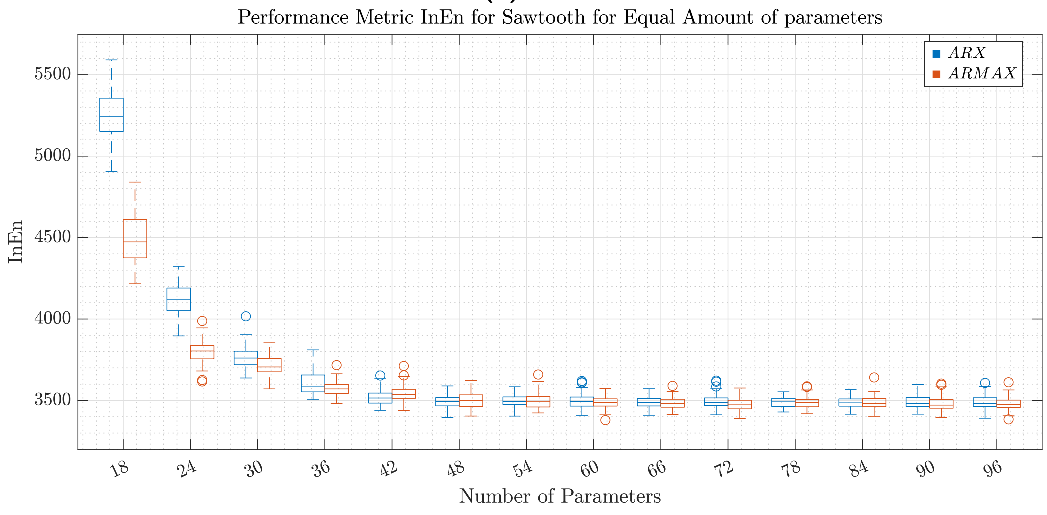
In summary, the SPC results demonstrate that explicit noise modelling improves closed-loop performance for low-order models. Although both ARX and ARMAX converge in performance as the number of parameters increases, the ARMAX model achieves comparable results with a smaller model order p, improving interpretability. Since the computational load scales quadratically with the number of parameters for both models (Table 3-3), this reduced model order p advantage does not translate into reduced computational cost. The next section investigates the controller behaviour under additional external noise disturbances.



(a) ISE



(b) IAE



(c) InEn

Figure 4-11: Boxplots of ISE, IAE, and InEn for 50 Monte Carlo runs for ARX and ARMAX with the same number of parameters for a Sawtooth reference.

Additional Coloured Output Disturbance

To further investigate the robustness of the controllers under stochastic conditions, an additional additive disturbance is introduced at the output side of the system. The weighting matrices are set to $Q_y = 1000I$ and $R_u = 1I$. The control input is limited to ± 200, with no explicit output constraints. A prediction horizon of $N_p = 25$ and a forgetting factor $\lambda = 0.9998$ is selected, as the noise dynamics change during control operation. This modification allows evaluation of how well the ARX- and ARMAX-based Subspace Predictive Control schemes handle unmodelled dynamics and coloured measurement noise.

$$y_k = Cx_k + Du_k + e_k + d_k, \tag{4-30}$$

$$x_{k+1} = Ax_k + Bu_k + Ke_k, \tag{4-31}$$

where d_k represents an external disturbance generated by a coloured noise process. Specifically, the disturbance is obtained by filtering a zero-mean Gaussian white noise sequence through a second-order filter of the form

$$d_k = \frac{1 + 0.8q^{-1} - 0.6q^{-2}}{1}\tilde{e}_k, \tag{4-32}$$

where $(w_k \sim \mathcal{N}(0, 3^2))$ denotes a white noise sequence. This filtering introduces temporal correlation in the disturbance, resulting in an extra coloured noise process.

The performance comparison for different model orders and reference trajectories is summarised in Table 4-7 to 4-10. The ARMAX-based SPC controller demonstrates greater robustness to added disturbance, particularly for lower model orders ($p =$ 6-10).

This improvement follows again from the explicit noise modelling via the $C(q^{-1})$ polynomial, which enables the ARMAX model to better predict the coloured component of the disturbance. In contrast, the ARX-based controller shows a significant increase in both tracking error and control effort, as it attributes part of the stochastic variation to the deterministic dynamics and responds to the noise through increased control effort. Even in higher-order models, the difference in InEn ranges from 20% to 282%.

However, for the tracking metrics ISE and IAE, the difference between ARX and ARMAX again diminishes at higher model orders ($p =$ 15-20), consistent with the earlier observation that sufficiently high-order ARX models can approximate coloured noise through increased model complexity.

Table 4-7: Performance comparison between ARX and ARMAX models with an additional disturbance d for varying past horizon p for model A, with a square wave of frequency 0.5Hz for $0 \leq t < 10$. The right block shows the relative improvement factor > 1 indicating better performance of ARMAX.

	ARX			ARMAX			ARX/ARMAX		
p	ISE	IAE	InEn	ISE	IAE	InEn	ISE	IAE	InEn
6	12821	57.5	25853	11618	47.1	20634	1.10	1.22	1.25
10	11452	45.6	20462	11001	39.5	11043	1.04	1.15	1.85
15	10857	39.3	12491	10656	38.2	8748.3	1.02	1.03	1.43
20	10714	37.7	8881.2	10665	37.2	8141.1	1.00	1.01	1.09

Table 4-8: Performance comparison between ARX and ARMAX models with an additional disturbance d for varying past horizon p for model A, with a sinusoidal wave of frequency 1Hz for $10 \leq t < 20$. The right block shows the relative improvement factor > 1 indicating better performance of ARMAX.

p	ARX			ARMAX			ARX/ARMAX		
	ISE	IAE	InEn	ISE	IAE	InEn	ISE	IAE	InEn
6	236.72	12.0	20443	78.652	6.8	12405	3.01	1.78	1.65
10	155.06	9.9	14966	138.93	8.5	5751.7	1.12	1.16	2.60
15	70.074	6.6	4429.3	45.551	5.5	1158.1	1.54	1.21	3.82
20	43.716	5.4	1259.5	38.733	5.0	387.51	1.13	1.08	3.25

Table 4-9: Performance comparison between ARX and ARMAX models with an additional disturbance d for varying past horizon p for model A, with a sinusoidal wave of frequency 2.61Hz for $20 \leq t < 30$. The right block shows the relative improvement factor > 1 indicating better performance of ARMAX.

p	ARX			ARMAX			ARX/ARMAX		
	ISE	IAE	InEn	ISE	IAE	InEn	ISE	IAE	InEn
6	1281.8	30.35	22600	561.08	19.44	16130	2.28	1.56	1.40
10	592.7	19.8	18562	356.49	15.27	8733.7	1.66	1.30	2.13
15	314.67	14.26	7774	304.4	14.15	4825.3	1.03	1.01	1.61
20	287.99	13.72	5331.3	297.98	14.05	4536.6	0.97	0.98	1.18

Table 4-10: Performance comparison between ARX and ARMAX models with an additional disturbance d for varying past horizon p for model A, with a sawtooth wave of frequency 0.5Hz for $30 \leq t < 40$. The right block shows the relative improvement factor > 1 indicating better performance of ARMAX.

p	ARX			ARMAX			ARX/ARMAX		
	ISE	IAE	InEn	ISE	IAE	InEn	ISE	IAE	InEn
6	5334.3	34.55	23980	4487.9	23.62	14132	1.19	1.46	1.70
10	4733.3	26.25	17655	4329.7	20.96	7472.1	1.09	1.25	2.36
15	4245.5	21.12	6541.1	4262.9	20.51	3833.1	1.00	1.03	1.71
20	4298.4	20.29	4100.8	4268	19.58	3407.6	1.01	1.04	1.20

Chapter 5

Subspace Predictive Control in Practice: Lab Implementation

5-1 System Description

The experimental setup in the lab consists of a 1-DOF torsional system, schematically represented in Figure 5-1. The assembly includes a rotary torsion module coupled to a rotary servo base unit from Quanser. The variables and parameters used in the model are listed in Table 5-1.

Table 5-1: Parameters of the 1-DOF torsional system.

Parameter	Description
V_m [V]	motor input voltage
τ_1 [Nm]	motor torque applied to the motor side
θ_1, θ_2 [rad]	motor- and load-side angles
$\dot{\theta}_1, \dot{\theta}_2$ [rad/s]	angular velocities
J_1, J_2 [kgm^2]	motor- and load-side inertias
B_1, B_2 [Nms/rad]	damping coefficients
K_s [Nm/rad]	torsional spring stiffness

The system is similar to the inertia-spring-damper model introduced in section 4-1. The equations of motion can be derived through first-principle modelling using Newton's laws of motion:

$$J_1\ddot{\theta}_1 = \tau_1 - B_1\dot{\theta}_1 - K_s(\theta_1 - \theta_2), \tag{5-1}$$

$$J_2\ddot{\theta}_2 = -B_2\dot{\theta}_2 + K_s(\theta_1 - \theta_2). \tag{5-2}$$

Defining the state vector as

$$x = \begin{bmatrix}\theta_1 & \dot{\theta}_1 & \theta_2 & \dot{\theta}_2\end{bmatrix}^T, \quad u = \tau_1, \tag{5-3}$$

the system can be expressed in state-space form:

$$\dot{x} = Ax + Bu, \quad y = Cx, \tag{5-4}$$

with

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -\frac{K_s}{J_1} & -\frac{B_1}{J_1} & \frac{K_s}{J_1} & 0 \\ 0 & 0 & 0 & 1 \\ \frac{K_s}{J_2} & 0 & -\frac{K_s}{J_2} & -\frac{B_2}{J_2} \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ \frac{1}{J_1} \\ 0 \\ 0 \end{bmatrix}, \quad C = \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix}. \tag{5-5}$$

The output y can represent either the motor-side angle θ_1 or the load-side angle θ_2, depending on the measurement configuration. In the following experiments, the motor-side angle θ_1 is used, as it contains the desired anti-resonance.

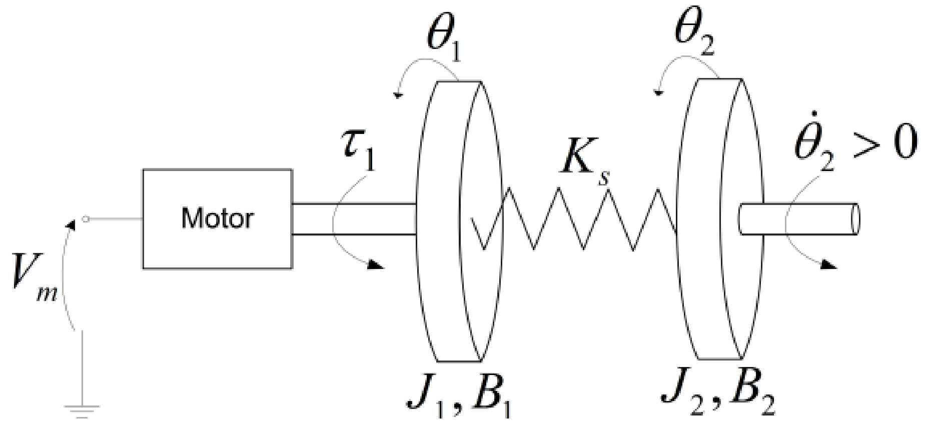


Figure 5-1: Graphical representation of the Quanser setup in the lab [35].

5-2 Experiment Design

The experiments are designed to evaluate the performance of Subspace Predictive Control (SPC) using models identified with both ARX and ARMAX structures. The procedure described below ensures a fair comparison between the two approaches.

For model identification, the system was excited with a chirp signal sweeping from 8 Hz to 1 Hz over a period of 80.256 seconds with an amplitude of $A = 3$. A variable amplitude was applied to account for resonance behaviour at specific frequencies and to prevent potential damage to the setup at higher frequencies. The variable gain is defined as:

$$\text{Amplitude} = \begin{cases} A - 3\sqrt{f-7} & f \geq 7 \text{ Hz} \\ |2(f-6)| + 1 & 5 \leq f \leq 7 \text{ Hz} \\ A & \text{else} \end{cases} \tag{5-6}$$

The sampling time was fixed at $T_s = 0.02$s.

Measurement noise was assessed both in idle state, and through the application of a constant input. The output signal showed no unexplained variation, indicating that the sensor noise is negligible. In addition, the output angle θ_1 is quantised to integer values.

Both ARX and ARMAX models were identified from the same chirp dataset to ensure comparability, this implies that any differences in the results are not due to a difference in the initial dataset. The identified models on this training data were used as initial models in the SPC algorithm. During closed-loop operation, these models were updated online using the (Extended) Recursive Least Squares algorithm as new input-output data became available.

The SPC experiments were carried out using identical controller settings for both ARX- and ARMAX-based models. Multiple prediction horizons, $N_p \in \{10, 15, 25\}$, were tested, with the control horizon chosen equal to the prediction horizon. The weighting matrices were set to

$$Q = I, \qquad R = 50I, \tag{5-7}$$

where I denotes an identity matrix of appropriate size corresponding to the prediction horizon. Input constraints were enforced as $-15 = u_{\min} \leq u \leq u_{\max} = 15$, while no output constraints were applied.

To evaluate control performance under different operating conditions, three different reference trajectories were applied.

- A square wave ($0 \leq t < 20$ s): with an amplitude of 200 an offset of +200, and a frequency of 0.5 Hz.
- A sinusoidal wave ($20 \leq t < 40$ s): with an amplitude of 50 and an offset of +200, and a frequency of 4 Hz.
- A sawtooth wave ($40 \leq t < 60.226$ s): with an amplitude of 200 and an offset of +200, and a frequency of 0.75 Hz.

The performance between ARX- and ARMAX-based SPC is compared through the following measures:

i) Model fit: quantified by the Variance Accounted For (VAF) on training data.

ii) Control performance: evaluated in terms of IAE, ISE, and InEn.

iii) Computational effort: measured for both the online identification and the online optimisation of the computation of the control input.

These experiments allow us to conclude whether ARMAX models offer advantages over ARX in a real-time application of subspace predictive control and whether such benefits justify the additional model complexity.

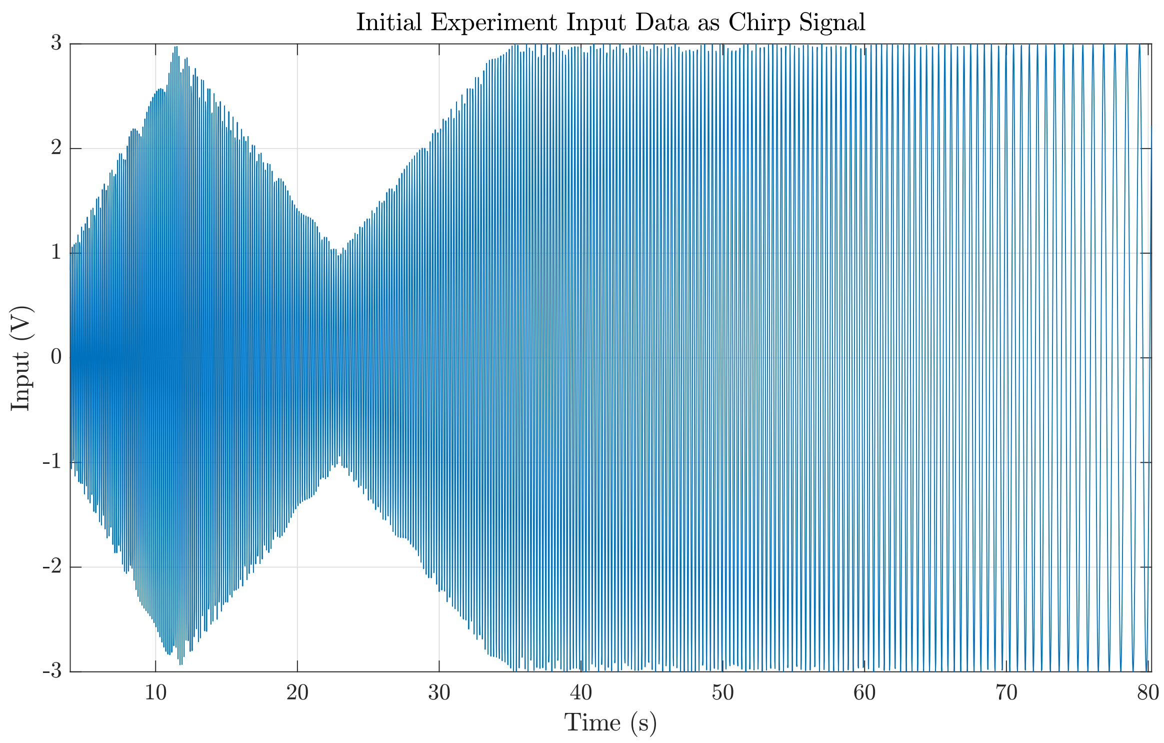
5-3 Identification Results for ARX and ARMAX Models

The model fit is evaluated using the Variance Accounted For (VAF) on the initial training dataset. This dataset was obtained with the chirp excitation signal described in section 5-2. The corresponding input and output signals are shown in Figure 5-2a and Figure 5-2b, respectively. During the experiment, the system exhibited nonlinear behaviour with sudden jumps in the output signal. This effect is likely related to back electromotive force (back-EMF): rotating anti-clockwise appears to be more difficult than rotating clockwise. However, the jumps were observed in both directions, therefore the behaviour is not fully explained by the back-emf principle. To mitigate some of these effects, detrending was applied to obtain a more linear dataset, although not all non-linearities could be removed. Nevertheless, the identification results were sufficiently accurate to produce initial models suitable for use in the SPC algorithm, meaning that a fully perfect linear dataset is not necessarily required.

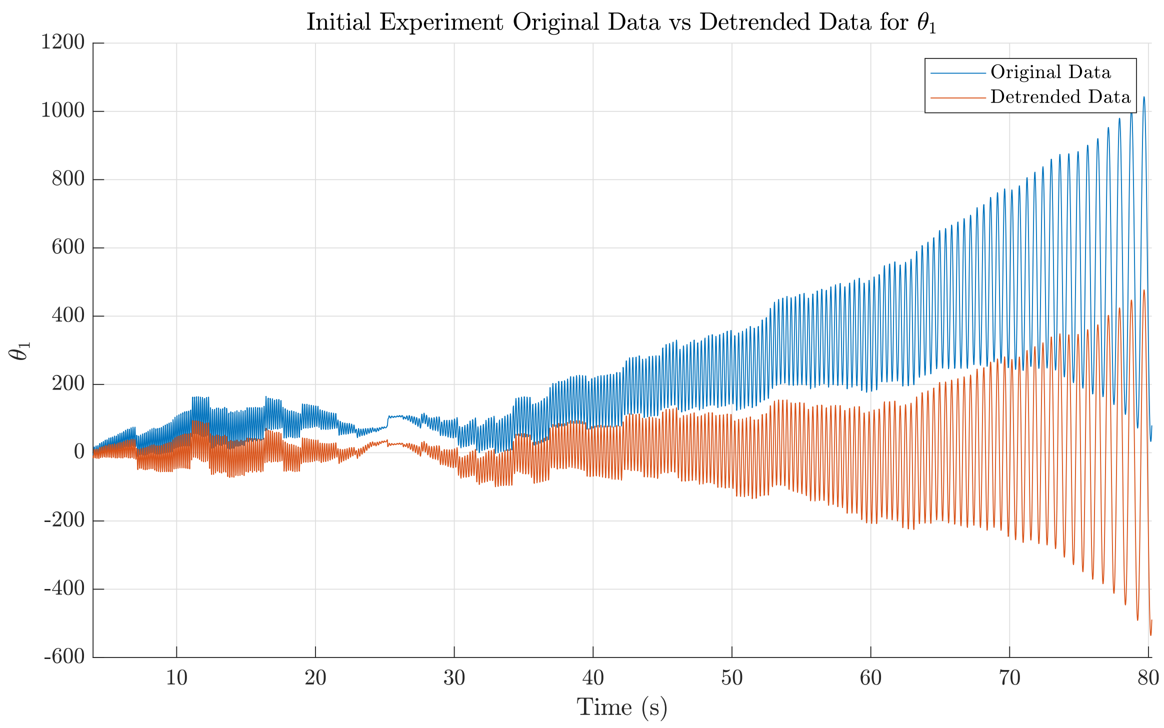
Figure 5-3a presents the VAF for one-step-ahead predictions with a past window of $p = 10$. The algorithms tested include Recursive Least Squares (RLS), Extended RLS (ERLS), and its square-root implementations, batch-wise ARX, iterative batch-wise ARMAX, and the modified ARMAX algorithm by [62]. Among these, the recursive ARMAX implementation achieved the highest VAF, although all methods were able to capture the system dynamics. The BLS based ARMAX models showed some vertical displacement for $p = 10$, most likely caused by the nonlinear jumps in the dataset, yet all models still managed to reproduce the anti-resonance behaviour around $t = 20$ s. Comparable results were obtained for $p = 20$, as shown in Figure 5-3b.

A comparison of VAF values across different model orders is given in Table 5-2. The algorithms exhibit similar performance overall, with the ERLS and BLS ARMAX remaining stable across all tested configurations. The bode plots of the dynamical and noise models for $p = 10$ and $p = 20$ are provided in Figure 5-4a and Figure 5-4b. The dynamical models show similar behaviour and for $p = 20$, the dynamical models of ARX and ARMAX converge to each other. However, the noise models show greater variation. Besides that, both noise models do not show the expected anti-resonance behaviour from Figure 4-1. By comparison, the ARX noise models show unrealistic phase characteristics, with oscillatory behaviour.

Given the comparable VAF performance across methods, the remainder of this chapter focuses on the (Extended) Recursive Least Squares algorithms for ARX and ARMAX.

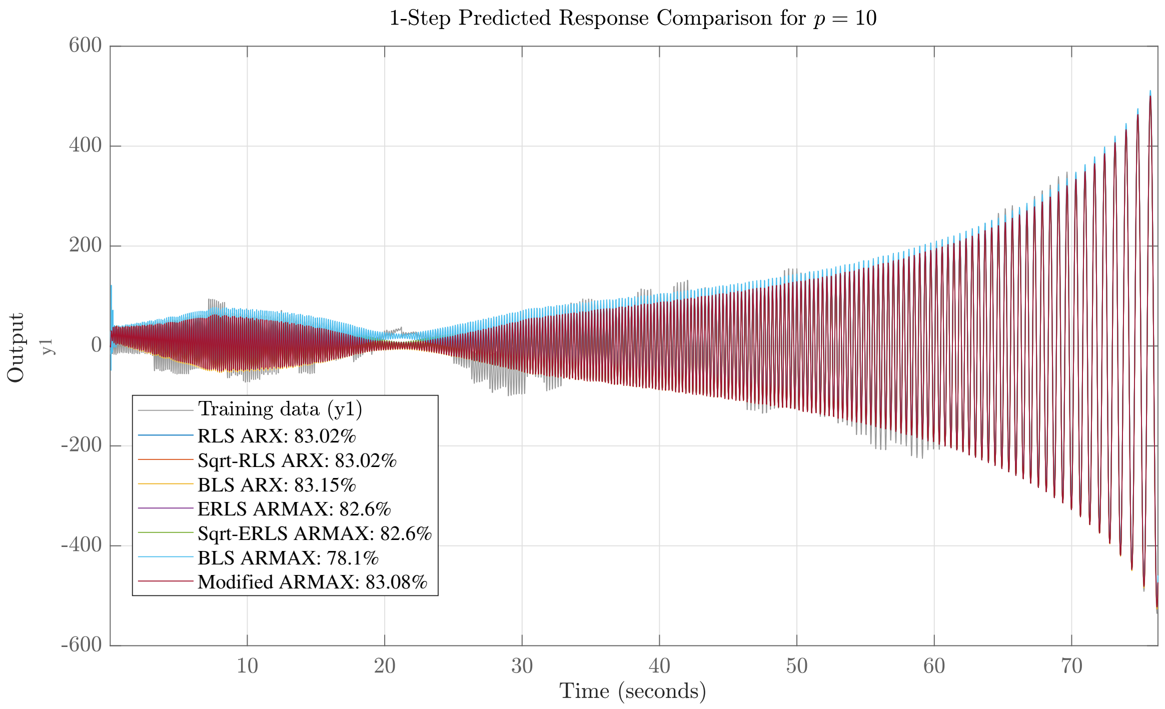


(a) Chirp signal control input

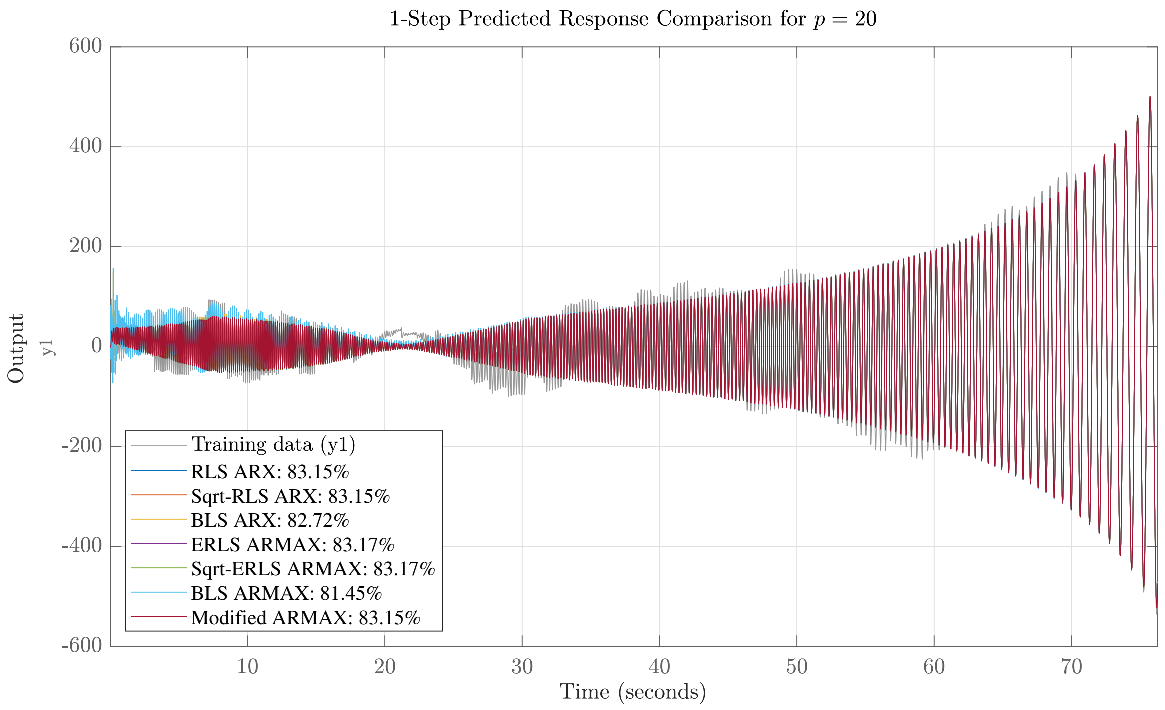


(b) Trended vs detrended output θ_1

Figure 5-2: (a) The chirp signal from 8 Hz to 1 Hz as input signal with variable gain and in (b) the corresponding trended output in blue, and detrended output in orange. The models are trained on the detrended output.



(a) VAF percentages for $p = 10$.

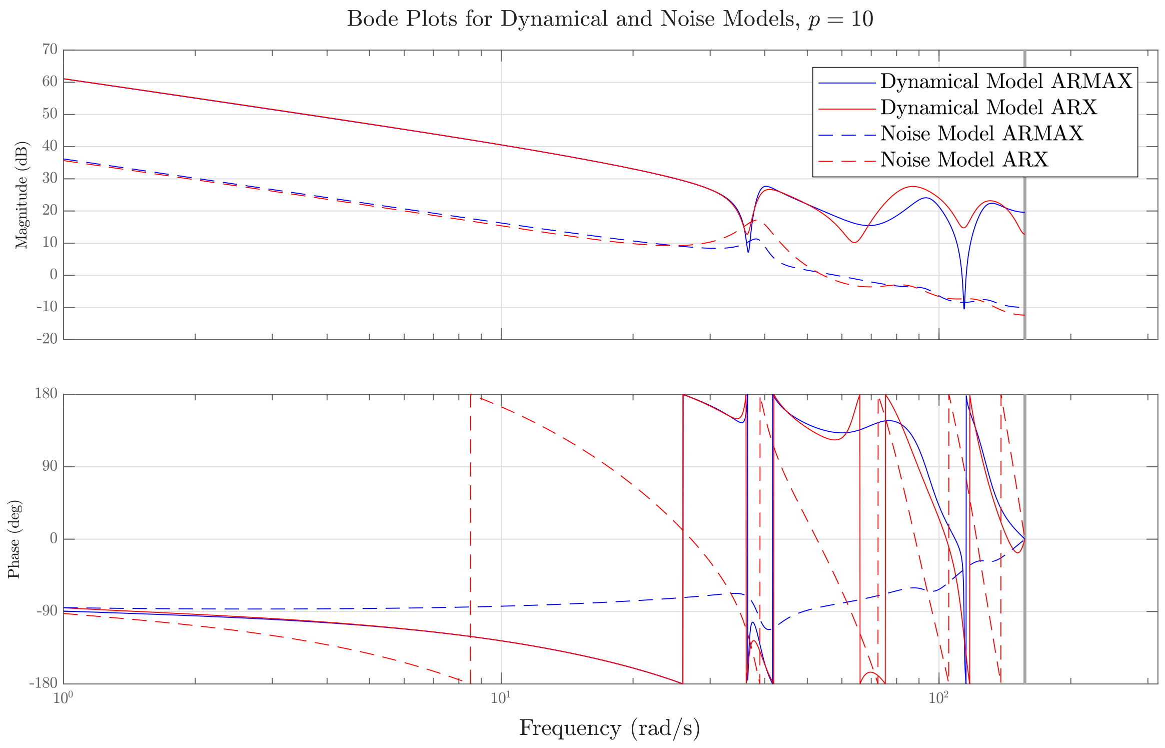


(b) VAF percentages for $p = 20$.

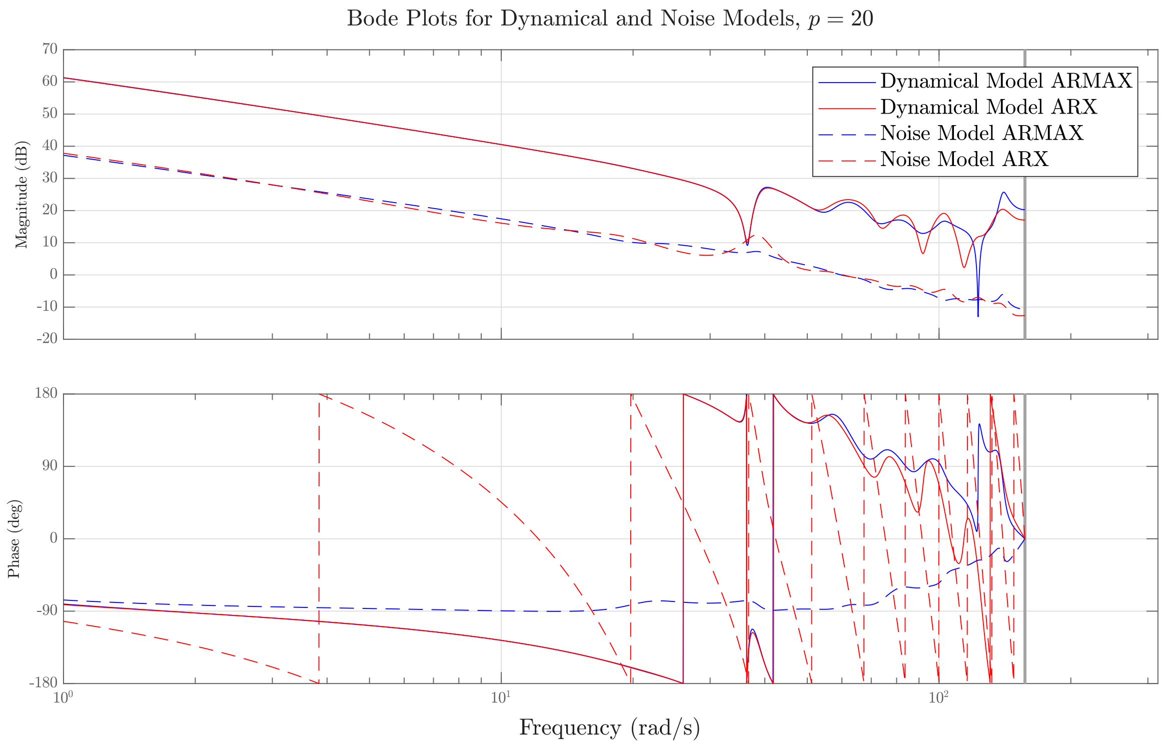
Figure 5-3: Variance Accounted For (VAF) for different identification algorithms and past window sizes $p \in \{10, 20\}$. The algorithms include Recursive Least Squares (RLS), Extended RLS (ERLS), their square-root variants, Batch Least Squares (BLS), and the modified ARMAX algorithm by [62].

Table 5-2: Variance Accounted For (VAF) for different identification algorithms and past window sizes $p \in \{2, 5, 10, 15, 20, 25\}$. The algorithms include Recursive Least Squares (RLS), Extended RLS (ERLS), their square-root variants, Batch Least Squares (BLS), and the modified ARMAX algorithm by [62]. Instability refers to divergence in the 1-step ahead prediction simulations.

Model	p = 6	p = 10	p = 15	p = 20
RLS ARX	Unstable	83.02%	77.46%	83.15%
Sqrt-RLS ARX	Unstable	83.02%	77.46%	83.15%
BLS ARX	Unstable	83.15%	77.14%	82.72%
ERLS ARMAX	82.05%	82.6%	56.8%	83.17%
Sqrt-ERLS ARMAX	82.05%	82.6%	56.8%	83.17%
BLS ARMAX	81.11%	81.53%	83.14%	81.78%
Modified ARMAX	Unstable	83.08%	82.48%	83.15%



(a) $p = 10$.



(b) $p = 20$.

Figure 5-4: Comparison of identification results and the resulting bode plots for magnitude and phase for a past window of $p = 10$ and $p = 20$ for the lab setup.

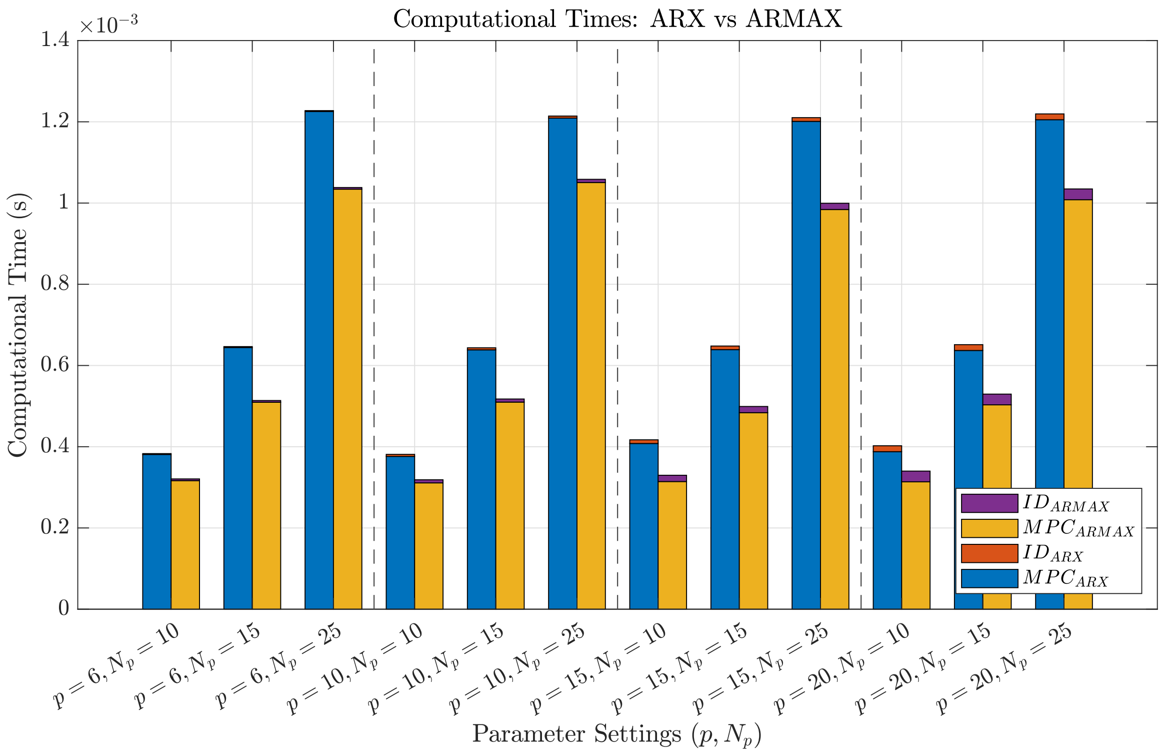
5-4 Comparing ARX- and ARMAX-based Subspace Predictive Control

This section presents the control results obtained from the ARX- and ARMAX-based SPC implementations. The analysis covers three main aspects: computational effort, reference tracking performance, and the evolution of the model identification. The reference tracking capabilities are quantified using the Integral Absolute Error (IAE), Integral Squared Error (ISE), and Input Energy (InEn), evaluated across multiple combinations of model order p and prediction horizon N_p.

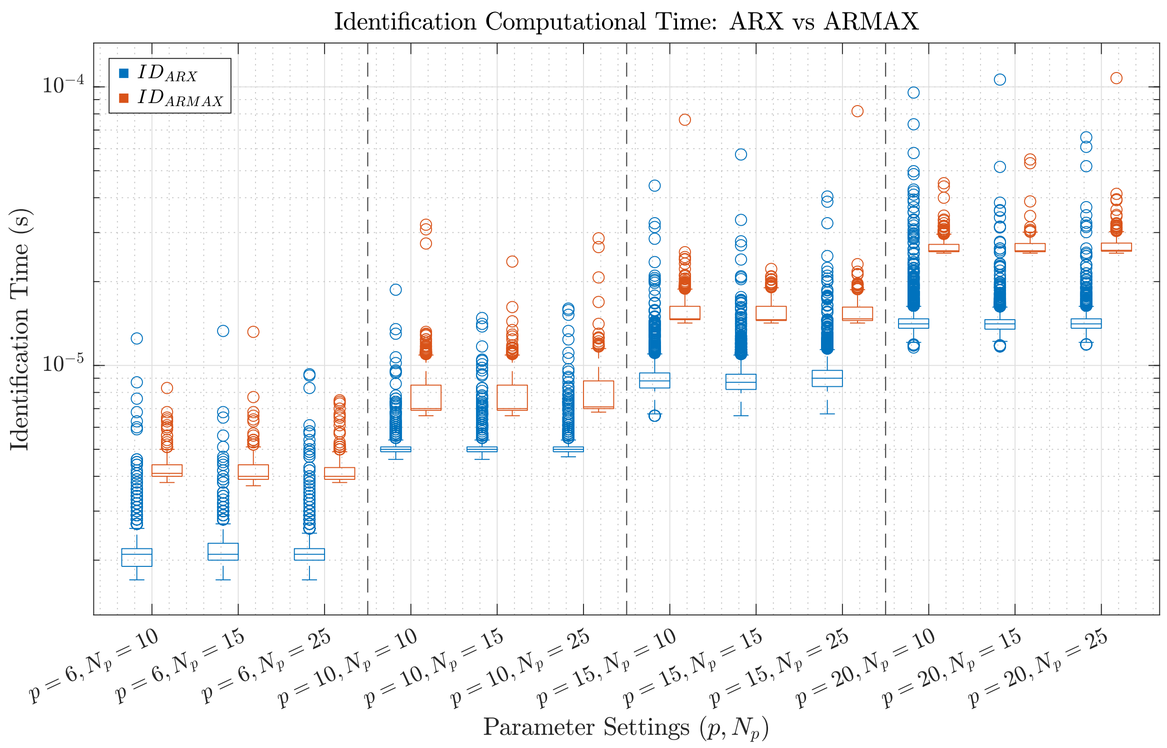
Computational Time

Table 5-3 summarises the computational times for both ARX and ARMAX models. As expected, identification times are consistently higher for ARMAX across all configurations, in line with the theoretical complexity scaling presented in Table 3-3. Interestingly, the MPC execution time is on average lower for ARMAX in all tested cases. The distributions of the measured computational times are illustrated in Figure 5-5b and Figure 5-5c, while Figure 5-5a compares the mean values for the identification and MPC phases. The results indicate that the majority of the computational effort is associated with MPC execution rather than model identification.

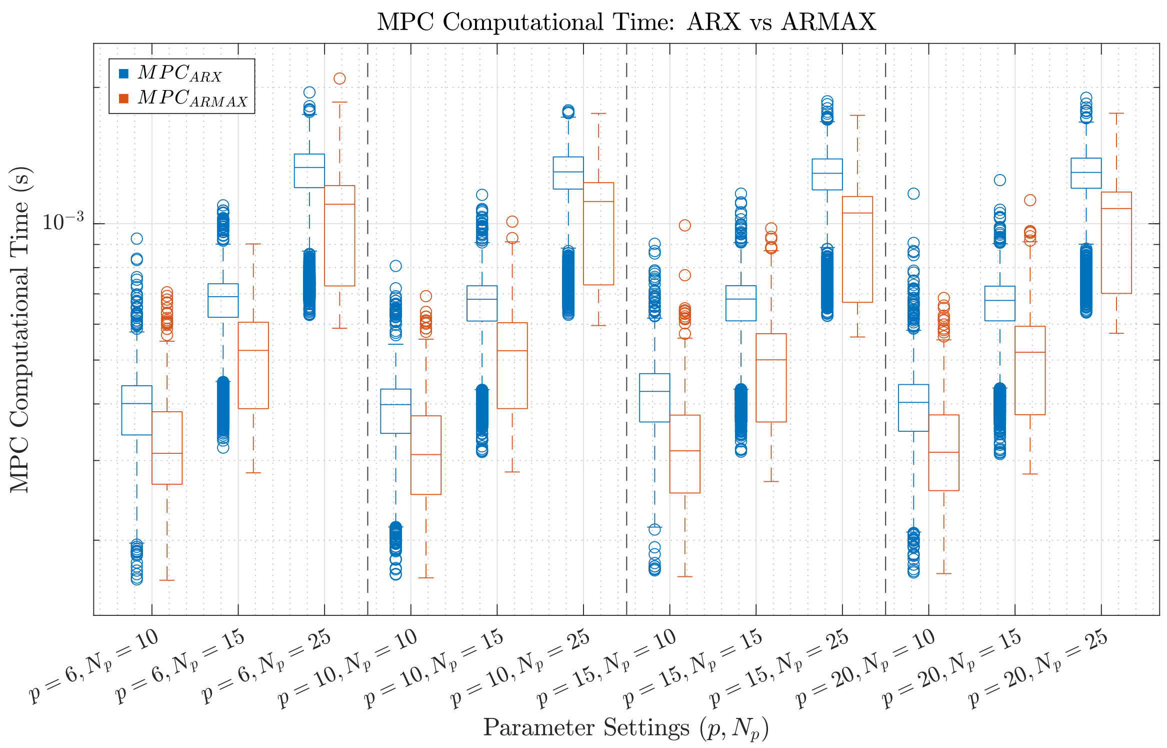
All computational times remain well below the sampling period of $T_s = 20$ ms, with maximum observed values of 1.96 ms and 2.10 ms for ARX and ARMAX, respectively, in the case of $p = 6$ and $N_p = 25$. The boxplots confirm that identification time scales approximately quadratically with the number of model parameters, consistent with the behaviour of (Extended) Recursive Least Squares algorithms. Since ARMAX employs approximately 1.5 times as many parameters as ARX for the same order p, its identification phase requires roughly twice as long. In contrast, the MPC computational load is dominated by the prediction horizon N_p, and despite its higher model complexity, ARMAX show slightly faster MPC execution on average, with a higher variance.



(a) Stacked Bar



(b) Recursive Least Squares



(c) MPC

Figure 5-5: Computational times of the (Extended) Recursive Least Squares step, and the MPC step.

Table 5-3: Computational time statistics for ARX and ARMAX models across different (p, N_p) pairs. A sampling time T_s of 20ms is used.

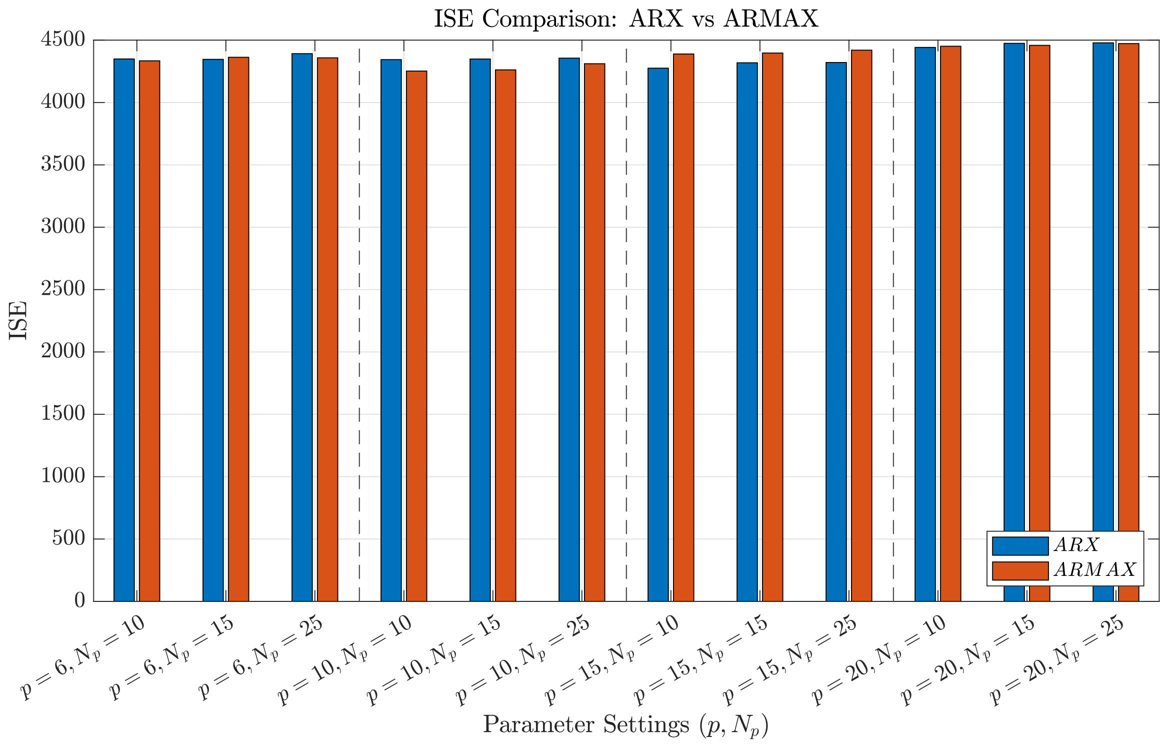
		Identification Time (ms)			MPC Time (ms)		
(p, N_p)	Model	Mean	Std	Max	Mean	Std	Max
(6, 10)	ARX	0.00211	0.00038	0.01250	0.38099	0.09066	0.92710
	ARMAX	0.00419	0.00030	0.00830	0.31657	0.08309	0.70570
(6, 15)	ARX	0.00218	0.00034	0.01330	0.64437	0.14254	1.09810
	ARMAX	0.00417	0.00035	0.01320	0.50962	0.12136	0.90250
(6, 25)	ARX	0.00215	0.00037	0.00930	1.22542	0.28853	1.95050
	ARMAX	0.00413	0.00030	0.00750	1.03408	0.24523	2.09260
(10, 10)	ARX	0.00507	0.00047	0.01870	0.37614	0.08645	0.80690
	ARMAX	0.00771	0.00132	0.03200	0.31095	0.07793	0.69200
(10, 15)	ARX	0.00511	0.00050	0.01480	0.63847	0.13982	1.15810
	ARMAX	0.00764	0.00115	0.02360	0.50988	0.12031	1.01090
(10, 25)	ARX	0.00510	0.00055	0.01600	1.20910	0.28110	1.78000
	ARMAX	0.00785	0.00128	0.02860	1.05054	0.25096	1.75240
(15, 10)	ARX	0.00904	0.00151	0.04420	0.40797	0.08923	0.90300
	ARMAX	0.01547	0.00167	0.07620	0.31429	0.07947	0.99170
(15, 15)	ARX	0.00895	0.00163	0.05720	0.63899	0.13880	1.16530
	ARMAX	0.01530	0.00111	0.02220	0.48382	0.11520	0.97660
(15, 25)	ARX	0.00921	0.00156	0.04040	1.20106	0.27591	1.86380
	ARMAX	0.01529	0.00160	0.08180	0.98413	0.23878	1.73510
(20, 10)	ARX	0.01457	0.00304	0.09540	0.38782	0.09255	1.16530
	ARMAX	0.02645	0.00127	0.04510	0.31356	0.07745	0.68560
(20, 15)	ARX	0.01433	0.00238	0.10620	0.63700	0.13781	1.24840
	ARMAX	0.02642	0.00130	0.05490	0.50316	0.12039	1.12720
(20, 25)	ARX	0.01434	0.00200	0.06590	1.20513	0.27571	1.89740
	ARMAX	0.02660	0.00195	0.10750	1.00815	0.23978	1.75380

Reference Tracking

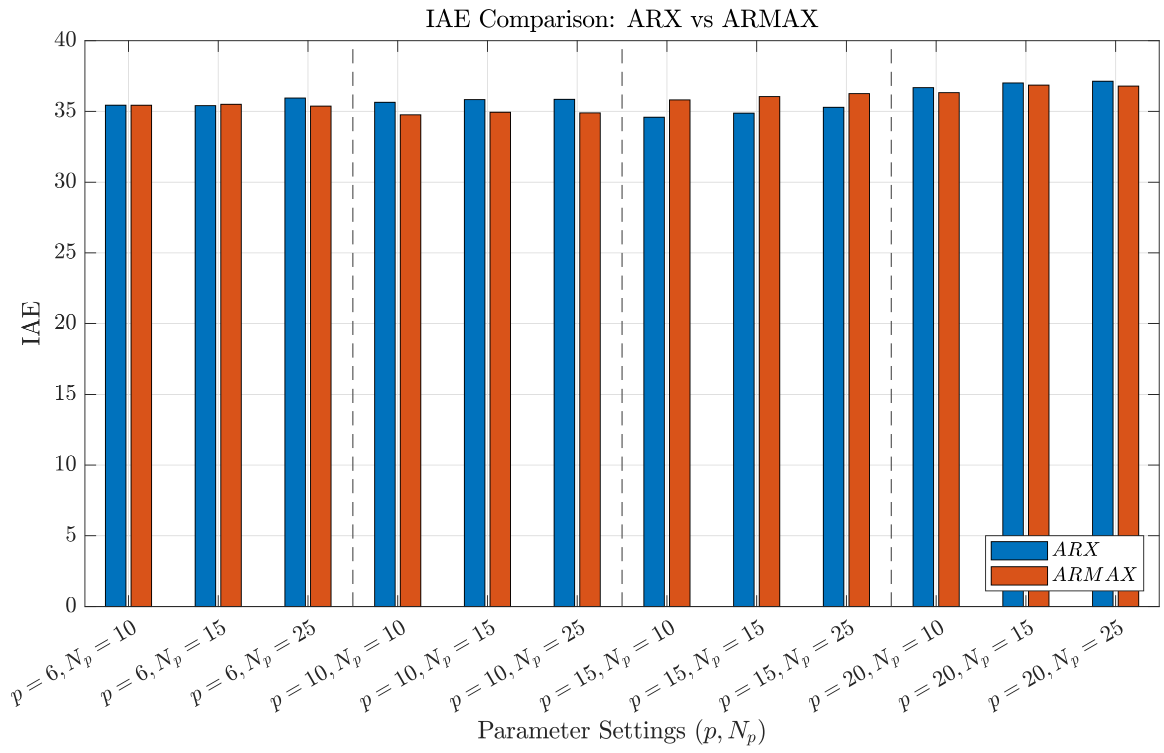
Three reference signals, a square wave, sinusoid, and sawtooth, were used to evaluate the tracking performance of the ARX- and ARMAX-based SPC controllers. The corresponding control metrics for different (p, N_p) configurations are summarised in Table 5-4, while the time-domain responses for $p = 10$ and $p = 15$ with $N_p = 10$ are shown in Figure 5-7, Figure 5-8, and Figure 5-9, respectively, over a four-second interval.

Overall, the best tracking performance is achieved for $p = 10$ in both models, with ARMAX showing a slightly better performance across all evaluated metrics. For other parameter configurations, however, the difference between ARX and ARMAX is insignificant. This trend is also reflected in the bar plots of ISE, IAE, and InEn shown in Figure 5-6a to 5-6c.

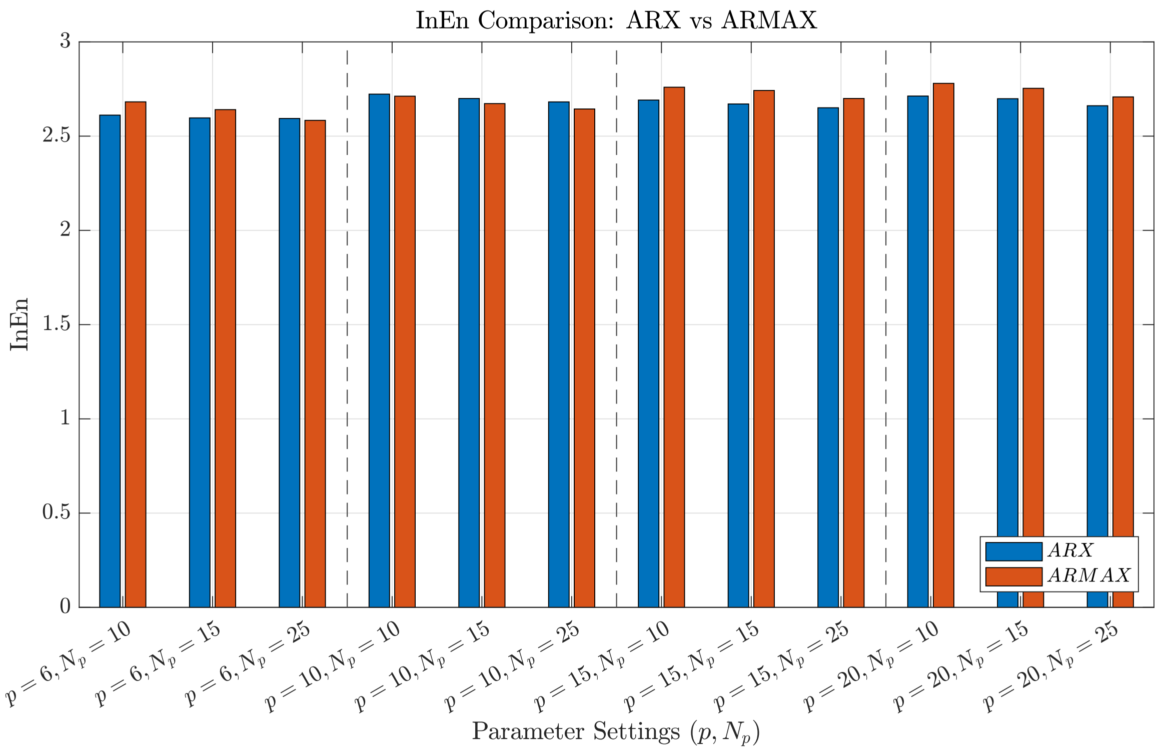
A closer inspection of the time-domain responses reveals some notable differences. For instance, in Figure 5-8, the ARMAX controller exhibits a higher amplitude response than ARX when tracking the sinusoidal reference for $p = 15$. Although this results in slightly worse quantitative metrics, the discrepancy can be attributed to a small phase delay, which skews the performance metrics.


ISE Comparison: ARX vs ARMAX
ISE
Parameter Settings (p, N_p)
ARX
$ARMAX$
0
500
1000
1500
2000
2500
3000
3500
4000
4500
$p = 6, N_p = 10$
$p = 6, N_p = 15$
$p = 6, N_p = 25$
$p = 10, N_p = 10$
$p = 10, N_p = 15$
$p = 10, N_p = 25$
$p = 15, N_p = 10$
$p = 15, N_p = 15$
$p = 15, N_p = 25$
$p = 20, N_p = 10$
$p = 20, N_p = 15$
$p = 20, N_p = 25$

(a) ISE


IAE Comparison: ARX vs ARMAX
IAE
Parameter Settings (p, N_p)
ARX
$ARMAX$
0
5
10
15
20
25
30
35
40
$p = 6, N_p = 10$
$p = 6, N_p = 15$
$p = 6, N_p = 25$
$p = 10, N_p = 10$
$p = 10, N_p = 15$
$p = 10, N_p = 25$
$p = 15, N_p = 10$
$p = 15, N_p = 15$
$p = 15, N_p = 25$
$p = 20, N_p = 10$
$p = 20, N_p = 15$
$p = 20, N_p = 25$

(b) IAE

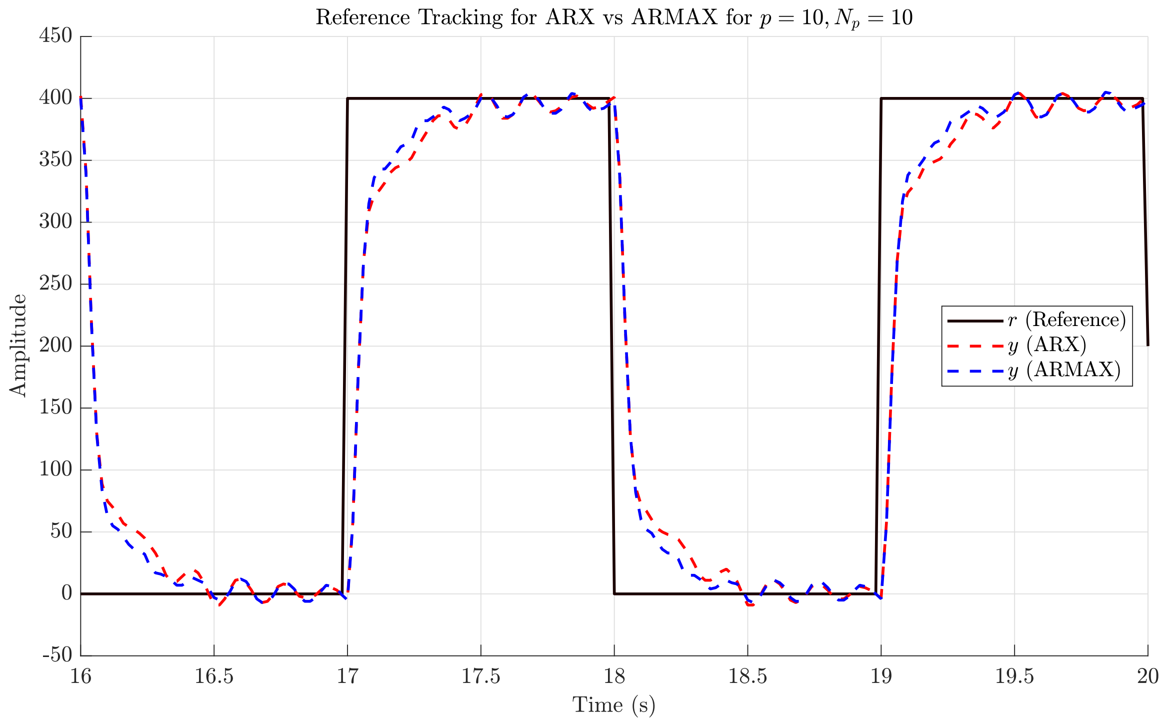


(c) InEn

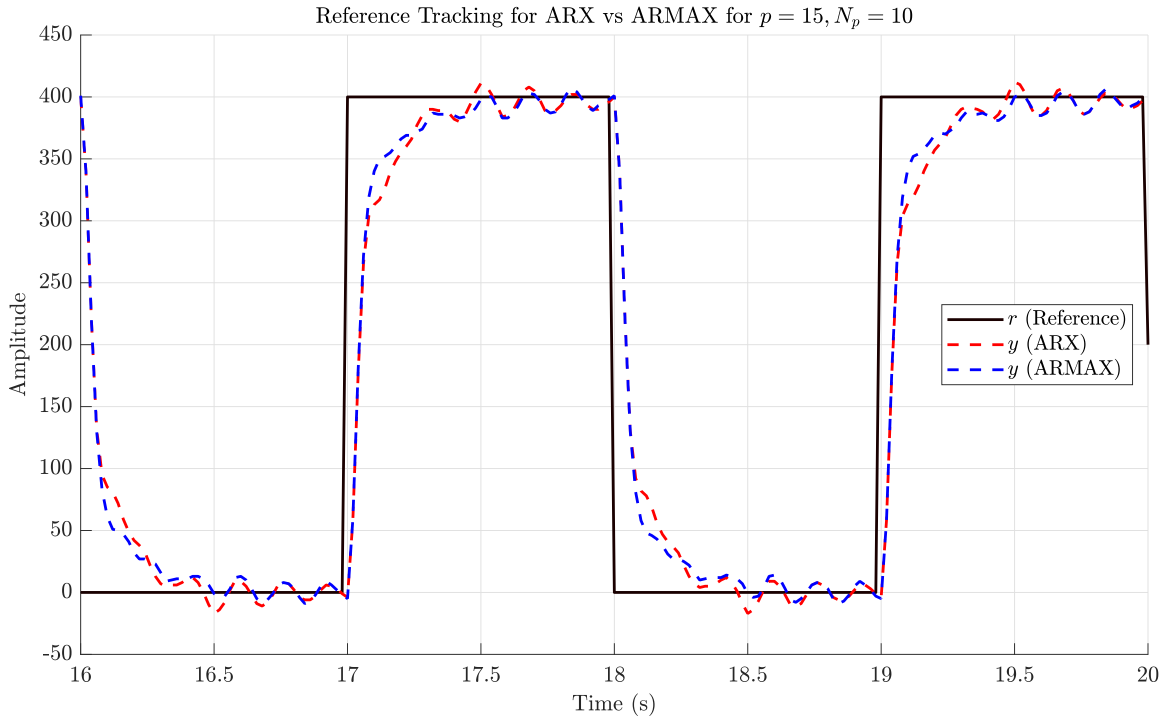
Figure 5-6: Bar plots of the performance metrics ISE, IAE and InEn.

Table 5-4: The performance metrics ISE, IAE, and InEn for ARX and ARMAX models across different (p, N_p) pairs.

	ISE		**IAE**		**InEn**	
(p, N_p)	ARX	ARMAX	ARX	ARMAX	ARX	ARMAX
(6, 10)	4349	4334	35.44	35.44	2.61	2.68
(6, 15)	4346	4363	35.41	35.50	2.56	2.64
(6, 25)	4392	4358	35.95	35.38	2.59	2.58
(10, 10)	4344	4252	35.64	34.76	2.72	2.71
(10, 15)	4349	4262	35.83	34.94	2.70	2.67
(10, 25)	4356	4311	35.85	34.90	2.68	2.64
(15, 10)	4275	4389	34.59	35.82	2.69	2.76
(15, 15)	4318	4397	34.88	36.05	2.67	2.74
(15, 25)	4320	4420	35.29	36.26	2.65	2.70
(20, 10)	4442	4451	36.68	36.33	2.71	2.78
(20, 15)	4474	4458	37.01	36.86	2.70	2.75
(20, 25)	4479	4472	37.13	36.79	2.66	2.71

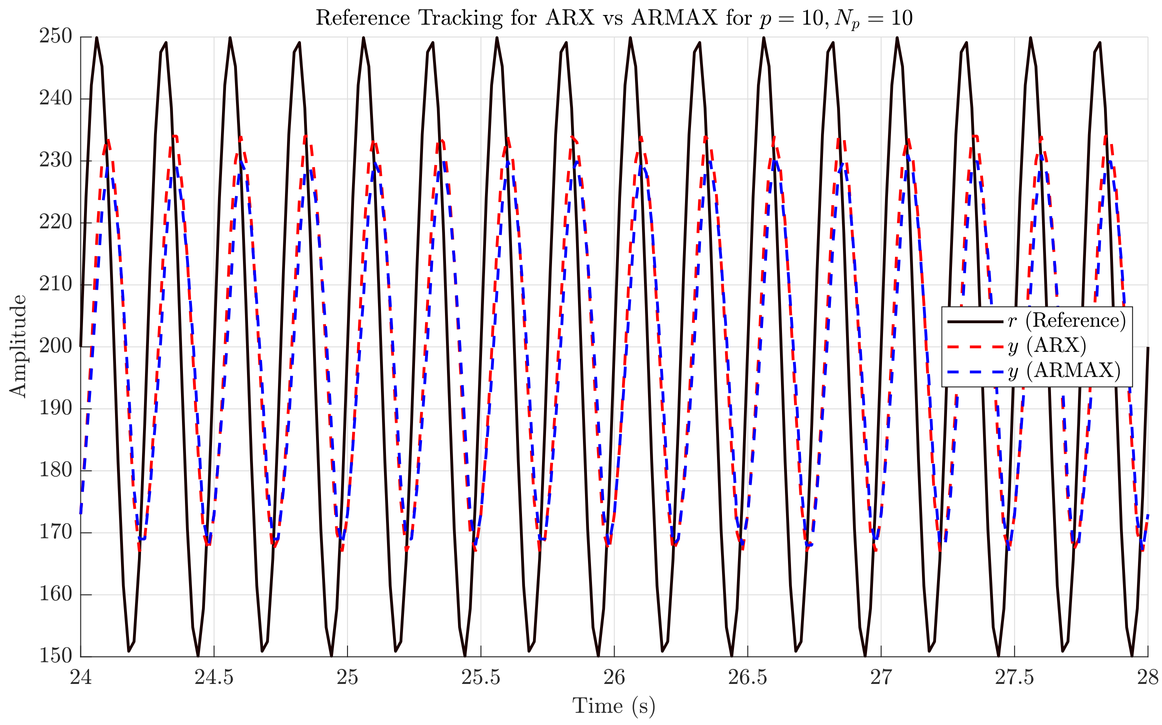


(a) $p = 10,\ N_p = 10$

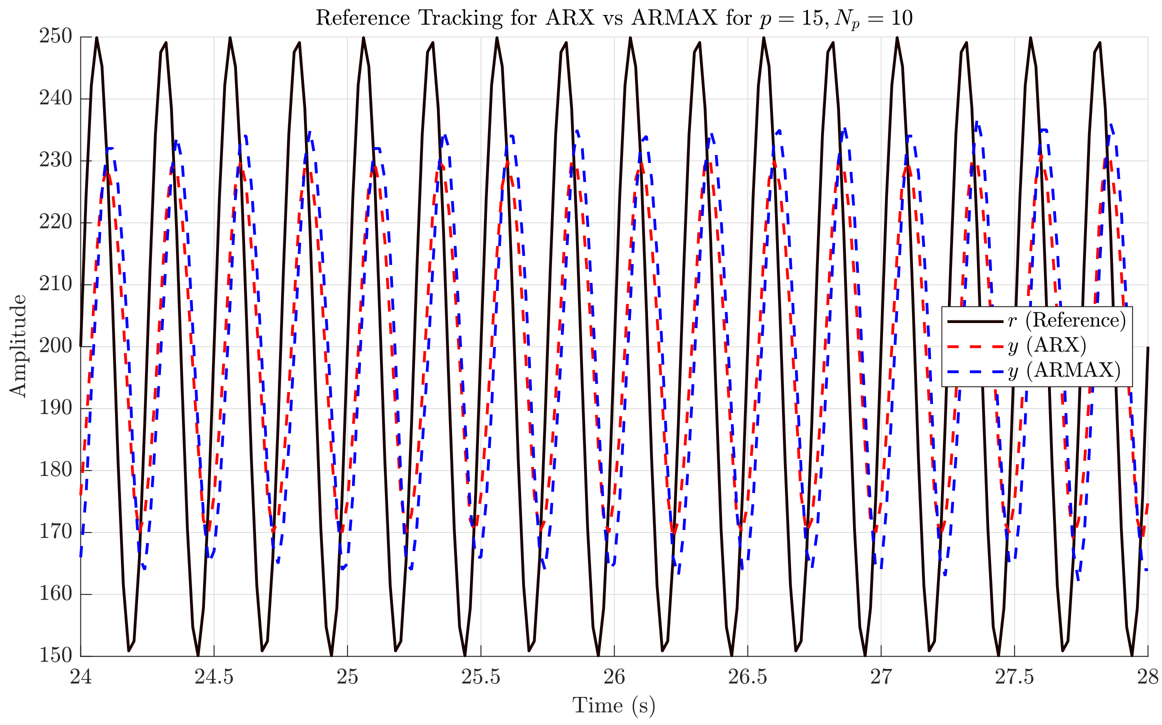


(b) $p = 15,\ N_p = 10$

Figure 5-7: Reference Tracking of a square wave (0.5 Hz) for ARX and ARMAX for $p = 10$ and $p = 15$. The prediction horizon is kept constant with $N_p = 10$.

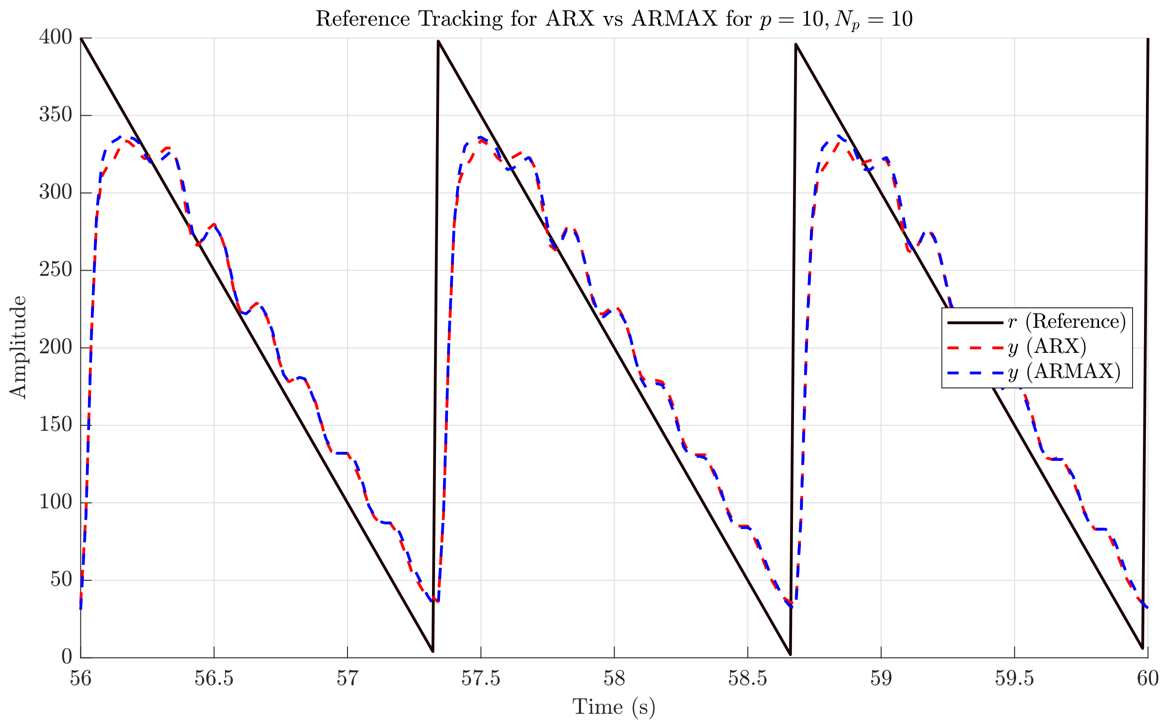


(a) $p = 10,\ N_p = 10$

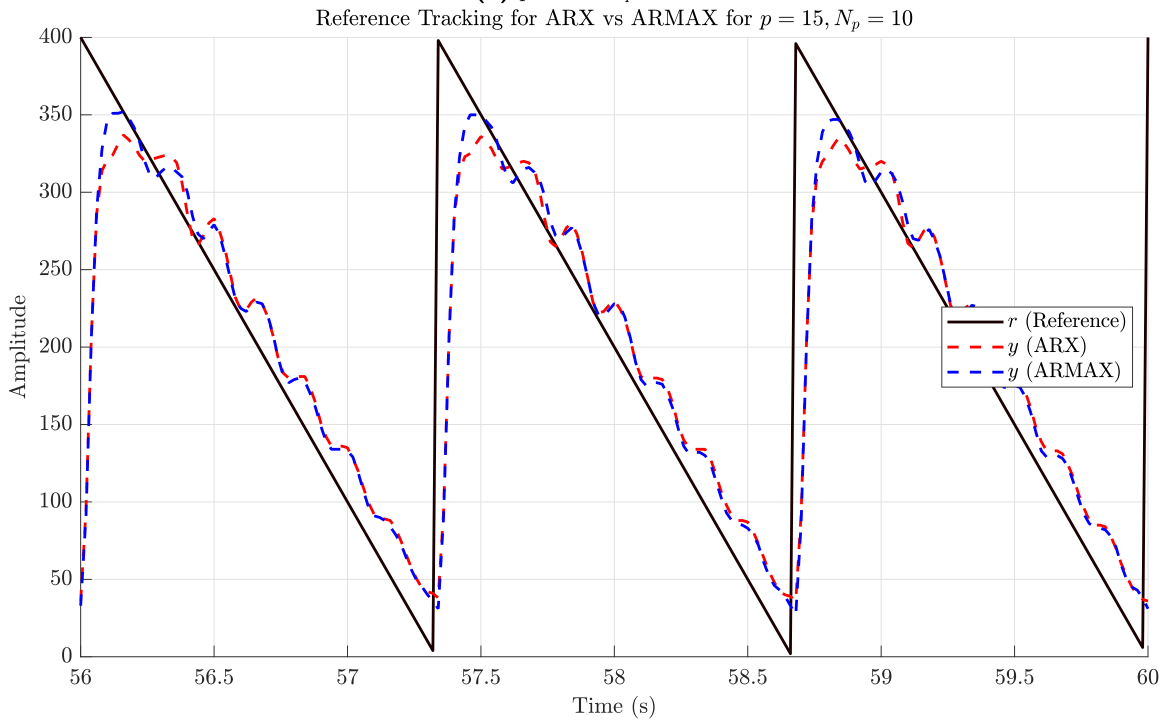


(b) $p = 15,\ N_p = 10$

Figure 5-8: Reference Tracking of a sinusoidal wave (4.0 Hz) for ARX and ARMAX for $p = 10$ and $p = 15$. The prediction horizon is kept constant with $N_p = 10$.



(a) $p = 10,\ N_p = 10$



(b) $p = 15,\ N_p = 10$

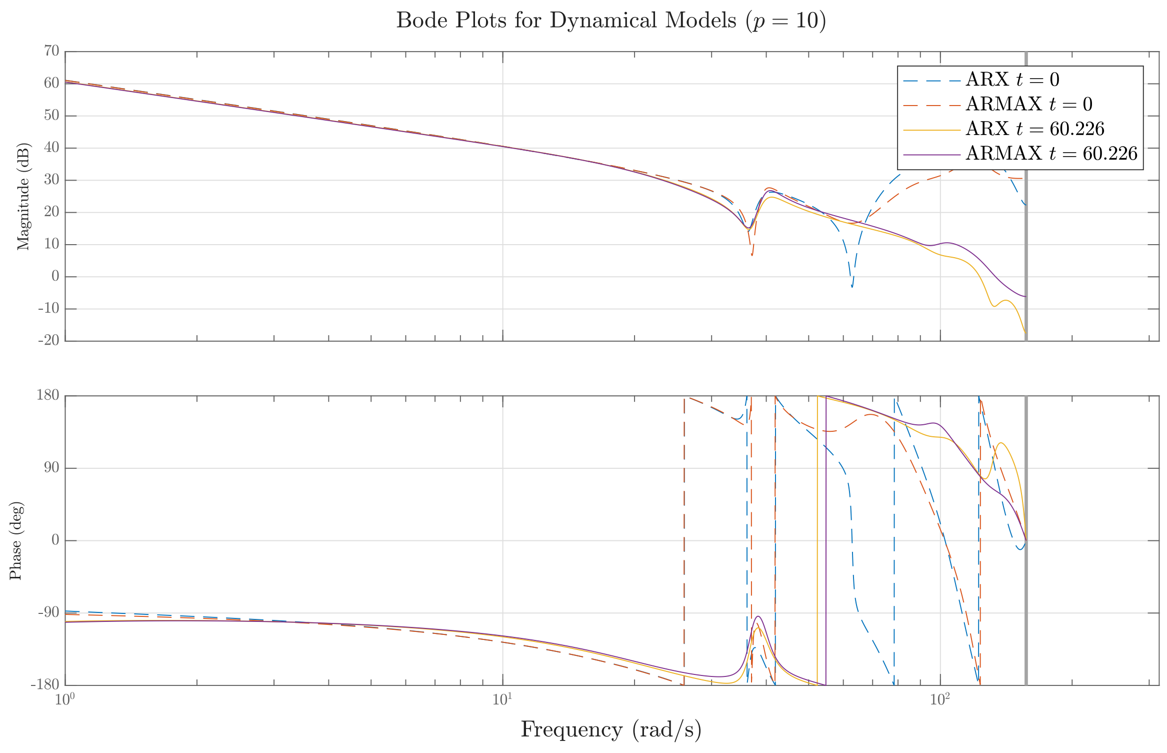
Figure 5-9: Reference Tracking of a sawtooth wave (0.75 Hz) for ARX and ARMAX for $p = 10$ and $p = 15$. The prediction horizon is kept constant with $N_p = 10$.

Model Identification

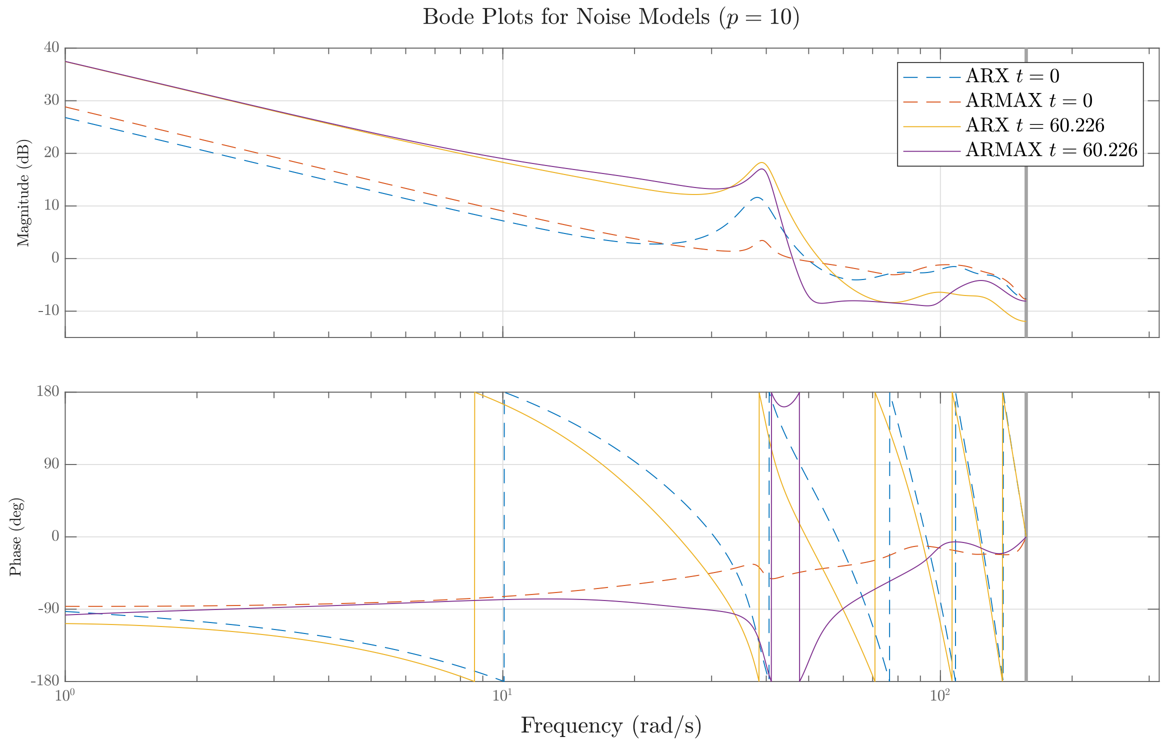
The evolution of the identified models during closed-loop operation is shown in Figure 5-10a and Figure 5-10b. The identified dynamical model remains largely consistent throughout the experiment for both ARX and ARMAX, with only a slight shift in the amplitude of the anti-resonance frequency observed over time. Since both methods identify a similar dynamical model this explains the comparable control results. In contrast, the estimated noise model exhibits more pronounced variations. These changes are primarily attributed to the large prediction errors occurring during abrupt reference transitions, where sudden input changes exceed the systems capacity to respond instantaneously. Implementing an input rate limiter could help mitigate this issue by smoothing the control signal. A forgetting factor of $\lambda = 0.9998$ was used, corresponding to an effective memory length of approximately $N = \frac{1}{1-\lambda} \approx 5000$ samples.

Summary

In summary, both ARX- and ARMAX-based SPC achieve comparable closed-loop performance across all tested configurations. Although ARMAX identification requires slightly higher computational effort for equal values of p, its additional cost is negligible relative to the total control computation time. However, the marginal performance improvement of ARMAX over ARX suggests that, under the tested conditions, the simpler ARX structure is sufficient. Nonetheless, the ARMAX framework remains valuable for scenarios involving coloured disturbances or stronger noise dynamics.



(a) Dynamic Models



(b) Noise Models

Figure 5-10: The dynamical models and noise models for ARX and ARMAX for $p = 10$. The noise models change significantly over time, while the dynamical model remains consistent.

Chapter 6

Conclusion

The purpose of this thesis was to study the Subspace Predictive Control (SPC) algorithm beyond its original formulation with an ARX model. Two research questions, each with several sub-questions, were central to this work:

1. Is it possible to integrate an ARMAX model into the SPC framework to achieve improved noise modelling??
   - What system assumptions are required for the SPC algorithm to function with the ARMAX model?
   - How does the ARMAX model change the identification of the Markov parameters?
   - How does the ARMAX model change the formulation of the data equations?
   - How does the ARMAX model change the predictor equations required for Model Predictive Control?
2. How can ARMAX-based SPC be applied to a real-life system exhibiting an anti-resonance?
   - How can the ARMAX-based SPC algorithm be implemented efficiently within a real-time control framework?
   - Which numerical techniques can enhance computational efficiency, accuracy, and robustness?
   - How does the control performance and computational cost compare between ARX- and ARMAX-based SPC implementations?
   - How does the increased number of parameters in the ARMAX model influence identification accuracy and real-time feasibility?

6-1 Integration of an ARMAX Model into the Subspace Predictive Control Framework

This thesis demonstrated that Subspace Predictive Control (SPC) can be successfully extended to incorporate AutoRegressive Moving Average with eXogenous input (ARMAX) models by introducing the $C(q)$ polynomial, which explicitly captures coloured noise dynamics. The theoretical assumptions largely coincide with those of the ARX-based formulation: the system must be asymptotically stable, with all poles located inside the unit circle such that $\tilde{A}^{p-1} \to 0$ as $p \to \infty$. In practice, however, satisfactory closed-loop performance can already be achieved with a smaller past window p. Compared to ARX-based SPC, ARMAX-based SPC can operate more effectively with a reduced model order p. This does not lead to a computational advantage, as the computational load for both methods scales equal and quadratically with the number of parameters, and not the model order p.

The inclusion of the moving-average term modifies the standard linear least-squares formulation into a nonlinear problem, as the residuals depend on the model parameters themselves. This extension enables a more accurate separation between deterministic and stochastic dynamics, leading to significantly improved estimation of both the dynamics and the noise behaviour. The resulting nonlinear least-squares formulation was solved using the Extended Recursive Least Squares (ERLS) algorithm, which updates both the deterministic and stochastic parameters online. This method preserves the recursive structure of standard RLS while incorporating the estimated residuals into the regression vector ϕ, enabling real-time adaptation.

However, the extension increases both the number of parameters and the degree of correlation between the autoregressive, exogenous, and moving-average terms, which can affect numerical conditioning and increase parameter variance. To mitigate these effects for higher model orders, future implementations could include advanced regularisation such as kernel-based priors to enforce smoothness or exponential decay in the estimated Markov parameters. Despite this added complexity, ARMAX-based SPC achieves better closed-loop performance at lower model orders compared to ARX-based SPC. This advantage vanishes as the model order increases, where both formulations converge to similar lower performance.

For prediction and control, the challenge arises from the fact that future noise realisations are unknown. While parts of the noise sequence can be estimated from past data, the deterministic MPC prevents full exploitation of the stochastic model. Although a stochastic or robust MPC formulation could explicitly account for this uncertainty, the certainty-equivalence approach adopted in this thesis was found sufficient for the considered system and experimental conditions.

The next section concludes how this extended SPC framework performs when applied to a real-life system.

6-2 Application of ARMAX-Based Subspace Predictive Control to a Real-Time System

The proposed framework was implemented and validated both through simulations and laboratory experiments. Recursive Least Squares (RLS) and Extended RLS (ERLS) algorithms were employed for online identification, demonstrating stable parameter convergence. The total computational cost was dominated by the MPC optimisation rather than identification, confirming the real-time feasibility of both ARX- and ARMAX-based SPC implementations. All computations remained well below the sampling period of $T_s = 20$ms, with typical execution times around 2ms.

Simulation results (section 4-2) confirmed that high-order ARX models can reproduce both system and noise behaviour with high accuracy, but at the cost of interpretability and parameter variance. ARMAX models achieved comparable or improved identification quality with a lower-order model p, resulting in more compact models that better captured coloured noise effects. Under stochastic disturbances, ARMAX-based SPC achieved improved tracking in terms of lower IAE and ISE while maintaining significantly lower control effort InEn. These improvements were most visible for lower model orders, as an increasing model order for ARX and ARMAX resulted in similar performance. Moreover, the ARMAX models provided a more consistent phase response in the frequency domain of the noise model, while the higher-order ARX models exhibited oscillatory phase changes due to a pole model.

Laboratory experiments (chapter 5) did not confirm these improved tracking trends and lower actuator effort observed in simulations. The discrepancy is primarily attributed to nonlinearities in the physical system, which introduced inconsistent prediction errors, as well as the absence of significant measurement noise. Both ARX- and ARMAX-based implementations achieved stable closed-loop control.

The combined simulation and experimental findings suggest that ARMAX-based SPC provides a more general and robust framework for predictive control, particularly in systems with coloured noise. In deterministic or low-noise environments, however, the simpler ARX-based SPC remains preferable due to its lower model complexity and equivalent control performance.

From a numerical standpoint, both RLS and ERLS proved to be computationally efficient, with square-root formulations offering further improvements in numerical stability. Nonetheless, the nonlinear nature of ARMAX estimation introduced highly correlated parameters, suggesting that more sophisticated identification strategies could further enhance identification performance, without increasing the overall computational cost of the complete SPC algorithm significantly.

6-3 Discussion

The results presented in this work demonstrate that the Subspace Predictive Control (SPC) framework can be successfully extended to include ARMAX models, enabling a more explicit treatment of coloured noise dynamics. While the theoretical development and numerical implementation were validated both in simulation and through experimental testing, several observations require further discussion regarding model complexity, experimental performance, numerical conditioning, and predictive control design.

The inclusion of the noise model increased the total number of estimated parameters and introduced nonlinearity into the identification problem. This extension improved the representation of coloured noise and enhanced prediction quality, but only led to improvements in tracking performance at lower model orders p in simulation, while higher-order models showed similar performance. However, lower-order ARMAX models offer higher interpretability, as individual dynamic and noise contributions can be more easily analysed from the estimated parameters. Furthermore, in deterministic or low-noise environments, the ARX formulation achieved comparable control performance. Hence, the practical advantage of ARMAX-based SPC depends strongly on the systems noise characteristics.

The experimental results did not fully confirm the improved tracking and reduced control effort observed in simulation. This discrepancy likely arises from nonlinearities in the hardware setup, an additional rate-limiter in the MPC formulation could potentially address this issue. Besides that, the system lacked measurement noise. Therefore, the ARMAX noise model designed to capture stochastic behaviour had limited opportunity to contribute in the experimental conditions. These results suggest that the theoretical benefits of ARMAX-based SPC are most evident when coloured noise significantly influences system behaviour. For systems where noise is negligible, the ARX-based SPC is sufficient.

Both Recursive Least Squares (RLS) and Extended RLS (ERLS) algorithms proved to be computationally efficient for online identification, with total computation times well below real-time requirements. However, the nonlinear coupling of ARMAX parameters introduced stronger correlations between the autoregressive, exogenous, and moving-average terms, resulting in higher parameter variance. Future implementations could include instrumental variable methods, advanced regularisation such as kernel-based priors or a more complex recursive algorithm. Moreover, square-root formulations should theoretically improve long-term numerical stability and are recommended for future implementations.

The deterministic certainty-equivalence assumption adopted in this thesis was sufficient for accurate tracking in the examined cases but does not exploit the full stochastic potential of the ARMAX model. Integrating stochastic or robust MPC formulations could allow explicit treatment of uncertainty and improved constraint handling under noisy conditions.

6-4 Recommendations

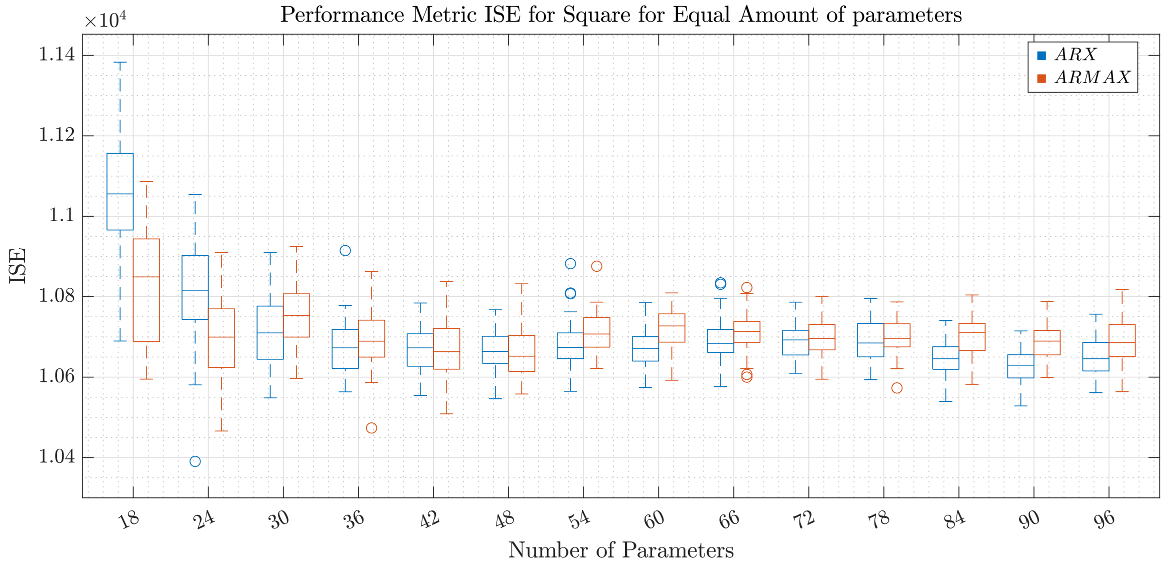
The results of this thesis demonstrated that Subspace Predictive Control (SPC) can be effectively extended to include ARMAX models, yielding improved noise representation and improved closed-loop performance for lower-order models, while maintaining real-time feasibility. Nevertheless, several open challenges remain with respect to scalability, robustness, and theoretical guarantees. Future research should therefore focus on extending the current framework to more complex system structures and strengthening the theoretical and numerical foundation of the SPC algorithm.

Future research could focus on the following aspects:

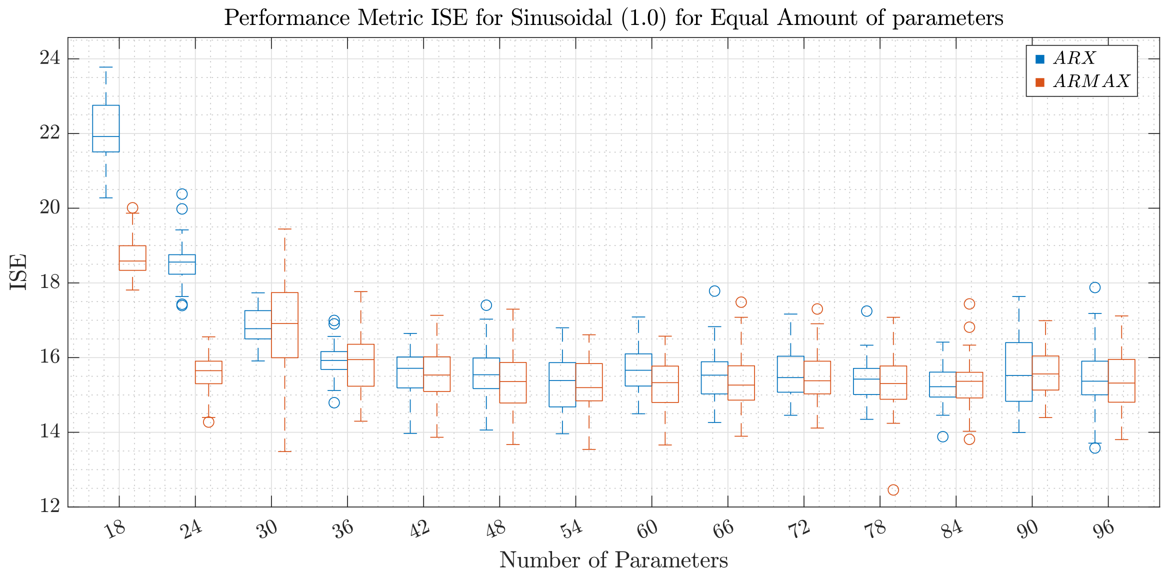
- Extension to MIMO systems: Extend the current single-input single-output (SISO) formulation to multi-input multi-output (MIMO) systems, where coloured noise coupling and cross-correlation between channels could become more problematic.
- Stochastic MPC formulations: Investigate stochastic MPC formulations that explicitly incorporate the ARMAX noise model, enabling probabilistic constraint handling and potentially improved robustness to uncertainty.
- Advanced recursive identification: Explore alternative recursive estimation schemes, such as Generalised Extended Recursive Least Squares, as the identification time is negligible compared to the MPC computation. Further, the use of instrumental variables could help reduce parameter correlation within the ARMAX model.
- Regularisation and kernel-based identification: Introduce regularisation or kernel methods to the recursive estimation process to include prior system knowledge and improve conditioning. The use of structured kernels such as the Diagonal/Correlated (DC) kernel could enforce smoothness or exponential decay in the Markov parameters.
- Hybrid modelling approaches: Investigate hybrid-SPC formulations in which part of the system dynamics is known a priori, consistent with recent developments in hybrid methods for Data-enabled Predictive Control (DeePC) [59]. This information could potentially be included in the regularisation term, or in other parts of the algorithm.
- Adaptive and directional forgetting: Incorporate input rate limiting to enhance robustness against abrupt reference changes, and implement directional forgetting factors to selectively forget information only in the current excited direction.
- Stability and convergence analysis: Develop formal theoretical guarantees for the closed-loop stability of ARX- and ARMAX-based SPC, as discussed in [2].
- Optimisation and computational efficiency: Investigate other convex solvers or structure-exploiting algorithms for the MPC optimisation step, for instance through sparse implementations for higher-order systems. This would further reduce computational time.
- Experimental validation and scalability: Conduct long-term experimental validation of ARX- and ARMAX-based SPC to evaluate adaptability, numerical stability, and robustness under varying operating conditions.

Appendix A

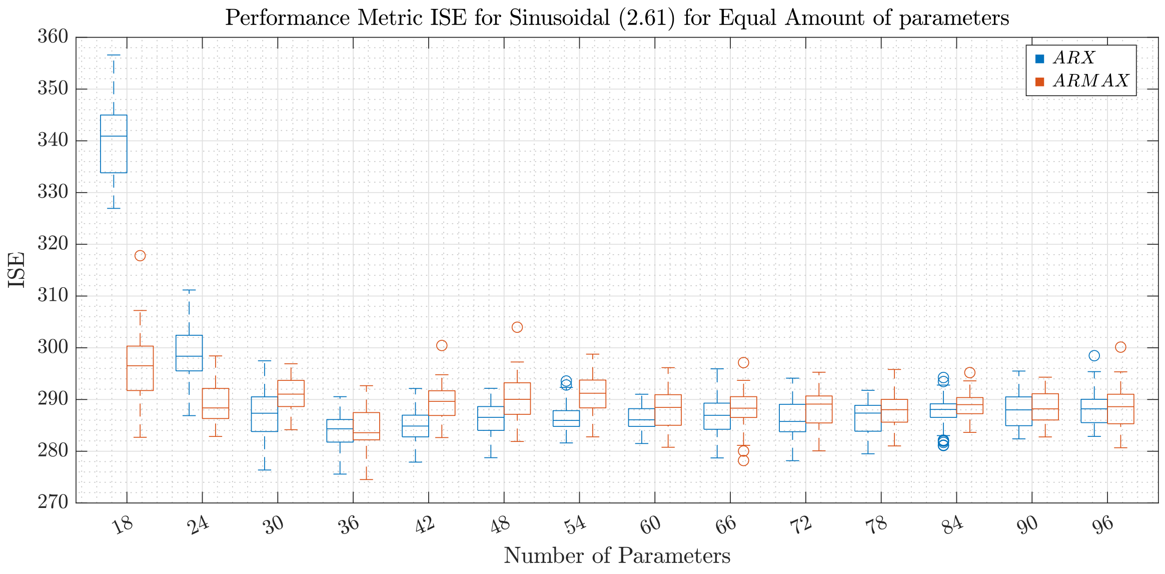
A-1 Additional Results: Inertia-Spring-Damper Model



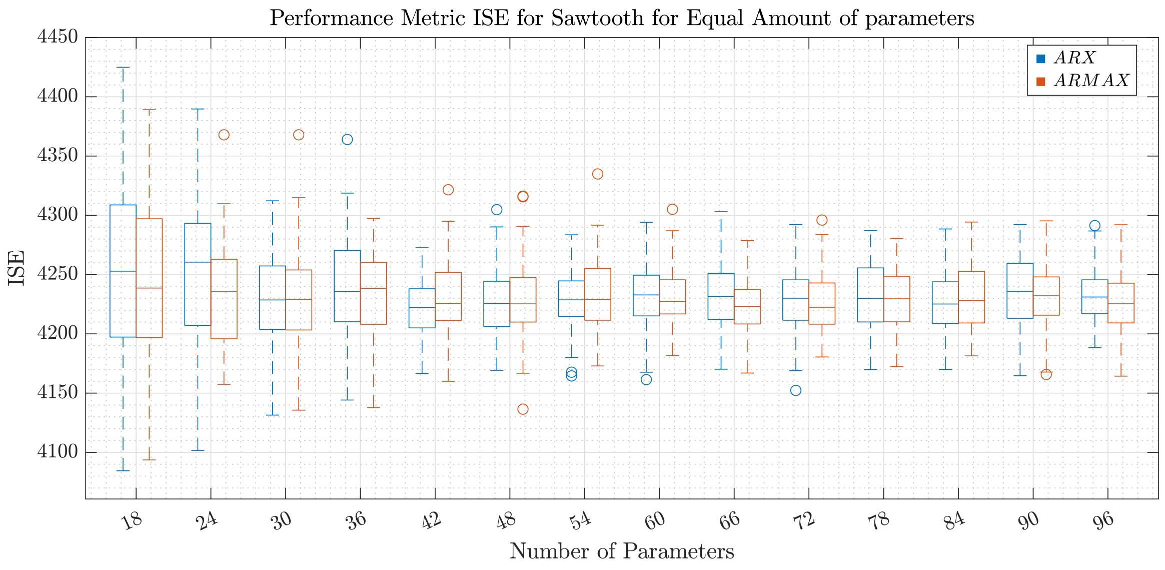
(a) Square (0.5 Hz)



(b) Sinusoidal (1 Hz)

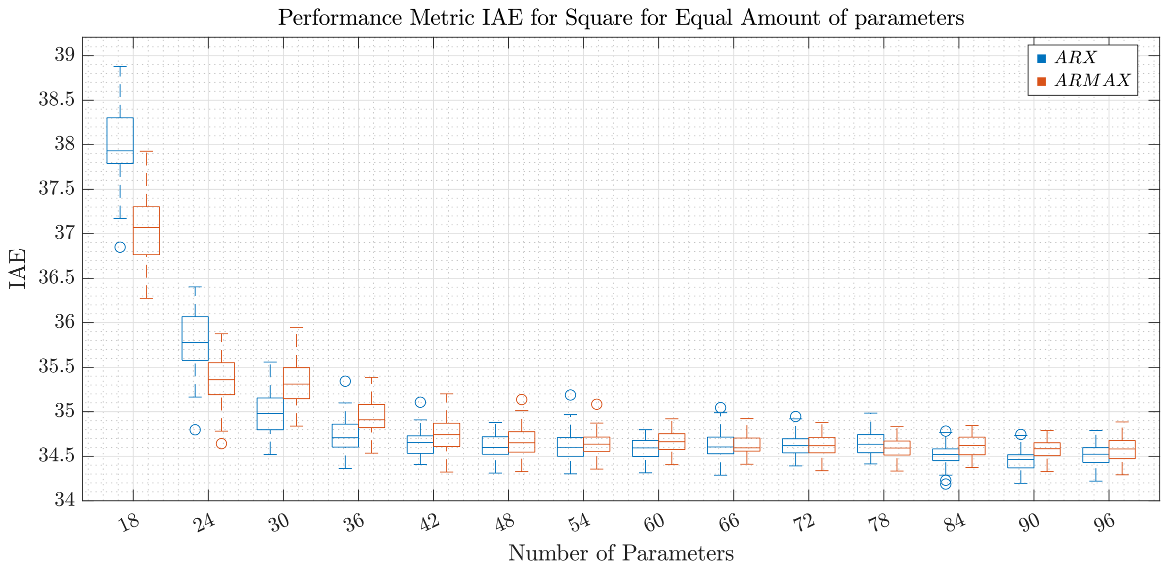


(c) Sinusoidal (2.61 Hz)

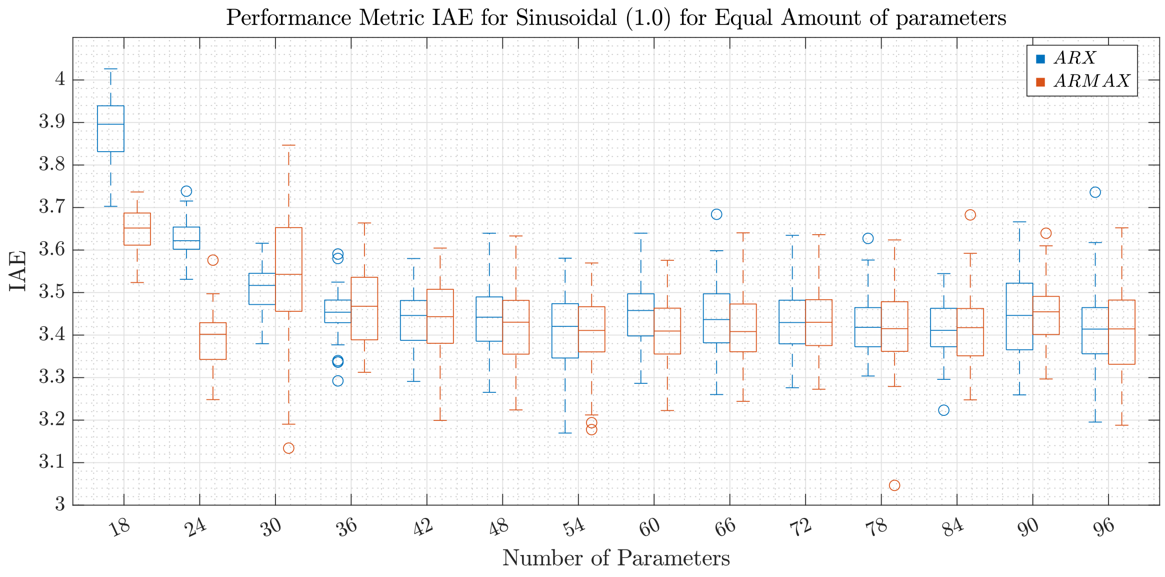


(d) Sawtooth (0.5 Hz)

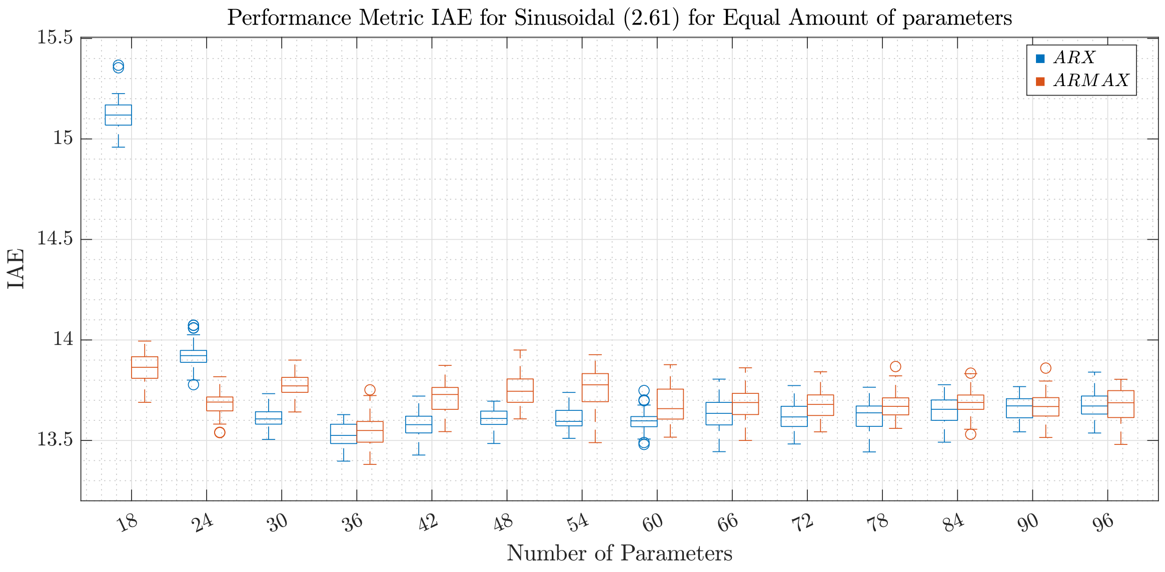
Figure A-1: Boxplots of integral squared error (ISE) for 50 Monte Carlo runs for ARX and ARMAX with the same number of parameters. This performance metric is calculated for the four different references: the square wave, the sinusoidal wave of 1 Hz, the sinusoidal wave of 2.61 Hz, and the sawtooth wave. The past window size p for ARX is the number of parameters divided by 2, while for ARMAX the past window size p is the number of parameters divide by 3. The differences at 18 and 24 parameters are significant, while for a larger number of parameters the metric difference becomes insignificant, and the behaviour becomes similar.



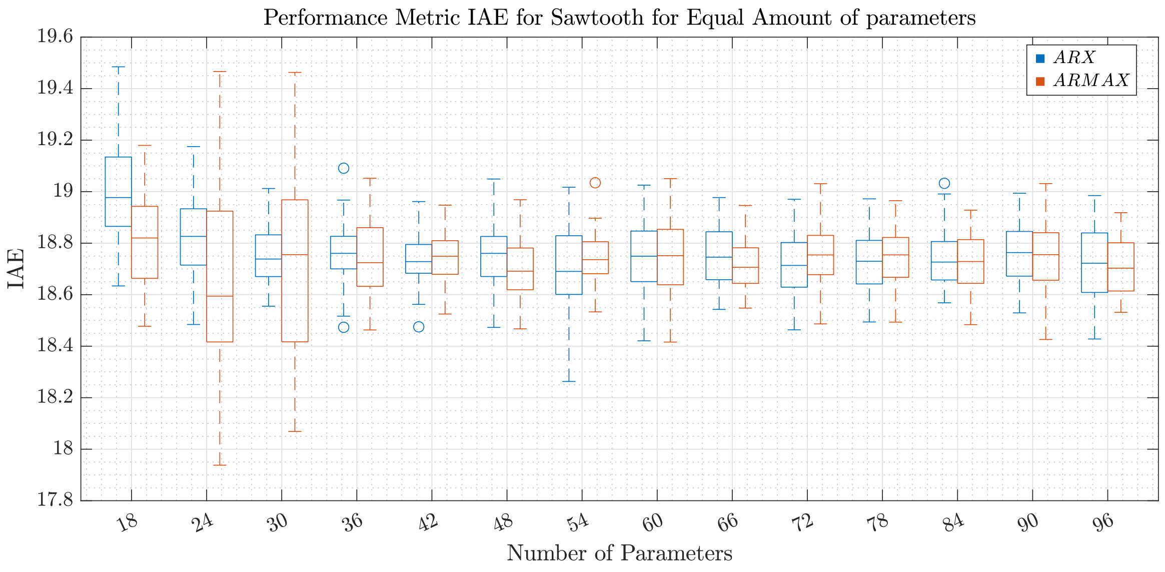
(a) Square (0.5 Hz)



(b) Sinusoidal (1 Hz)

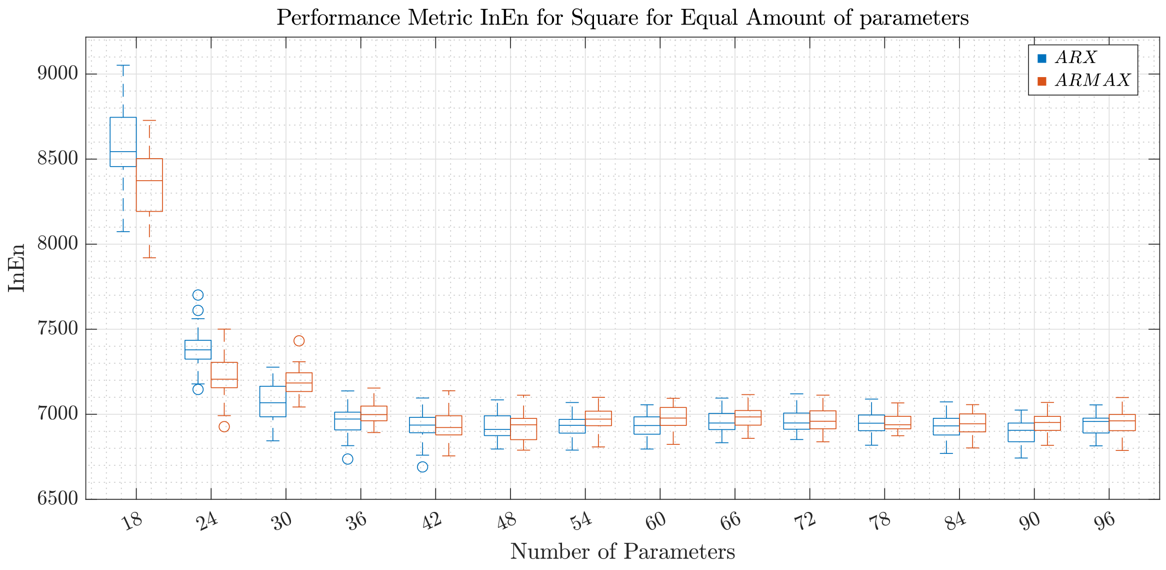


(c) Sinusoidal (2.61 Hz)

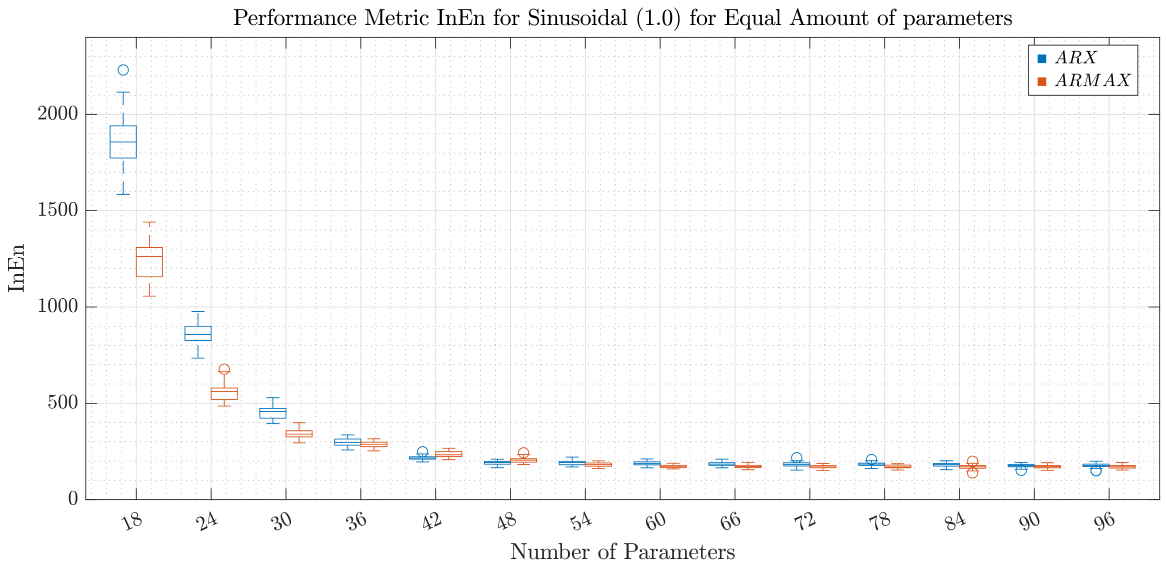


(d) Sawtooth (0.5 Hz)

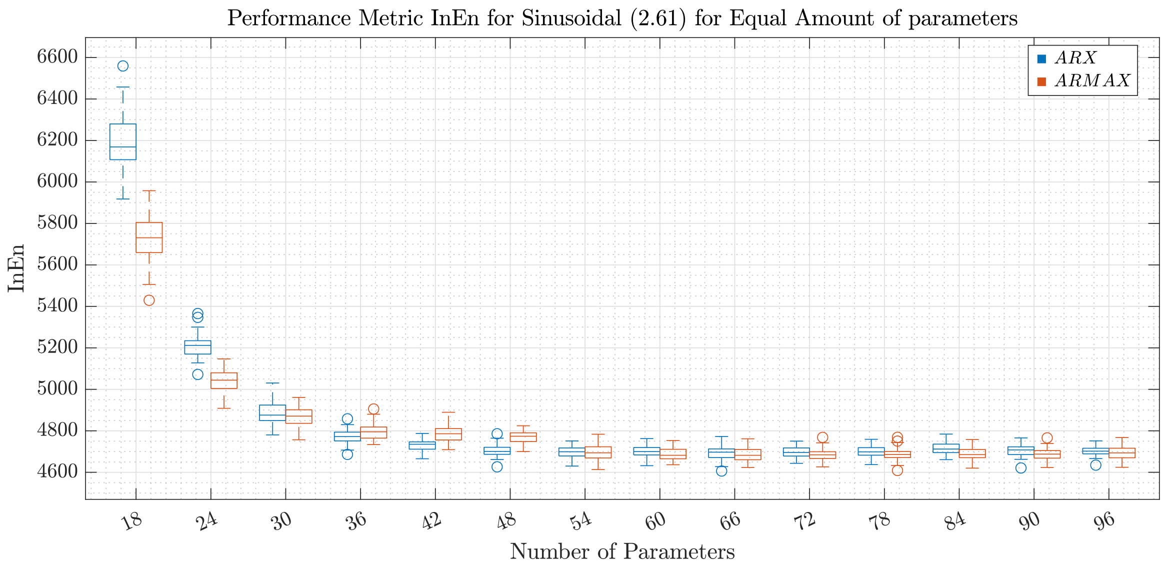
Figure A-2: Boxplots of integral absolute error (IAE) for 50 Monte Carlo runs for ARX and ARMAX with the same number of parameters. The differences in the metric across all options are rather small.



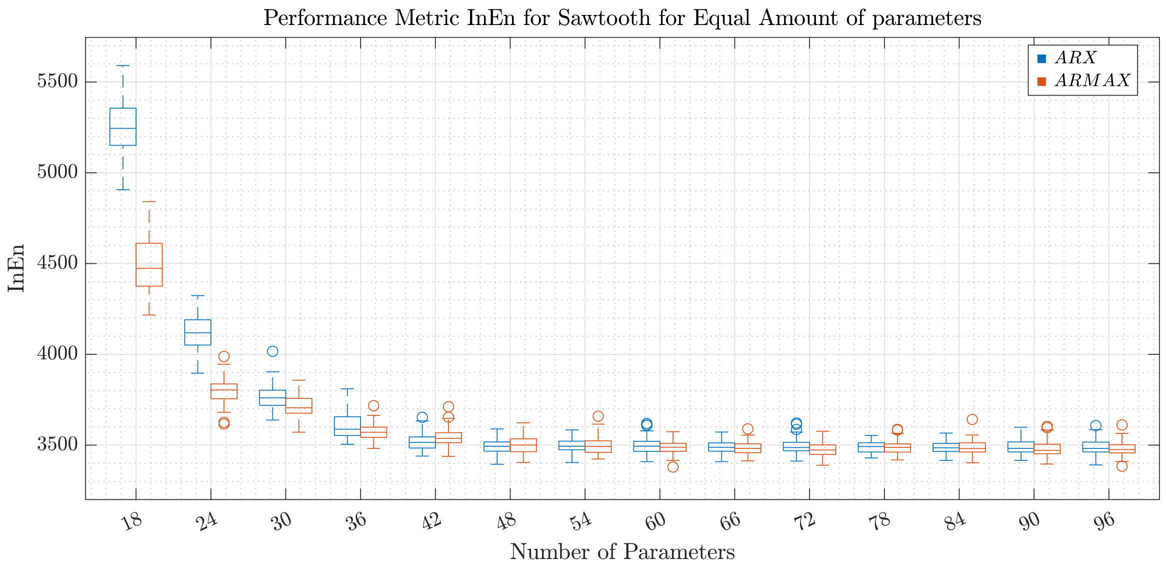
(a) Square (0.5 Hz)



(b) Sinusoidal (1 Hz)



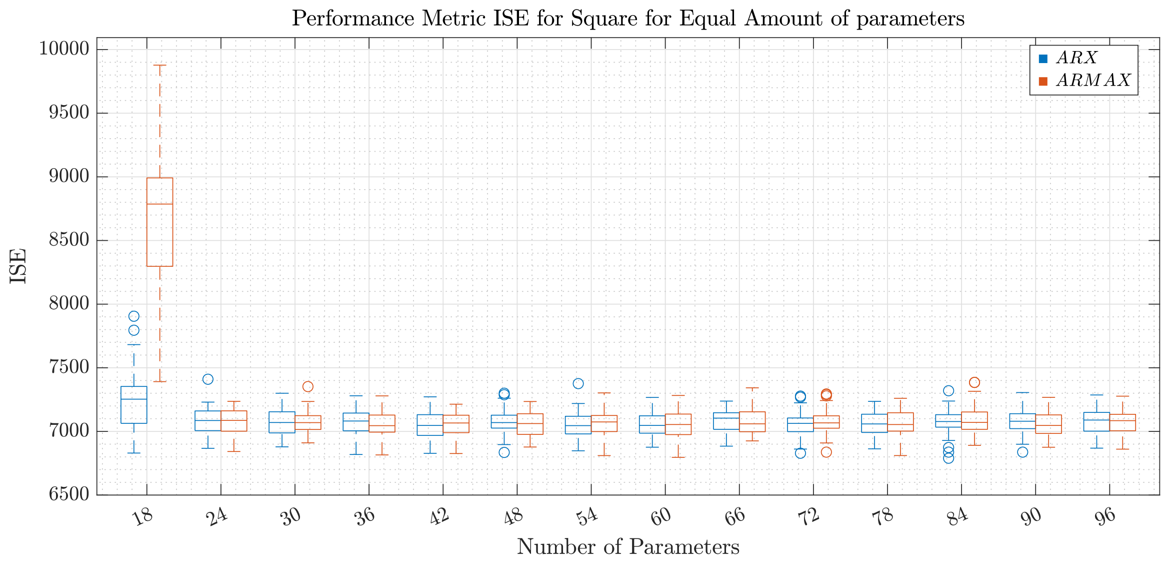
(c) Sinusoidal (2.61 Hz)



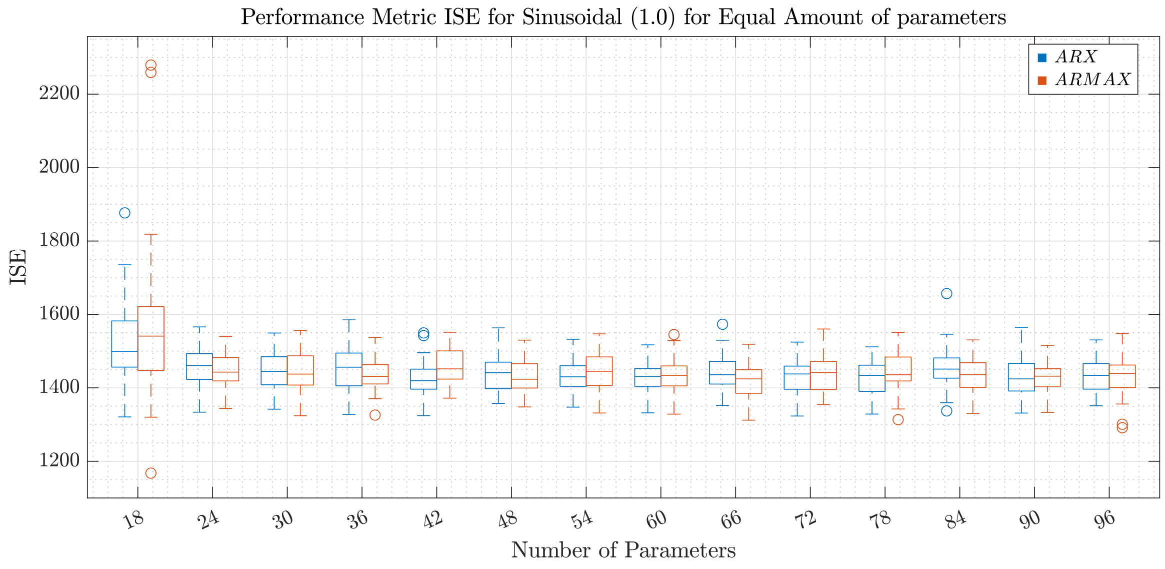
(d) Sawtooth (0.5 Hz)

Figure A-3: Boxplots of the input energy (InEn) for 50 Monte Carlo runs for ARX and ARMAX with the same number of parameters. The differences at 18 and 24 parameters are significant, while for a larger number of parameters the metric difference becomes insignificant, and both ARX and ARMAX converge to the same input energy.

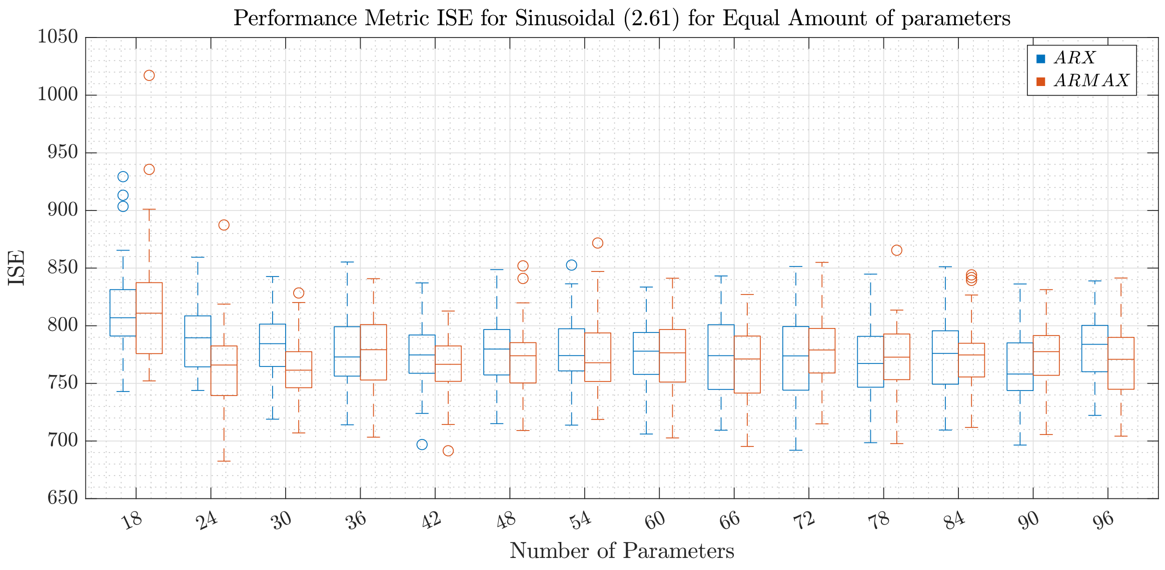
A-2 Additional Results: Fifth Order SISO Model



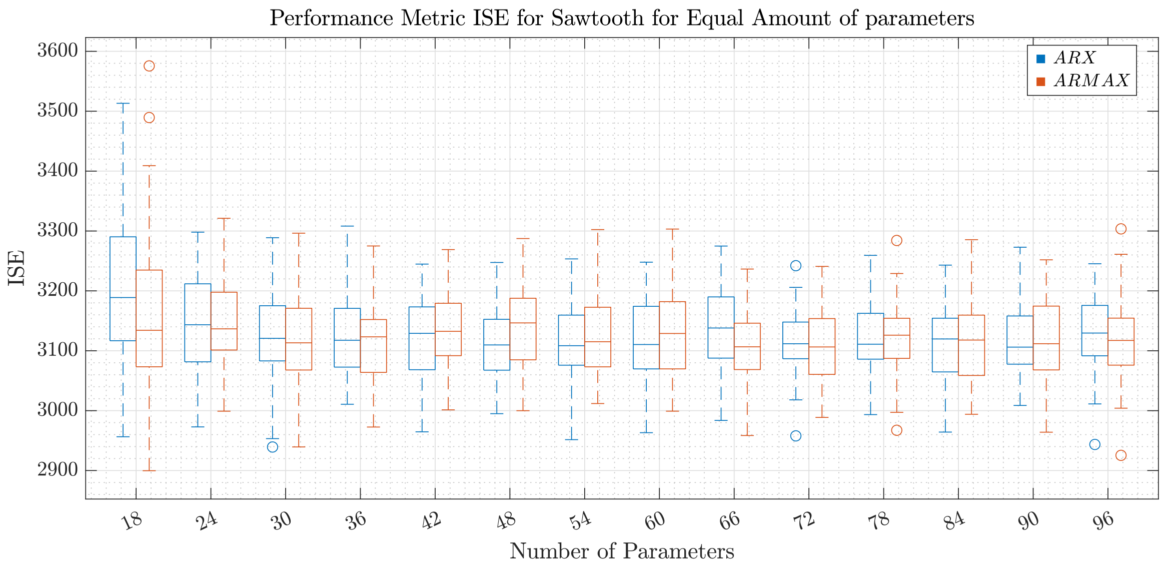
(a) Square (0.5 Hz)



(b) Sinusoidal (1 Hz)

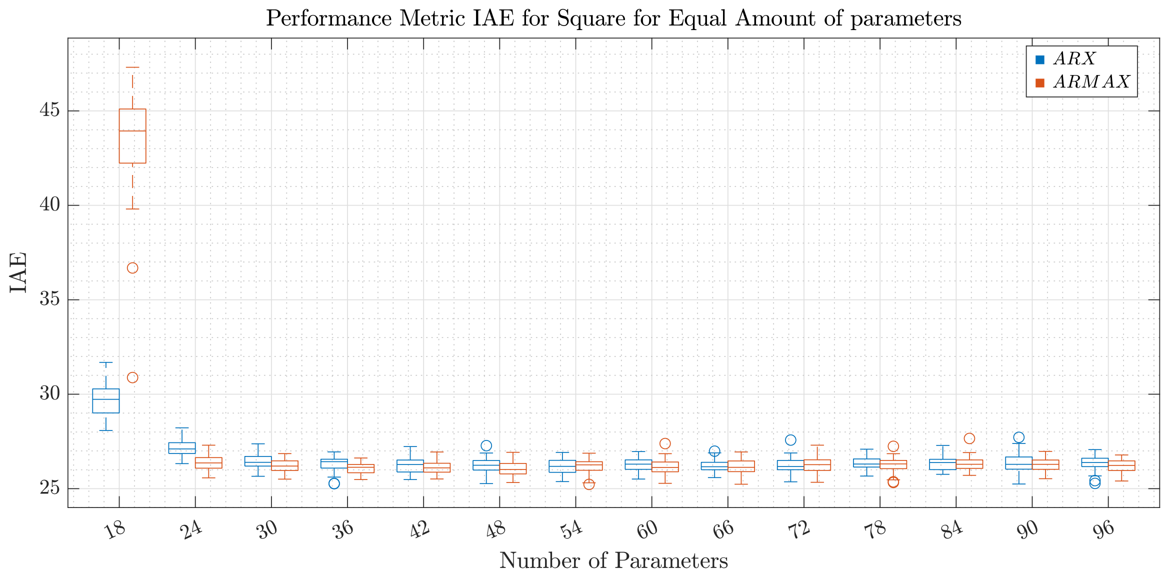


(c) Sinusoidal (2.61 Hz)

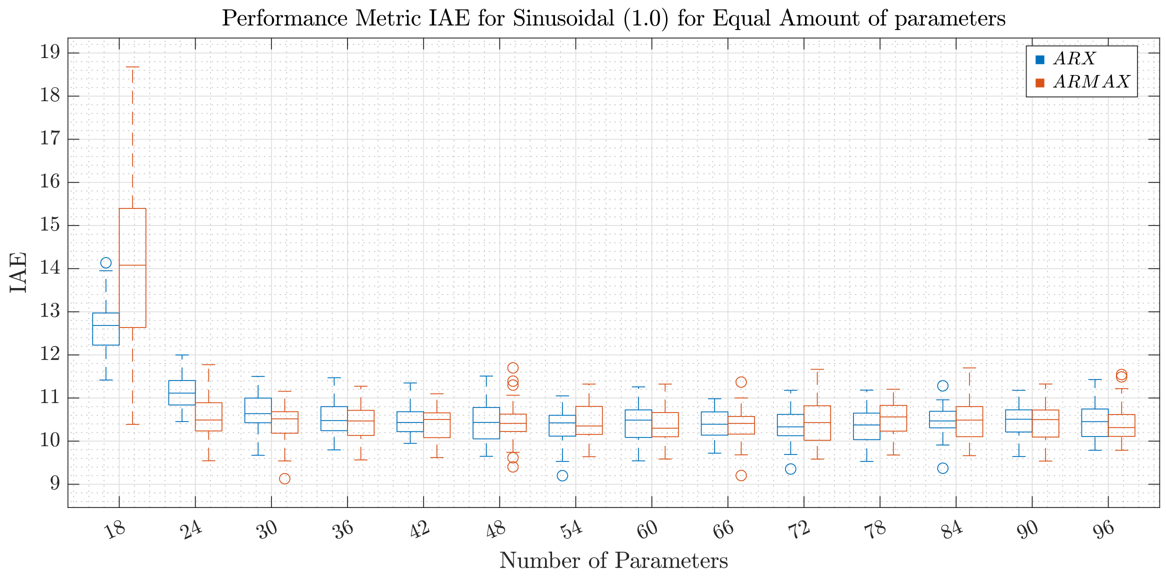


(d) Sawtooth (0.5 Hz)

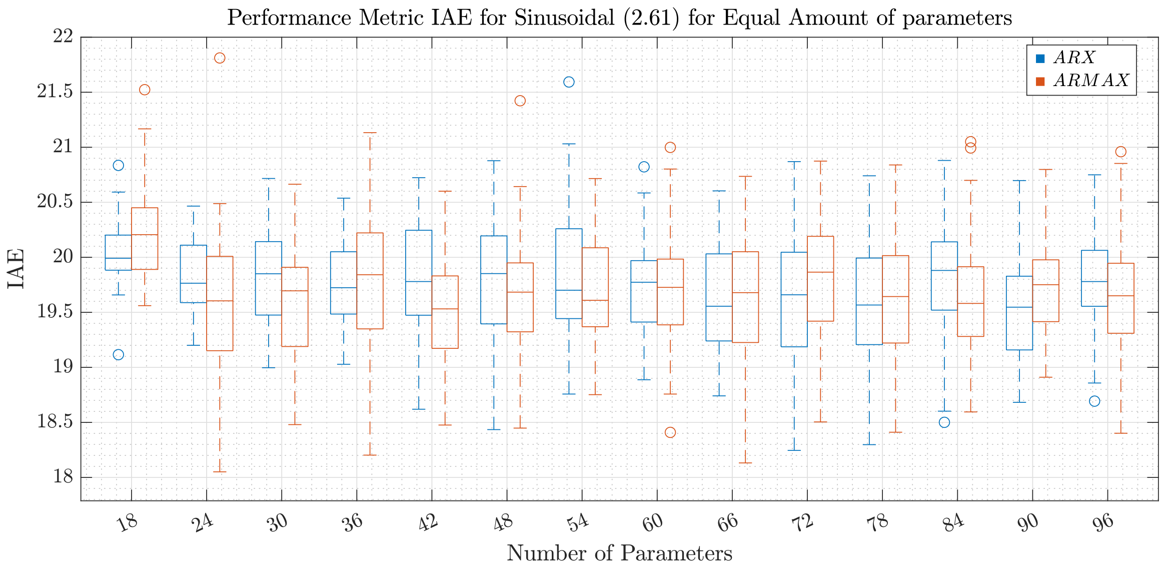
Figure A-4: Boxplots of integral squared error (ISE) for 50 Monte Carlo runs for ARX and ARMAX with the same number of parameters. This performance metric is calculated for the four different references: the square wave, the sinusoidal wave of 1 Hz, the sinusoidal wave of 2.61 Hz, and the sawtooth wave. The past window size p for ARX is the number of parameters divided by 2, while for ARMAX the past window size p is the number of parameters divide by 3. The differences at 18 parameters is significant, while for a larger number of parameters the metric difference becomes insignificant, and the behaviour becomes similar.



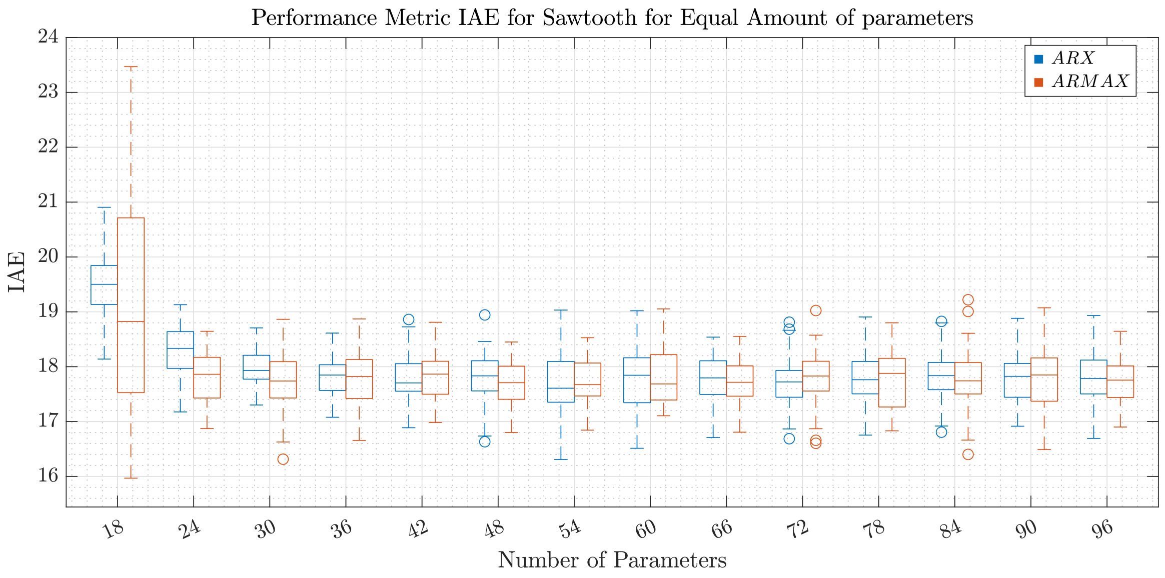
(a) Square (0.5 Hz)



(b) Sinusoidal (1 Hz)

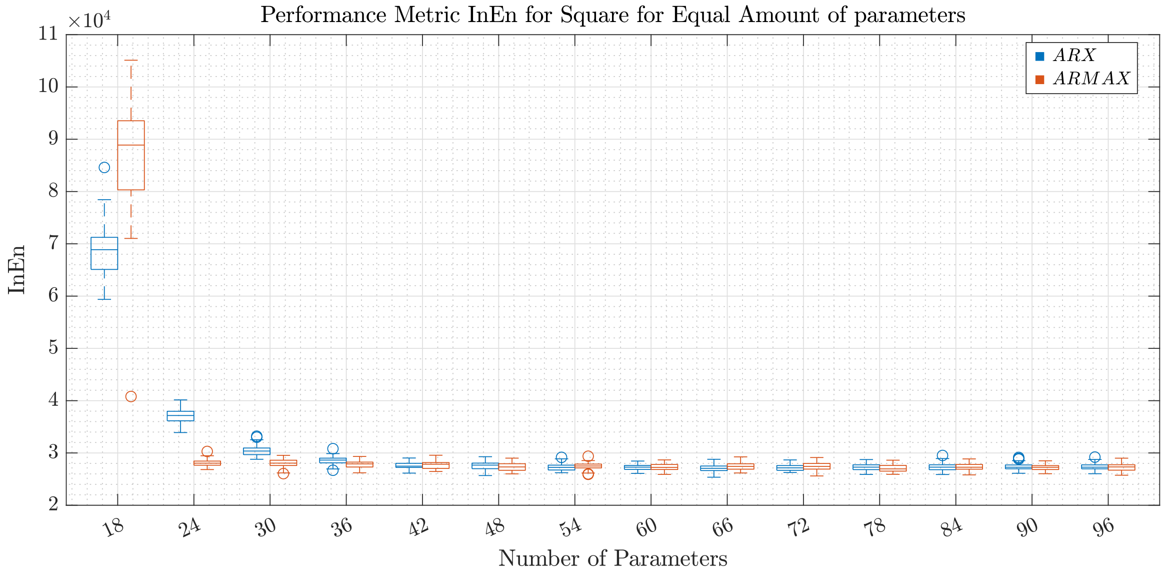


(c) Sinusoidal (2.61 Hz)

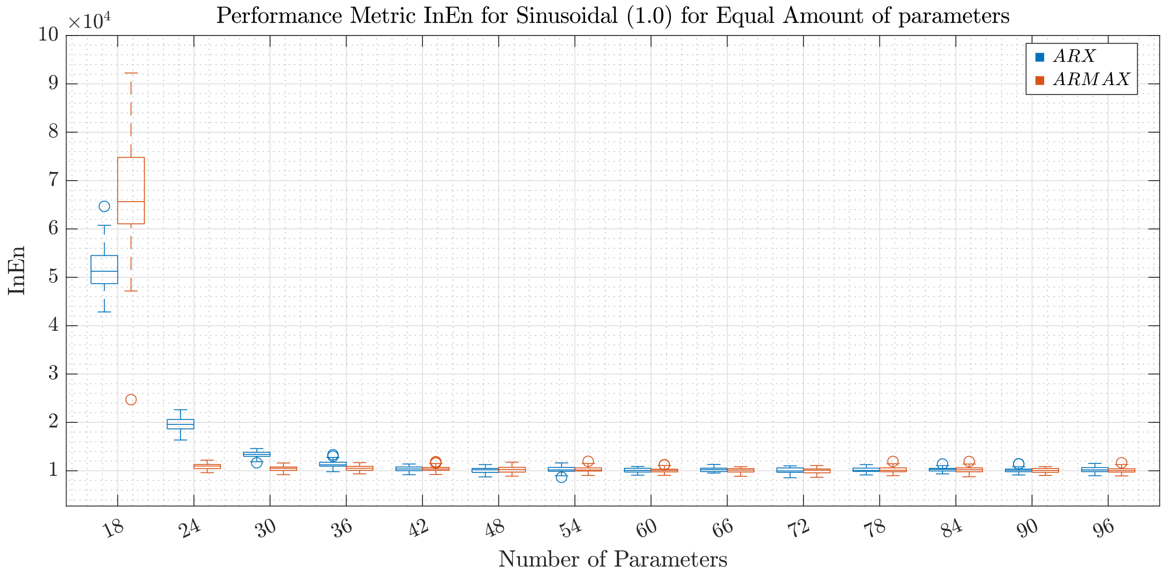


(d) Sawtooth (0.5 Hz)

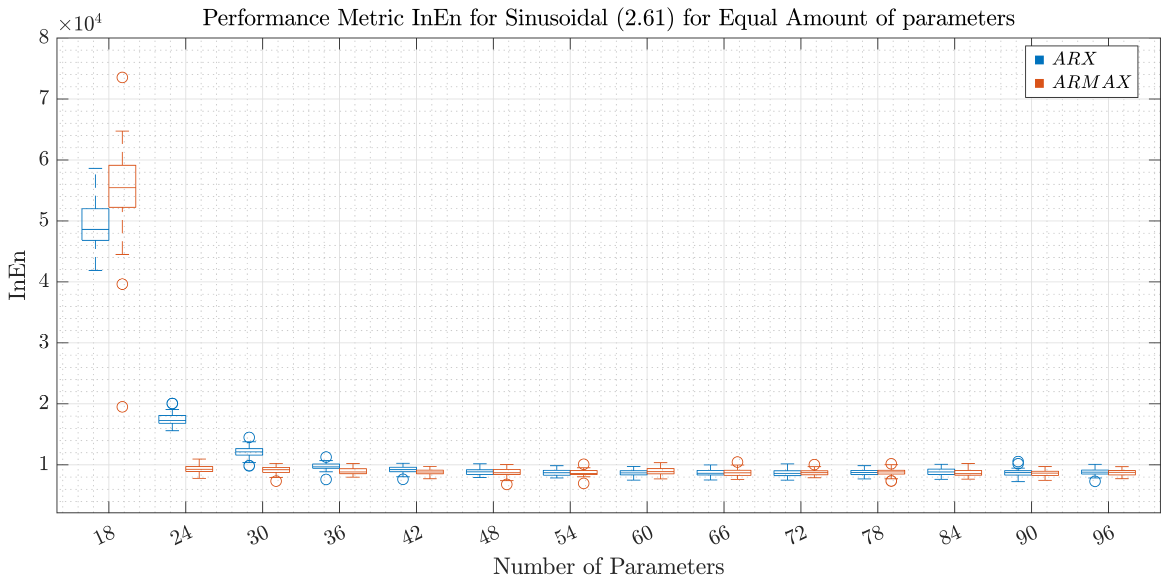
Figure A-5: Boxplots of integral absolute error (IAE) for 50 Monte Carlo runs for ARX and ARMAX with the same number of parameters. The difference at 18 parameters is large, while for a large number of parameters the behaviour is similar.



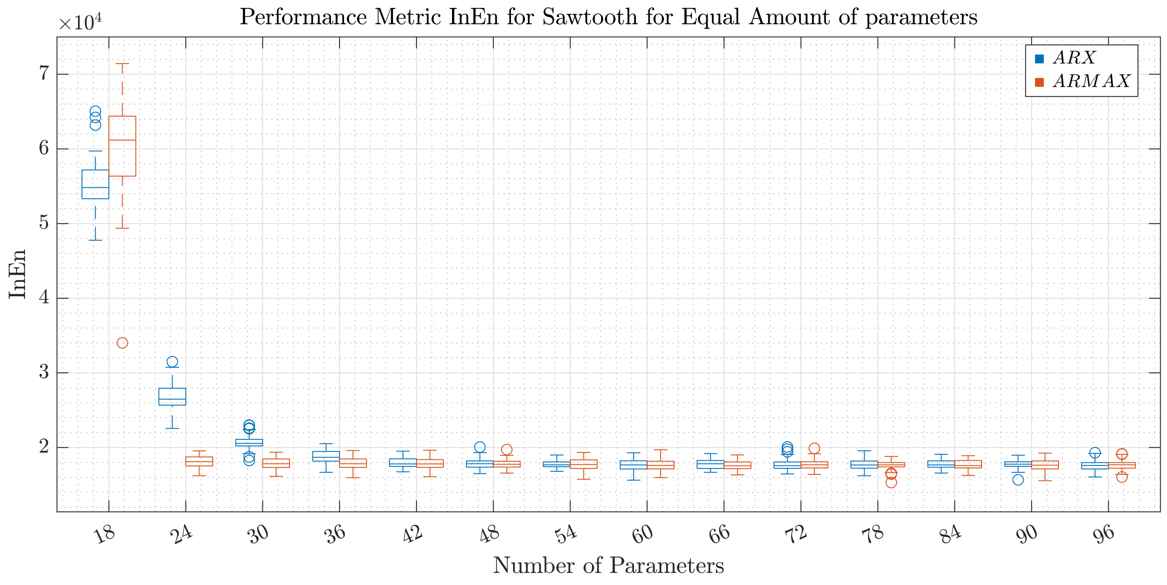
(a) Square (0.5 Hz)



(b) Sinusoidal (1 Hz)



(c) Sinusoidal (2.61 Hz)



(d) Sawtooth (0.5 Hz)

Figure A-6: Boxplots of the input energy (InEn) for 50 Monte Carlo runs for ARX and ARMAX with the same number of parameters. The differences at 18, 24 and 30 parameters are significant, while for a larger number of parameters the metric difference becomes insignificant, and both ARX and ARMAX converge to the same input energy.

Bibliography

[1] Brian DO Anderson and John B Moore. *Optimal Filtering.* Courier Corporation, 2005.

[2] Julian Berberich and Frank Allgöwer. An overview of systems-theoretic guarantees in data-driven model predictive control, 2024.

[3] V. Breschi, M. Fabris, S. Formentin, and A. Chiuso. Uncertainty-aware data-driven predictive control in a stochastic setting. *IFAC-PapersOnLine*, 56(2):10083–10088, 2023.

[4] Eduardo F. Camacho and Carlos Bordons. *"Constrained MPC"*, pages 167–207. Springer London, London, 1999.

[5] Alessandro Chiuso and Giorgio Picci. Consistency analysis of some closed-loop subspace identification methods. *Automatica*, 41(3):377–391, 2005. Data-Based Modelling and System Identification.

[6] Rogier Dinkla, Sebastiaan P. Mulders, Jan-Willem van Wingerden, and Tom Oomen. Closed-loop Aspects of Data-Enabled Predictive Control. *IFAC-PapersOnLine*, 56(2):1388–1393, 2023. 22nd IFAC World Congress.

[7] Jianfei Dong and Michel Verhaegen. On the equivalence of closed-loop subspace predictive control with LQG. In *2008 47th IEEE Conference on Decision and Control*, pages 4085–4090, 2008.

[8] Florian Dörfler. Data-Driven Control: Part Two of Two: Hot Take: Why not go with Models? *IEEE Control Systems Magazine*, 43(6):27–31, 2023.

[9] Florian Dörfler, Jeremy Coulson, and Ivan Markovsky. Bridging Direct and Indirect Data-Driven Control Formulations via Regularizations and Relaxations. *IEEE Transactions on Automatic Control*, 68(2):883–897, 2023.

[10] E. Elokda, J. Coulson, P. N. Beuchat, J. Lygeros, and F. Dörfler. Data-enabled predictive control for quadcopters. *International Journal of Robust and Nonlinear Control*, 31(18):8916–8936, 2021.

[11] Wouter Favoreel, Bart De Moor, and Michel Gevers. SPC: Subspace Predictive Control. *IFAC Proceedings Volumes*, 32(2):4004–4009, 1999. 14th IFAC World Congress 1999, Beijing, China, 5-9 July.

[12] Miguel Galrinho, Niklas Everitt, and Håkan Hjalmarsson. ARX modeling of unstable linear systems. *Automatica*, 75:167–171, 2017.

[13] William H. Greene. *Econometric Analysis.* Pearson Education, fifth edition, 2003.

[14] William W. Hager. Updating the Inverse of a Matrix. *SIAM Review*, 31(2):221–239, 1989.

[15] Per C Hansen. The truncated SVD as a method for regularization. Technical report, Stanford University, Stanford, CA, USA, 1986.

[16] Tor Aksel N. Heirung, Joel A. Paulson, Jared OLeary, and Ali Mesbah. Stochastic model predictive control how does it work? *Computers & Chemical Engineering*, 114:158–170, 2018. FOCAPO/CPC 2017.

[17] Zhong-Sheng Hou and Zhuo Wang. From model-based control to data-driven control: Survey, classification and perspective. *Information Sciences*, 235:3–35, 2013. Data-based Control, Decision, Scheduling and Fault Diagnostics.

[18] I. Houtzager, J.W. van Wingerden, and M. Verhaegen. Fast-array Recursive Closed-loop Subspace Model Identification. *IFAC Proceedings Volumes*, 42(10):96–101, 2009. 15th IFAC Symposium on System Identification.

[19] Ivo Houtzager, Jan-Willem van Wingerden, and Michel Verhaegen. VARMAX-based closed-loop subspace model identification. In *Proceedings of the 48h IEEE Conference on Decision and Control (CDC) held jointly with 2009 28th Chinese Control Conference*, pages 3370–3375, Dec 2009.

[20] Rob J. Hyndman and George Athanasopoulos. *Forecasting: Principles and Practice.* OTexts, 2nd edition, 2018.

[21] Magnus Jansson. A NEW SUBSPACE IDENTIFICATION METHOD FOR OPEN AND CLOSED LOOP DATA. *IFAC Proceedings Volumes*, 38(1):500–505, 2005. 16th IFAC World Congress.

[22] Yue Ju, Biqiang Mu, Lennart Ljung, and Tianshi Chen. Asymptotic Theory for Regularized System Identification Part I: Empirical Bayes Hyperparameter Estimator. *IEEE Transactions on Automatic Control*, 68(12):7224–7239, 2023.

[23] R. E. Kalman. A New Approach to Linear Filtering and Prediction Problems. *Journal of Basic Engineering*, 82(1):35–45, 03 1960.

[24] H. K. Khalil. *Nonlinear Control.* Pearson, 2014.

[25] S. K. Laha. A sparse approach to transfer function estimation via Least Absolute Shrinkage and Selection Operator. *IFAC Journal of Systems and Control*, 31:100299, 2025.

[26] M. Lazar and P. C. N. Verheijen. Generalized Data-Driven Predictive Control: Merging Subspace and Hankel Predictors. *Mathematics*, 11(9), 2023.

[27] Zhe Li, Haoda Wang, Qiu Fang, and Yaonan Wang. A data-driven subspace predictive control method for air-cooled data center thermal modelling and optimization. *Journal of the Franklin Institute*, 360(5):3657–3676, 2023.

[28] Lennart Ljung. *System identification (2nd ed.): theory for the user.* Prentice Hall PTR, USA, 1999.

[29] Lennart Ljung. "System Identification: An Overview". In John Baillieul and Tariq Samad, editors, *Encyclopedia of Systems and Control*, pages 1–20. Springer London, London, 2013.

[30] Lennart Ljung and Bo Wahlberg. Asymptotic Properties of the Least-Squares Method for Estimating Transfer Functions and Disturbance Spectra. *Advances in Applied Probability*, 24(2):412–440, 1992.

[31] L Nugroho and Rini Akmeliawati. Comparison of black-grey-white box approach in system identification of a flight vehicle. *Journal of Physics: Conference Series*, 1130:012024, 11 2018.

[32] Gianluigi Pillonetto, Tianshi Chen, Alessandro Chiuso, Giuseppe De Nicolao, and Lennart Ljung. *Regularized system identification. Learning dynamic models from data.* Communications and Control Engineering. Springer, Cham, Switzerland, 2022.

[33] Gianluigi Pillonetto, Francesco Dinuzzo, Tianshi Chen, Giuseppe De Nicolao, and Lennart Ljung. Kernel methods in system identification, machine learning and function estimation: A survey. *Automatica*, 50(3):657–682, 2014.

[34] Farzin Piltan, Shahnaz Tayebihaghighi, and Nasri Sulaiman. Comparative study between ARX and ARMAX system identification. *International Journal of Intelligent Systems and Applications*, 2:25–34, 01 2017.

[35] Quanser. Multi-DOF Torsion, n.d. Accessed: Oct. 14, 2025.

[36] Saša Raković. Robust model-predictive control. In John Baillieul and Tariq Samad, editors, *Encyclopedia of Systems and Control*, pages 1–11. Springer London, London, 2013.

[37] J.B. Rawlings, D.Q. Mayne, and M. Diehl. *Model Predictive Control: Theory, Computation, and Design.* Nob Hill Publishing, 2017.

[38] Ali H. Sayed. *Adaptive Filters.* Wiley-IEEE Press, 2008.

[39] Josef Shinar and Turetsky Vladimir. What happens when certainty equivalence is not valid?: Is there an optimal estimator for terminal guidance? *Annual Reviews in Control*, 27(2):119–130, 2003.

[40] Jinya Su, Baibing Li, and Wen-Hua Chen. On existence, optimality and asymptotic stability of the Kalman filter with partially observed inputs. *Automatica*, 53:149–154, 2015.

[41] Jinming Sun, Yanqiu Huang, Wanli Yu, and Alberto Garcia-Ortiz. Nonlinear System Identification: Prediction Error Method vs Neural Network. In *2021 10th International Conference on Modern Circuits and Systems Technologies (MOCAST)*, pages 1–4, 2021.

[42] J. Kenneth Tay, Balasubramanian Narasimhan, and Trevor Hastie. Elastic Net Regularization Paths for All Generalized Linear Models, 2021.

[43] Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 58(1):267–288, 1996.

[44] Fahim Uddin, Lemma Dendena Tufa, Suhaib Mohd Taha Yousif, and Abdulhalim Shah Maulud. Comparison of ARX and ARMAX Decorrelation Models for Detecting Model-Plant Mismatch. *Procedia Engineering*, 148:985–991, 2016. Proceeding of 4th International Conference on Process Engineering and Advanced Materials (ICPEAM 2016).

[45] Vineet Vajpayee, Siddhartha Mukhopadhyay, and Akhilanand Pati Tiwari. Data-Driven Subspace Predictive Control of a Nuclear Reactor. *IEEE Transactions on Nuclear Science*, 65(2):666–679, 2018.

[46] Gijs van der Veen and Jan-Willem van Wingerden. Subspace predictive control of flexible structures actuated by piezoelectric elements. In *2024 IEEE Conference on Control Technology and Applications (CCTA)*, pages 316–322, 2024.

[47] Peter Van Overschee and Bart De Moor. N4SID: Subspace algorithms for the identification of combined deterministic-stochastic systems. *Automatica*, 30(1):75–93, 1994. Special issue on statistical signal processing and control.

[48] Peter Van Overschee and Bart De Moor. *Subspace Identification for Linear Systems: Theory - Implementation - Applications.* Kluwer Academic Publishers, Dordrecht, 1996.

[49] Jan-Willem van Wingerden, Sebastiaan P. Mulders, Rogier Dinkla, Tom Oomen, and Michel Verhaegen. Data-enabled predictive control with instrumental variables: the direct equivalence with subspace predictive control. In *2022 IEEE 61st Conference on Decision and Control (CDC)*, pages 2111–2116, 2022.

[50] Gijs Veen, J. W. Wingerden, Marco Bergamasco, Marco Lovera, and Michel Verhaegen. Closed-loop subspace identification methods: An overview. *Control Theory & Applications, IET*, 7:1339–1358, 07 2013.

[51] M. Verhaegen and V. Verdult. *Filtering and system identification: a least squares approach.* Cambridge University Press, 2007.

[52] M.H. Verhaegen. Round-off error propagation in four generally-applicable, recursive, least-squares estimation schemes. *Automatica*, 25(3):437–444, 1989.

[53] Michel Verhaegen and Patrick Dewilde. Subspace model identification Part 1: The output-error state-space model identification class of algorithms. *International Journal of Control*, 56(5):1187–1210, 1992.

[54] P. C. N. Verheijen, G. R. Gonçalves da Silva, and M. Lazar. Data-driven predictive control with estimated prediction matrices and integral action, 2021. arXiv preprint arXiv:2104.04972.

[55] P.C.N. Verheijen, V. Breschi, and M. Lazar. Handbook of linear data-driven predictive control: Theory, implementation and design. *Annual Reviews in Control*, 56:100914, 2023.

[56] Chris Verhoek, Hossam S. Abbas, Roland Tóth, and Sofie Haesaert. Data-Driven Predictive Control for Linear Parameter-Varying Systems. *IFAC-PapersOnLine*, 54(8):101–108, 2021. 4th IFAC Workshop on Linear Parameter Varying Systems LPVS 2021.

[57] Alexey Voskov. QR decomposition for the least squares method: theory and practice. *International Journal of Mathematical Education in Science and Technology*, 55:1–18, 12 2022.

[58] B. Wahlberg. System identification using high-order models, revisited. In *Proceedings of the 28th IEEE Conference on Decision and Control,*, pages 634–639 vol.1, 1989.

[59] Jeremy D. Watson. Hybrid Data-Enabled Predictive Control: Incorporating Model Knowledge into the DeePC, 2025. arXiv preprint arXiv:2502.12467.

[60] Jan C. Willems, Paolo Rapisarda, Ivan Markovsky, and Bart L.M. De Moor. A note on persistency of excitation. *Systems & Control Letters*, 54(4):325–329, 2005.

[61] Ai-Guo Wu, Shi-Wei Liu, and Rui-Qi Dong. Latest-Estimation Based Hierarchical Recursive Extended Least Squares algorithm for ARMAX model. In *2016 35th Chinese Control Conference (CCC)*, pages 273–278, 2016.

[62] Ai-Guo Wu and Zhi-Guang Wang. Modified recursive extended least squares identification algorithms. In *2013 25th Chinese Control and Decision Conference (CCDC)*, pages 1562–1567, 2013.

[63] Kaixiang Zhang, Kaian Chen, Xinfan Lin, Yusheng Zheng, Xunyun Yin, Xiaosong Hu, Ziyou Song, and Zhaojian Li. Data-Enabled Predictive Control for Fast Charging of Lithium-Ion Batteries with Constraint Handling, 2023.

[64] Yucai Zhu and Ton Backx. *Identification of Multivariable Industrial Processes: for Simulation, Diagnosis and Control.* Advances in Industrial Control. Springer, 1993.