POLITECNICO
MILANO 1863

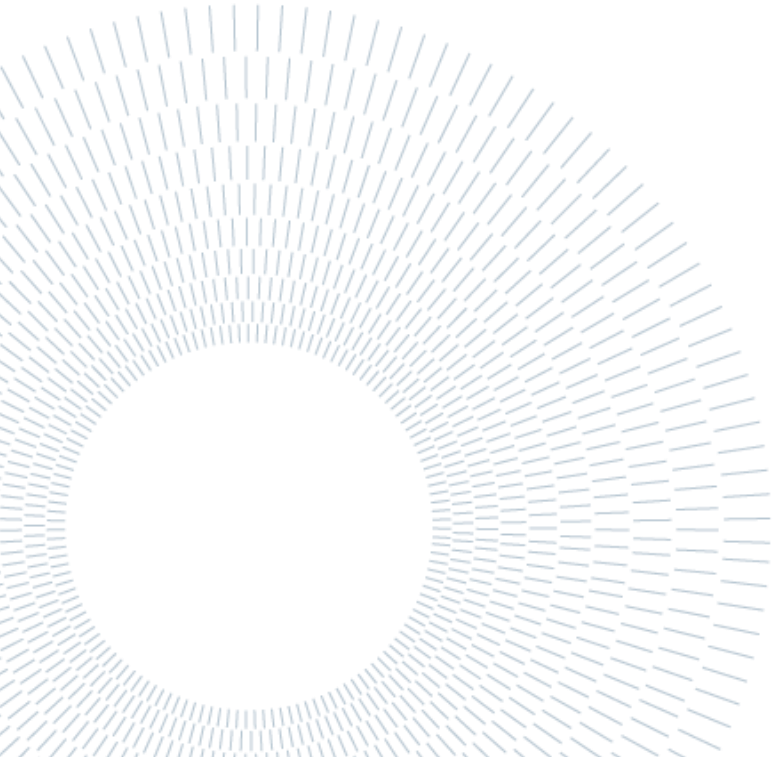
SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

Data-Driven Moment Matching Model Order Reduction for Energy System Models

TESI DI LAUREA MAGISTRALE IN
AUTOMATION AND CONTROL ENGINEERING
INGEGNERIA DELL'AUTOMAZIONE

Author: Marco Piazzini

Student ID: 10720785
Advisor: Prof. Francesco Casella
Co-advisor: Dr. Matteo Luigi De Pascali
Academic Year: 2025-26



Summary

a Sebastiano

Abstract

Large-scale physics-based models are widely employed to describe complex systems; however, their high dimensionality often limits their applicability in simulation, control, and optimization tasks. In particular, the computational burden associated with high-order models is frequently incompatible with the time scale requirements of the underlying system dynamics, making the implementation of advanced control strategies, such as Model Predictive Control (MPC), impractical. In such contexts, the computational times required by full-order models exceed the characteristic time scales of the system dynamics, thereby preventing their use within real time control strategies. Classical model order reduction (MOR) techniques rely on full knowledge of the system equations, which may be unavailable or impractical. Recently, a data-driven moment matching approach has been proposed to construct reduced-order models directly from input-output data. Despite its theoretical appeal, the applicability of this approach to large-scale systems has not yet been systematically investigated. This thesis evaluates the effectiveness of the data-driven moment matching method on two high order models, namely a heat exchanger-turbine system of order 80 and a Solid Oxide Fuel Cell (SOFC) system of order 268, both developed by Dipartimento di Elettronica Informazione e Bioingegneria (DEIB) of Politecnico di Milano. The method is first validated in single-input single-output (SISO) configurations and consequently extended to multi-input multi-output (MIMO) systems. For the MIMO case, a non data-driven strategy based on the averaging of moment information computed at operating points of interest for the analysis and reduction is proposed. Numerical results show that the resulting reduced-order models achieve a significant reduction in dimensionality while accurately preserving the dominant nonlinear dynamic behavior and the singular value structure of the original systems. This order reduction further acts as an enabling factor for the adoption of model-based predictive control strategies, making the use of MPC feasible in scenarios where the employment of full-order models would be computationally prohibitive.

Keywords: Moment matching, Model order reduction, MOR, Heat exchanger-turbine, Solid Oxide Fuel Cell, SOFC.

Abstract in italiano

I modelli fisici di grandi dimensioni sono comunemente impiegati per descrivere i sistemi complessi; tuttavia, la loro elevata dimensione limita spesso l'utilizzo in attività di simulazione, controllo e ottimizzazione. In particolare, il costo computazionale associato ai modelli di alto ordine risulta frequentemente incompatibile con i requisiti temporali dei sistemi dinamici di interesse, rendendo di fatto impraticabile l'implementazione di strategie di controllo avanzate, quali il Model Predictive Control (MPC). In tali contesti, i tempi di calcolo richiesti dall'impiego del modello completo superano le scale temporali della dinamica del sistema, precludendone l'utilizzo all'interno di strategie di controllo in tempo reale. Le tecniche classiche di riduzione d'ordine dei modelli (MOR) richiedono una conoscenza completa delle equazioni del sistema, che può risultare non disponibile o poco pratica. Recentemente, è stato proposto un approccio di moment matching data-driven per la costruzione di modelli ridotti direttamente a partire dai dati di ingresso e uscita. Nonostante il suo interesse teorico, l'applicabilità di tale approccio a sistemi di grandi dimensioni non è stata ancora analizzata in modo sistematico. Questa tesi valuta l'efficacia del metodo di moment matching data-driven su due modelli ad elevato ordine di complessità, rappresentativi rispettivamente di un sistema scambiatore di calore-turbina di ordine 80 e di una cella a combustibili a ossidi solidi (SOFC) di ordine 268, entrambi sviluppati dal Dipartimento di Elettronica Informazione e Bioingegneria (DEIB) del Politecnico di Milano. Il metodo viene inizialmente validato in configurazioni a singolo ingresso e singola uscita (SISO) e successivamente su configurazioni dei modelli multi-ingresso multi-uscita (MIMO). Per quest'ultimo, viene proposta una strategia non data-driven basata sulla media delle informazioni di momento calcolate in corrispondenza dei diversi punti di lavoro di interesse per analisi e la riduzione. I risultati ottenuti mostrano che i modelli ridotti ottenuti consentono una significativa riduzione della dimensione, preservando al contempo il comportamento dinamico nonlineare dominante alle frequenze di interesse e la struttura dei valori singolari dei sistemi originali. Tale riduzione d'ordine costituisce un elemento abilitante per l'impiego di strategie di controllo predittivo model based, rendendo possibile l'utilizzo di MPC su sistemi per i quali l'impiego del modello completo risulterebbe proibitivo dal punto di vista computazionale.

Parole chiave: Moment matching, Riduzione d'ordine, MOR, Scambiatore di calore e turbina, Cella combustible a ossidi solidi, SOFC.

Preface

"Human life is like a pendulum swinging incessantly between pain and boredom, with fleeting and, what is more, illusory intervals of pleasure and joy"

*(*Arthur Schopenhauer, *The World as Will and Representation)*

This thesis has been developed as part of the Master of Science degree in Automation and Control Engineering at the Politecnico di Milano. The work is located within the broad research area of modeling, simulation, and control of complex dynamical systems, with a particular focus on MOR techniques for large-scale industrial models. The research activity presented in this thesis is motivated by advances in data-driven MOR, and in particular by a data-driven moment matching framework developed at Imperial College London. While the theoretical foundations of this approach have been established and supported by dedicated numerical tools for moment estimation, its applicability to large scale industrial models has not yet been thoroughly explored. This observation provided the initial motivation for the present work. The thesis aims to bridge the gap between theoretical developments in data-driven MOR and their practical application to complex, physics-based models commonly used in industrial contexts. In particular, the focus is on high-order models derived from detailed thermo-dynamics, fluid-dynamics and electrochemical descriptions, whose dimensionality often makes them unsuitable for simulation and control-oriented tasks. Within this framework, the thesis investigates the feasibility and effectiveness of data-driven moment matching when applied to systems of significantly higher order than those typically considered in the literature. The work is based on two industrially relevant case studies developed within the Politecnico di Milano modeling framework: a heat exchanger-turbine system and a Solid Oxide Fuel Cell system. These models provide a realistic testbed to assess the scalability of the method, its performance in both SISO and MIMO configurations, and its potential role as an enabling tool for advanced control strategies through order reduction. The remainder of the thesis is organized to first introduce the theoretical background and the data-driven moment matching framework and subsequently present the validation methodology and the numerical result obtained on the selected case studies. The final

chapters discuss the main findings, limitations, and possible directions for future research.

1 Introduction

1.1 Motivation and Context

In recent years, the increasing complexity of modern industrial systems has led to the development of highly detailed physics-based models, capable of accurately describing heterogeneous phenomena such as thermo-fluid dynamics, energy conversion processes, and electrochemical reactions. In many cases, these models are derived from first principles and implemented within advanced modeling environments, such as Modelica, which allow for a modular and highly parametrized representation of physical systems [1]. The level of detail achieved by such models, however, comes at the cost of significant increase in their dimensionality. In industrial applications, it is now common to deal with dynamic models characterized by several tens or even hundreds or thousands of state variables, particularly in the energy and power conversion sectors. While these high-fidelity models are essential during the design and offline analysis phases, their numerical complexity severely limits their usability in contexts requiring fast or repeated simulations. A particularly critical issue arises in control-oriented applications. Advanced model-based control strategies, such as MPC, explicitly rely on the system dynamics to solve an optimization problem online at each control step. When high-order models are employed, the resulting computational burden often exceeds the available computational resources or violates the real-time constraints imposed by the system dynamics, making the direct use of full-order models infeasible in practice [2], [3]. Within this context, MOR plays a crucial role in enabling the use of complex physical models for simulation, optimization, and control purposes. By reducing the dimensionality of the system while preserving its dominant dynamic behavior, MOR techniques provide a fundamental tool to bridge the gap between high-fidelity modeling and real-time applicability.

1.2 Model Order Reduction

Model order reduction has been extensively studied in the control and dynamic systems literature. Classical MOR techniques are based on an explicit mathematical description of the system, typically given in state-space or transfer function form, and exploit structural properties such as controllability, observability, and time scale separation. Among the most established methods, balanced truncation provides reduced-order models with guaranteed error bounds in the norm and preserves important system-theoretic properties [4], [5]. Projection-based techniques relying on

Krilov subspaces constitute another widely adopted class of methods and are closely related to moment matching approaches, which aim at reproducing a finite number of moments of the original system transfer function around selected expansion points [6], [7]. Despite their effectiveness, classical model-based MOR techniques fundamentally rely on full access to system equations, and work only on linear or linearized models. In many modern industrial scenarios, this assumption is no longer valid. The system may only be available through a complex numerical simulator, may be subject to intellectual property constraints, or may be accessible exclusively through measured or simulated data without requiring explicit knowledge of the system dynamics [8], [9]. While data-driven methods offer increased flexibility, ensuring that the reduced models faithfully capture the relevant dynamic characteristics of the original systems remains a central challenge.

1.3 Moment Matching and Data-Driven Extensions

Moment matching is a well-established MOR technique that aims at approximating the input-output behavior of a dynamic system by enforcing the exact matching of a finite number of moments of its transfer function. These moments are defined as the coefficients of the Taylor expansion of the transfer function around selected expansion points in the complex plane and encode local frequency-domain information of the system dynamics. The moment matching framework has been rigorously formalized for both linear and nonlinear systems in [10], where model reduction is achieved by enforcing suitable interpolation conditions through projection-based techniques. In the linear case, moment matching can be interpreted as a projection of the original system onto Krilov subspaces, leading to reduced-order models that reproduce the moments of the full-order system at the chosen interpolation points. For nonlinear systems, the concept is extended through steady-state responses to appropriately defined signal generators, providing a unifying theoretical framework via interpolation. The key strength of moment matching lie in its flexibility, in particular, by appropriately selecting the expansion points and the number of matched moments, the reduced-order model can be tailored to accurately reproduce the system's behavior in frequency regions of interest. This property makes moment matching particularly attractive for control-oriented applications, where the preservation of dominant dynamics in specific operating regimes is often more relevant than a global approximation of the system's behavior. Despite its solid theoretical foundation, classical moment matching methods rely on explicit knowledge of the system equations, including state-space matrices or nonlinear vector fields. As discussed in the previous section, this requirement can be overly restrictive in practical

applications, especially when dealing with large-scale industrial systems available only through simulation tools or experimental data. Motivated by these limitations, recent research efforts have focused on extending the moment matching paradigm to data-driven settings. In this context, the moments of the system are no longer computed from an explicit analytical model but are instead referred directly from measured or simulated input-output data. A significant advancement in this direction has been achieved first by the work treated in [22], and after using kernel-based learning techniques, which allow nonlinear systems to be embedded into Reproducing Kernel Hilbert Spaces (RKHS), where linear identification and interpolation tools can be effectively applied. A data-driven formulation of nonlinear moment matching in RKHS has been proposed in [11], where the nonlinear dynamics are implicitly represented through kernel evaluations, enabling the estimation of system moments directly from data. This approach has been further extended in [12], where kernel-based learning is combined with moment matching principles to construct reduced-order models without requiring explicit knowledge of the system equations. These kernel-based data-driven methods preserve the interpolation-based philosophy of classical moment matching while significantly relaxing the assumption of model availability and structural knowledge. While these approaches have demonstrated promising results on benchmark nonlinear systems, their application has so far been limited to systems of relatively low or moderate order. In particular, the scalability of data-driven moment matching techniques to large-scale industrial models and their extensions to MIMO configurations remain open to research challenges. Addressing these issues constitutes one of the central motivations of the present thesis.

1.4 Open Challenges and Research Gap

Despite the significant theoretical advances in moment matching and its data-driven extensions, several challenges remain when these techniques are confronted with the requirements of a large-scale modeling framework. In particular, the application of data-driven moment matching complex physics-based models developed for energy and power generation systems introduces specific demands that go beyond the scenarios typically considered in the literature. High-fidelity industrial models, such as heat exchanger-turbine system, and SOFC system are characterized by many states variables, strong dynamic couplings among subsystems, and highly nonlinear behaviors arising from thermo-fluid and electrochemical phenomena. For such models, the dimensionality and multi-variable nature of the dynamics are not secondary aspects, but rather intrinsic features that must be explicitly addressed by

any model reduction methodology intended for practical use. In this context, MIMO configurations play a central role. Industrial energy systems inherently involve multiple actuators, disturbances, and measured outputs, and reduced-order models that neglect this structure are of limited practical relevance. While classical moment matching theory provides a theoretical foundation for MIMO systems when full model information is available [5], [10], the extension of data-driven moment matching techniques to MIMO industrial models of large dimension remains a critical aspect to be investigated, particularly in terms of accuracy, robustness, and numerical reliability. Another core challenge arises from the operating conditions under which such industrial systems are required to operate. Models developed within Politecnico di Milano for heat-exchangers-turbine and SOFC systems are typically designed to operate over a wide range of operating points, where the system dynamics may vary significantly. Capturing this variability within a reduced-order model is essential for both simulation and control purposes. Consequently, the ability of data-driven moment matching techniques to accommodate operating-point-dependent dynamics represents a key aspect for their applicability to these classes of systems. From a control-oriented perspective, these challenges are further amplified by the need to deploy reduced-order models within advanced control strategies, such as MPC. For large-scale systems, the use of full-order description is often computationally incompatible with real-time requirements [2], [3]. Therefore, assessing whether data-driven moment matching can produce reduced models that are sufficiently accurate while enabling real-time control implementation constitutes a pivotal motivation for this work. In light of these considerations, the research gap addressed in this thesis is not merely the lack of prior validation on large-scale or MIMO systems, but rather the need to assess the suitability of data-driven moment matching techniques for complex industrial models where high dimensionality, multi-variable interactions, and control-oriented requirements are intrinsic characteristics. The present thesis aims to contribute to this assessment by systematically evaluating the method on heat exchanger-turbine and SOFC models developed within an modeling framework by Politecnico di Milano.

1.5 Objectives and Contributions

The main objective of this thesis is to investigate the applicability and effectiveness of data-driven moment matching techniques for the reduction of large-scale industrial models, with particular emphasis on systems characterized by high dimensionality and MIMO dynamics. Rather than introducing a new theoretical framework, the focus of this work is on the systematic validation, critical assessment, and practical

extensions of existing data-driven moment matching methodology when applied to complex physics-based models. The objectives of this thesis can be summarized as follows:

- To assess the effectiveness of data-driven moment matching techniques on large-scale industrial models, evaluating their ability to produce accurate reduced-order representations for systems of order 80 for heat-exchanger-turbine model and 268 for SOFC model;
- To validate the method in SISO configurations, providing a clear benchmark against established theoretical expectations and enabling a controlled analysis of accuracy and stability;
- To investigate the extension of the method to MIMO systems, which are intrinsic to industrial energy applications and constitute a core requirement for practical relevance;
- To analyze the reduced-order models using control-oriented metrics, including frequency-domain characteristics and singular value analysis, with particular attention to the preservation of dominant input-output couplings.

Beyond these objectives, the thesis provides several original contributions to the existing literature. First, it presents a systematic application of data-driven moment matching techniques to models of high order, thereby extending the empirical validation of the method to a previously unexplored regime. Second, it introduces a practical alternative strategy to the MIMO case that is shown to be effective for complex energy systems operating around multiple operating points. Finally, the work establishes a structured validation framework that bridges the gap between data-driven model reduction theory and the requirements of large-scale modeling and control applications.

1.6 Thesis Structure

The thesis is organized as follows: Chapter 2 presents the state of the art in MOR, with particular emphasis on moment matching techniques and data-driven approaches. Chapter 3 introduces the theoretical background required for moment matching, including the definition of moment and its role in model reduction. Chapter 4 describes the models used in this thesis. Chapter 5 defines the reduction of the heat exchanger-turbine model and validates the results. Chapter 6 defines the reduction of the SOFC model and validates the results. Finally, Chapter 7 summarizes the main conclusions of the thesis and outlines possible directions for future research.

2 State of the Art

2.1 Overview of Model Order Reduction

MOR is a fundamental tool in the analysis and control of dynamic systems characterized by high-dimensional state-space representations. In many engineering applications, detailed physics-based modeling leads to systems with a large number of states, arising from spatial discretization of partial differential equations, multi-physics coupling, or detailed component descriptions [5]. The main objective of MOR is to derive a reduced-order model that approximates the input-output behavior of the original high-order system while significantly lowering its computational complexity. Ideally, the reduced order model should preserve the dominant dynamical characteristic of the original system, including stability, transient response, and frequency-domain properties, while maintaining a structure suitable for control-oriented analysis [13]. Over the past decades, a wide range of MOR techniques have been developed, particularly for linear time-invariant systems. Among the most established approaches are projection-based methods, such as balanced truncation and Krylov subspace techniques, which rely on full knowledge of the system matrices and exploit specific structural properties of the state-space representation [5], [14]. In particular, balanced truncation is based on the computation of controllability and observability Gramians, which are obtained as solutions of continuous or discrete-time Lyapunov equations. Despite their strong theoretical guarantees, classical MOR methods face significant limitations when applied to large-scale systems. The solution of large-scale Lyapunov equations and the computation of matrix factorizations or singular value decompositions can impose a substantial computational burden as the system dimension increases, especially when dealing with dense or poorly structured system matrices [5], [15]. As a result, the practical applicability of such techniques may be limited in settings characterized by very high-order models and tight computational constraints. These limitations have motivated growing interest in data-driven approaches to MOR, which aim to construct reduced-order models directly from input-output data without requiring explicit access to the underlying system equations [16]. Such methods are particularly appealing in industrial contexts, where detailed analytical models may be unavailable, while simulation or experimental data are readily accessible. The following sections review both model-based and data-driven MOR techniques, with particular emphasis on moment matching approaches and their relevance to large-scale systems.

2.2 Model-Based Model Order Reduction

Model-based model order reduction methods assume full access to the state-space representation of the system to be reduced. Given a high-order linear time-invariant system described by its system matrices, these techniques exploit algebraic and structural properties of the model to construct a reduced-order representation through suitable projections. As a result, they provide strong theoretical guarantees in terms of stability preservation and approximation accuracy, but their applicability as inherently tied to the availability and tractability of the underlying system equations [5]. The main model-based MOR techniques relies on projection-based approaches. In this framework, the original high-dimensional state vector is projected onto a low-dimensional subspace spanned by a reduced set of basis vectors. The reduced-order model is then obtained by enforcing a Petrov-Galerkin condition on the system equations, resulting in a lower-dimensional dynamic system that approximates the input-output behavior of the original model [5], [13]. The quality of the reduced model strongly depends on the choice of the projection subspaces, which are typically constructed to capture the dominant dynamic features of the system. Among projection-based methods, balanced truncation represents one of the most widely used and theoretically grounded approaches. Balanced truncation is based on a coordinate transformation that makes the controllability and observability Gramians equal and diagonal, thereby providing a measure of the joint controllability and observability of each state [16]. States associated with small Hankel singular values are considered dynamically negligible and can be truncated with a priori bound on the approximation error. Despite these appealing properties, balanced truncation requires the solution of large-scale Lyapunov equations and the computation of matrix factorizations, which may limit its practical applicability to high-order systems [5], [15]. Moreover, these techniques are limited to linear or linearized systems. An alternative and computationally efficient class of model-based MOR techniques is given by Krylov subspace methods. These approaches construct projection subspaces by matching moments of the system transfer function around selected expansion points in the complex plane. Krylov-based methods are particularly attractive for large-scale problems, as they avoid the explicit computation of Gramians and rely instead on sparse linear algebra operations [14]. However, the choice of expansion points and the lack of global error bounds may affect the robustness of the reduced models, especially when wide frequency ranges or multiple operating conditions are of interest. Moment matching provides a unifying interpretation of several projection-based MOR techniques and establishes a direct connection between time-domain and

frequency domain approximations. In the model-based setting, moment matching assumes full knowledge of the system matrices and allows the explicit computation of system moment through analytical expressions [10]. This framework has been extended to nonlinear systems and has played a key role in bridging classical MOR techniques with modern data-driven approaches, which aim to estimate moments directly from input-output data without requiring explicit access to the system equations.

2.3 Data-Driven System Identification

Data-driven system identification aims to construct mathematical models of dynamical systems directly from measured input-output data, without relying on a full first-principles description of the underlying dynamics. Depending on the amount of prior structural knowledge incorporated into the modeling process, system identification approaches are commonly classified as black-box or grey-box methods [17]. Black-box identification techniques rely solely on input-output data and make minimal assumptions on the internal structure of the system. These methods are particularly appealing when the system dynamics are highly complex or poorly understood, but they often require large amounts of data and may result in models with limited physical interpretability. In contrast, grey-box approaches exploit partial knowledge of the system structure, such as known physical relations or parametric models forms, while estimating unknown parameters and physic-based approaches, offering improved interpretability and potentially enhanced predictive performance [17], [18]. In the context of model order reduction, a common data-driven strategy consists of first identifying a high-order model, either black-box or grey-box, from data and subsequently applying classical model-based MOR techniques. In particular, the identification of high-dimension state-space models can be numerically challenging and sensitive to noise, and modeling errors introduced during the identification phase may be amplified during subsequent reduction step [17]. These issues become especially pronounced for large-scale systems, where the number of states required to accurately capture the system dynamics can be prohibitively large. Therefore, identification-based approaches alone may fail to deliver reliable reduced-order models for complex systems. This limitation has motivated the development of a direct data-driver MOR framework that aims to construct reduced-order models from data without explicitly identifying a full-order intermediate model.

2.4 Data-Driven Model Order Reduction

Data-driven MOR have attached increasing attention as an alternative to classical model-based MOR techniques, particularly in scenarios where the governing equations are unavailable, uncertain, or too complex to be exploited directly [5], [8]. Among data-driven MOR techniques, the Loewner framework represents one of the most established interpolation-based approaches. Originally introduced for linear time-invariant systems, the Loewner framework enables the construction of reduced-order realizations that interpolate frequency-domain or transfer function data at selected points, without requiring the identification of a high-order model [9], [19]. The framework has been successfully extended to parametrized systems, allowing reduced-order models to capture variations with respect to operating conditions or physical parameters directly from data [9]. Another important class of data-driven reduction techniques is rooted subspace identification methods. These approaches exploit the geometric structure of measured input-output data to extract low-dimensional state representation that captures the dominant system dynamics [20]. When used with the objective of obtaining compact models rather than accurate full-order realizations, subspace-based techniques can be interpreted as data-driven MOR methods and have been widely adopted in practice for linear systems [13], [20]. Despite their effectiveness, both Loewner-based and subspace-based data-driven MOR approaches exhibit limitations when applied to large-scale systems. In particular, numerical robustness and scalability issues may arise as the system dimension increases, especially in the presence of noise or limited excitation. Furthermore, these methods generally do not provide explicit guarantees on stability preservation or on the accurate reproduction of local dynamical behavior around specific operating points, which is often critical in control-oriented data-driven approaches, as discussed in the following section.

2.5 From Model-Based to Data-Driven Moment Matching

Moment matching is a fundamental paradigm in model order reduction, aiming at constructing reduced-order models that locally replicate the input-output behavior of the original system around selected expansion points. In its classical formulation, moment matching relies on the explicit availability of a state-space representation and is closely connected to Krylov subspace projection methods, where the reduced order model interpolates the transfer function and its derivatives at prescribed frequencies

[5], [6]. A unified and rigorous treatment of model-based moment matching for both linear and nonlinear systems was introduced in [10], where the concept of moments is interpreted through an operator-theoretic framework. This formulation clarified the structural properties of moment matching, its relation to system invariants, and its applicability to nonlinear dynamics, thereby providing a solid theoretical foundation for subsequent extensions beyond time-invariant systems. The theory of nonlinear MOR by model-based moment matching was further developed and systematized in the monograph [21]. This work provides a comprehensive analysis of the geometric and dynamical aspects of moment matching, establishing conditions for the existence and uniqueness of reduced-order models, and highlighting the role of invariant manifolds in the reducing process. By framing moment matching within a nonlinear systems perspective, this contribution represents a cornerstone for extending the methodology to complex dynamical systems while preserving local approximation properties. Building on these theoretical foundations, an explicit formulation of data-driven moment matching for linear and nonlinear systems was proposed in [22]. In this work, moment information is inferred directly from measured input-output data, thus relaxing the requirement of full model knowledge. This approach represents a critical conceptual shift, as it enables the construction of reduced-order models using experimentally accessible information while retaining the theoretical structure of classical moment matching. As such, it establishes a natural bridge between model-based reduction techniques and data-driven methodologies. More recently, data-driven moment matching has been further advanced through the integration of machine learning techniques. In particular, kernel-based learning methods have been proposed to estimate moment information in nonlinear systems using RKHS [11], [12]. By exploiting the expressive power of kernels, these approaches enable approximation of nonlinear moment operators directly from data, leading to fully data-driven reduced-order models. Importantly, these methods preserve the interpolation and local approximation properties inherent to moment matching while significantly enhancing flexibility and applicability. Compared to other data-driven MOR techniques, such as Loewner-based or subspace-based methods, data-driven moment matching exhibits several distinctive advantages. It provides reduced-order models that preserve local dynamic behavior around selected operating points, offers a clear interpretation in terms of system dynamics, and naturally accommodates nonlinear systems. Moreover, the close connection with classical projection-based methods facilitates its integration into control-oriented workflows, where local accuracy and dynamical consistency are of paramount importance [5], [10], [21]. Despite these strengths, several open challenges remain. In particular, the practical applicability of

data-driven moment matching large-scale systems has not yet been thoroughly investigated. Existing studies primarily focus on low or medium order systems and often consider SISO configurations. The extension to MIMO systems, especially in large-scale settings, remains an open research problem. Addressing these challenges is essential to assess the viability of data-driven moment matching in realistic industrial applications and motivates the contributions of this thesis.

3 Moment Matching for Model Reduction

3.1 General Problem Setting and Mathematical Framework

The development of high-fidelity models for industrial processes has been significantly facilitated by advances in physics-based modeling environments and object-oriented modeling languages. Such tools enable the systematic derivation of detailed dynamical models that capture the underlying physical phenomena with high accuracy. As a result, complex thermo-fluid, electrochemical, and energy systems are often described by nonlinear dynamical models with state dimensions ranging from several tens to several hundreds or even thousands of variables. While the availability of detailed models represents a major asset from a modeling perspective, it also introduces substantial challenges when such models are employed for tasks beyond offline simulation. In particular, the computational burden associated with high-dimensional state-space representations severely limits their applicability in real-time control, optimization, and estimation problems. This issue is especially pronounced in control-oriented applications, where strict timing constraints must be satisfied to ensure closed-loop stability and performance. From a systems-theoretical viewpoint, the models considered in this work can be described by nonlinear state-space equations of the form

$$\begin{aligned} \dot{x}(t) &= f\big(x(t),\, u(t)\big), \\ y(t) &= h\big(x(t)\big). \end{aligned} \tag{3.1}$$

where $x(t) \in \mathbb{R}^n$ denotes the system state, $u(t) \in \mathbb{R}^m$ the control input, and $y(t) \in \mathbb{R}^p$ the measured output. The dimension n is assumed to be large, reflecting the complexity of the physical system under consideration. The function f and h are assumed to be sufficiently smooth to guarantee the existence of unique solutions and to allow for local linearization and differential analysis. Moreover, in this thesis, only

strictly proper models are considered, but the reduction procedure could also be applied in the case of non-strictly proper systems. In many practical scenarios, the above nonlinear model is not used directly in its original form. Instead, linearized representations around specific operating points are often employed to analyze local behavior or to design control strategies. Given an equilibrium point (x^*, u^*), the corresponding linearized model can be written as

$$\dot{\xi} = A\xi(t) + Bv(t),$$
$$\eta(t) = C\xi(t).$$

where $\xi = x - x^*$, $v = u - u^*$, $\eta = y - y^*$, and matrices A, B, and C are obtained from the Jacobians of f and h. Even in this linearized setting, however, the system dimension often remains prohibitively large, motivating the use of systematic model order reduction techniques. MOR aims at constructing a reduced-order model that approximates the input-output behavior of the original system while significantly decreasing its state dimension. More precisely, the objective is to determine a dynamical system of the form

$$\dot{x_r}(t) = f_r\big(x_r(t), u(t)\big), \quad (3.2)$$
$$y_r(t) = h_r\big(x_r(t)\big).$$

with $x_r(t) \in \mathbb{R}^v$, $v \ll n$, such that the reduced output $y_r(t)$ provides an accurate approximation of $y(t)$ for relevant class of input signals. Importantly, the reduced model is not requiring reproducing the full internal dynamics of the original system, but rather to preserve the dominant input-output characteristics that are most relevant for analysis and control purposes. Within this general framework, different notions of approximation accuracy can be considered. Classical approaches often aim at minimizing global error measures, such as H_2 or H_∞ norms, over a broad frequency range. In contrast, moment matching techniques focus on reproducing the local behavior of the system transfer function around selected expansion points. This local approximation perspective is particularly appealing in control-oriented contexts, where the system is typically operated in a neighborhood of specific working conditions. A distinctive feature of the moment matching approach is its interpretation in terms of system moments, which characterize the coefficients of the Taylor expansion of the transfer function with respect to the complex frequency variable. By enforcing the equality of a prescribed number of moments between the original and reduced models, one can ensure that the reduced system exhibits

identical local input-output behavior up to a a given order. This concept extends naturally to nonlinear systems, where moment matching can be interpreted in terms of steady-state responses to specific classes of input signals. In recent years, increasing attention has been devoted to data-driven formulations of MOR.

In such settings, the explicit state-space representation of the system is not required for the construction of the reduced model. Instead, the reduction procedure relies solely on input-output measurements collected from the system under suitable excitation conditions. This paradigm is particularly relevant when dealing with complex systems, for which the governing equations may be partially unknow, difficult to access, or computationally too expensive to manipulate directly.

In the context of this thesis, the full-order models are assumed to be available (and indeed they are) for data acquisition, for validation and benchmarking purposes, but they are not exploited directly by the reduction algorithm. The reduced-order models are instead constructed using data-driven moment matching techniques, which estimate the relevant moment information directly from measured input-output data gathered from simulations, without measurement noise, on full-order models. This distinction is crucial as it allows the reduction methodology to be applied in scenarios where only experimental data are available, while still enabling a rigorous assessment on the reduction quality through comparison with the full-order models. Finally, it is worth emphasizing that the reduced-order models considered in this work are constructed with a clear control-oriented objective. In particular, the reduction process is designed to enable the use of advance control strategies, such as MPC, which are otherwise infeasible when employing high-order models due to excessive computational requirements. By significantly reducing the system dimension, and the computational burden of computing the RHS of the system equations, while preserving the dominant dynamical features, moment-matched reduced-order models provide an effective bridge between high-fidelity modeling and real-time control implementation.

The theoretical framework introduced in this section establishes the foundations for the moment matching techniques discussed in the following sections. Section 3.2 focuses on the mathematical formulation of moment matching for linear and nonlinear systems, while Section 3.3 extends these concepts to the data-driven setting.

3.2 Moment Matching for Linear and Nonlinear Systems

Now a mathematical formulation and formalization of model-based moment matching theory will be introduced both for linear and nonlinear systems, to permit a

clear comprehension of moment and reduce-order model, and more precisely the connection between these two concepts.

3.2.1 Moment Matching for Linear Systems

3.2.1.1 Intuitive Interpretation of Moment Matching

The notion of moment matching originates from approximation theory and can be intuitively understood by considering the local behavior of a system around specific operating conditions or frequencies. For a linear time-invariant (LTI) system the input-output behavior is described in the frequency domain by the transfer function

$$G(s) = C(sI - A)^{-1}B.$$

For a complex expansion point $s_0 \notin \sigma(A)$ where $\sigma(A)$ denotes the spectrum (i.e., the set of eigenvalues) of the matrix, the transfer function admits a Taylor expansion

$$G(s) = \sum_{k=0}^{\infty} M_k (s - s_0)^k,$$

where the coefficients

$$M_k = C(s_0 I - A)^{-(k+1)}B,$$

are known as the moment of the system at s_0.
Equivalently, the moments of order k can be defined as the k-th derivative of the transfer function evaluated at s_0

$$M_k = \frac{(-1)^k}{k} \frac{d^k}{ds^k} G(s)_{s=s_0}.$$

A reduced-order model matches the moments of the full-order system if it reproduces the same coefficients M_k up to some desired order. While this frequency-domain interpretation is insightful, it becomes inadequate when extending the concept to nonlinear systems. For this reason, moment matching theory adopts a time-domain formulation based on the notion of signal generator.

3.2.1.2 Linear Signal Generator: Structure and Conditions

Consider the linear time-invariant system

$$\dot{x}(t) = Ax(t) + Bu(t),$$
$$y(t) = Cx(t).$$

where:

- $x(t) \in \mathbb{R}^n$ is the input state vector;
- $u(t) \in \mathbb{R}^m$ is the input;
- $y(t) \in \mathbb{R}^p$ is the output;
- $A \in \mathbb{R}^{n \times n}$, $B \in \mathbb{R}^{n \times m}$, $C \in \mathbb{R}^{p \times n}$.

In the time-domain formulation of moment matching, the input is generated by an autonomous linear signal generator

$$\dot{w}(t) = Sw(t),$$
$$u(t) = Lw(t).$$

where:

- $w(t) \in \mathbb{R}^v$ is the generator state,
- $S \in \mathbb{R}^{v \times v}$ is the generator dynamics matrix,
- $L \in \mathbb{R}^{m \times v}$ maps the generator state to the system input.

The variable $w(t)$ does not generally have a physical meaning; rather, it is an auxiliary mathematical variable introduced to generate structured excitation signals. The trajectories of $w(t)$ fully determine the temporal and spectral content of input $u(t)$, and therefore, which dynamic modes of the system are excited and approximated. The matrix S encodes the spectral information of the excitation signals and is chosen in block Jordan canonical form

$$S = diag(J_1, \ldots, J_q),$$

where each Jordan block $J_i \in \mathbb{C}^{r_i \times r_i}$ has the structure

$$J_i = \begin{bmatrix} s_i & 1 & 0 & \cdots & 0 \\ 0 & s_i & 1 & \cdots & 0 \\ \vdots & & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & s_i & 1 \\ 0 & \cdots & \cdots & 0 & s_i \end{bmatrix}.$$

Each eigenvalue $s_i \in \mathbb{C}$ represents an interpolation point, while the dimension r_i of the block determines the number of consecutive moments matched at the interpolation point s_i. The matrix L is structured consistently with S and is chosen such that the pair (S, L) is controllable. In practice, for each Jordan block J_i , the corresponding columns of L excite the first state of the block, ensuring that all generator modes influence the input signal. The signal generator is said to be admissible if the following conditions are satisfied [10]:

1. Spectral separation

$$\sigma(A) \cap \sigma(S) = 0,$$

Which guarantees the absence of resonance between the plant and the generator.

2. Controllability

$$rank\, C(S, L) = v,$$

where $C(S, L)$ denotes the controllability matrix.

3. Marginal or asymptotic stability of S ensuring bounded generator trajectories.

3.2.1.3 Linear Sylvester Equation: Definition and Properties

When the plant is interconnected with the signal generator the augmented system is

$$\begin{aligned} \dot{x}(t) &= Ax(t) + BLw(t), \\ \dot{w}(t) &= Sw(t), \\ y(t) &= Cx(t). \end{aligned}$$

A key result in moment matching theory [10] states that the augmented system admits a linear invariant manifold

$$M := \{(x, w) \in \mathbb{R}^n \times \mathbb{R}^v : x = \Pi w\}, \quad (3.3)$$

if and only if the matrix $\Pi \in \mathbb{R}^{n\times v}$ satisfied the Sylvester equation

$$A\Pi + BL = \Pi S\,. \quad (3.4)$$

The Sylvester equation is a linear matrix equation whose solution Π defines a linear mapping between the generator states and the plant states. The manifold M is invariant if trajectories initialized on it remain on it for all future times. Invariance means that the vector field of the augmented system is tangent to M .Indeed, differentiating $x = \Pi w$ yields

$$\dot{x}(t) = \Pi\dot{w}(t) = \Pi S w(t),$$

on the other hand, the plant dynamics restricted to M satisfy

$$\dot{x}(t) = A\Pi w(t) + BLw(t).$$

Equating the two expressions for all w leads directly to the Sylvester equation (3.1). Thus, the Sylvester equation is the necessary and sufficient condition for the existence of an invariant linear manifold associated with the signal generator. Equation (3.1) admits a unique solution if and only if the spectral separation condition holds

$$\sigma(A) \cap \sigma(S) = 0,$$

which guarantees that the associated linear operator is invertible. The matrix Π is a linear mapping from the generator state space to the system state space and can be interpreted as an immersion operator. When S is in the Jordan form, the columns of Π span Krylov subspaces related to the interpolation points encoded in S. Consequently, the invariant manifold embeds the system's moments, and moment matching is achieved in the time domain.

3.2.1.4 Reduced-Order Model Construction

Once the signal generator (S, L) has been fixed and the Sylvester equation (3.4) has been solved, the matrix $\Pi \in \mathbb{R}^{n\times v}$ defines the invariant manifold (3.3) that embeds the steady-state behavior of the original system under the action of the generator. The construction of a reduced-order model is based on the idea of reproducing the same manifold-induced input-output behavior using a system of dimension v , whose state

directly parameterizes the generator dynamics. A reduced-order model of dimension v is defined as

$$\dot{x_r}(t) = Fx_r(t) + Gu(t),$$
$$y_r(t) = Hx_r(t).$$

Where:

- $x_r(t) \in \mathbb{R}^v$ is the reduced-order state, with $v \ll n$;
- $F \in \mathbb{R}^{v \times v}$ is the redeced-state matrix;
- $G \in \mathbb{R}^{v \times m}$ is the reduced input matrix;
- $H \in \mathbb{R}^{p \times v}$ is the reduced output matrix;

The reduced state x_r can be interpreted as a coordinate of the generator state, or, equivalently, as a parametrization of the invariant manifold. For the reduced model to match the moments associated with the signal generator (S, L), it must admit an invariant manifold analogous to that of the full-order system. This requirement leads to the reduced Sylvester equation

$$F\Pi_r + GL = \Pi_r S \text{ , (3.5)}$$

where $\Pi_r \in \mathbb{R}^{v \times v}$ defines the invariant manifold of the reduced system,

$$x_r = \Pi_r w.$$

The reduced system matches the moments of the original system if, in addition, the output consistency condition

$$C\Pi = H\Pi_r \text{ , (3.6)}$$

is satisfied. This condition ensures that the outputs of the full-order and reduce-order systems coincide along the trajectories induced by the generator. A key observation is that the reduced-order model is not unique. In particular, the matrices F and H are not uniquely determined by the moment matching conditions, but must satisfy the algebraic constrains imposed by equations (3.5) and (3.6).

A common and conceptually simple choice is to set

$$\Pi_r = I_v \text{ ,}$$

which implies that the reduced state directly coincides with the generator state, i.e., $x_r = w$. Under this choice, the reduced Sylvester equation (3.5) becomes

$$F + GL = S\,, \quad (3.7)$$

this equation explicitly shows how the reduced dynamics matrix F is related to the generator dynamics S and the input matrix G . Solving (3.7) for F yields

$$F = S - GL\,.$$

Similarly, substituting $\Pi_r = I_v$ into output consistency condition (3.6) gives

$$H = C\Pi\,.$$

From this construction, the role of the reduced-order matrices becomes clear:

- The matrix F defines the internal dynamics of the reduced model and is obtained by modifying the generator dynamics S through the feedback term GL ;
- The matrix G is a free design parameter and represent a degree of freedom that can be exploited to shape the reduced dynamics, for example to enforce stability or to improve transient behavior;
- The matrix H maps the reduced state to the output and is uniquely determined by invariant manifold of the original system through the relation $H = C\Pi$.

This construction ensures that, when driven by the same generator, the reduced-order model reproduces exactly the steady-state output of the full-order model. Conceptually, the reduced-order model is therefore constructed by:

1. Fixing a signal generator(S, L).
2. Solving the Sylvester equation of the full-order model to obtain Π .
3. Selecting a reduced invariant manifold (typically $\Pi_r = I_r$).
4. Defining F and H through algebraic relations derived from the Sylvester equations.
5. Exploiting the freedom in G to impose additional properties.

This perspective highlights that moment matching is not merely an approximation technique, but a geometric construction based on invariant manifolds, where reduced models are obtained as low-dimensional realizations of the same manifold-induced behavior [10].

3.2.2 Moment Matching for Nonlinear Systems

3.2.2.1 Motivation and Conceptual Interpretation

The extension of moment matching from linear to nonlinear systems represents a fundamental conceptual shift; moments are intrinsically linked to the transfer function and admit a clear interpretation either as coefficients of a Taylor expansion around selected interpolation points or as derivatives of the transfer function evaluated at those points. This interpretation, however, relies critically on linearity and superposition. For nonlinear systems, the absence of a global transfer function requires redefining the notion of a moment in a way that is independent of frequency-domain representation. The key idea introduced in [10] and further development in [21] is to define moments relative to a class of input signals generated by an auxiliary dynamical system, referred to as a signal generator. From this perspective, a moment is no longer a scalar coefficient, but a function describing the steady-state input-output behavior of the nonlinear system when driven by a specific generator. This viewpoint naturally leads to a trajectory-based and geometric formulation, where invariant manifolds replace algebraic interpolation conditions.

3.2.2.2 Nonlinear System Description

Consider a nonlinear input-output system described by (3.1); the system is assumed to admit an equilibrium point (x^*, u^*). Through a change of variables, the equilibrium can be shifted to the origin, which simplifies the exposition without loss of generality. In this discussion, the starting system (3.1) is considered strictly proper for simplicity; the discussion can be generalized to a non-strictly proper system by thinking of the output equation of the reduced model as also dependent on the input.

3.2.2.3 Nonlinear Signal Generator

Moments are defined with respect to a nonlinear signal generator of the form

$$\dot{w}(t) = s\big(w(t)\big), \quad (3.8)$$
$$u(t) = l\big(w(t)\big).$$

Where:

- $w(t) \in \mathbb{R}^v$ is the generator state;
- $s : \mathbb{R}^v \to \mathbb{R}^v$ defines the internal generator dynamics;
- $l : \mathbb{R}^v \to \mathbb{R}^m$ maps the generator state into admissible system inputs.

The generator plays a role analogous to the interpolation points in the linear case: it defines which aspects of the system behavior are to be preserved in the reduced

model. To ensure a meaningful definition of moments, the generator must satisfy three conditions:

1. $s(0) = 0$, i.e., the origin is an equilibrium.
2. The equilibrium is locally asymptotically stable.
3. $s(\cdot)$ and $l(\cdot)$ are sufficiently smooth.

These conditions guarantee the existence of steady-state trajectories and allow the construction of invariant manifolds. In analogy with the linear case, where interpolation points correspond to eigenvalues of matrix S, the nonlinear generator encodes generalized interpolation conditions through the structure of $s(w)$. A typical choice is

$$s(\omega) = \begin{bmatrix} s_1(\omega_1) \\ \vdots \\ s_\nu(\omega_\nu) \end{bmatrix}, \qquad l(\omega) = \begin{bmatrix} l_1(\omega) \\ \vdots \\ l_m(\omega) \end{bmatrix},$$

where each component $s_i(\cdot)$ generates a specific nonlinear excitation mode. In particular:

- Linear generators recover classical interpolation at fixed frequency;
- Nonlinear generators allow for amplitude-dependent, nonlinear, or transient-rich excitations.

Thus, the choice of $s(\cdot)$ determines the class of trajectories along which moment matching is enforced generalizing the notion of interpolation points to interpolation manifolds.

3.2.2.4 Invariant Manifold and Nonlinear Sylvester Equation

The core mathematical object underlying nonlinear moment matching is a smooth mapping

$$\pi : \mathbb{R}^\nu \to \mathbb{R}^n,$$

which defines an invariant manifold

$$M = \{(x, w) \in \mathbb{R}^n \times \mathbb{R}^\nu : \pi(w)\}, \quad (3.9)$$

for the interconnected system

$$\dot{x}(t) = f\left(x(t),\, l\big(w(t)\big)\right), \quad (3.10)$$
$$\dot{w}(t) = s\big(w(t)\big),$$
$$y(t) = h\big(x(t)\big).$$

This interconnection plays a role analogous to the frequency-domain excitation used in linear moment matching: it defines which modes, nonlinear responses, and steady-state behaviors are selected. The manifold (3.9) is invariant if and only if satisfies the nonlinear Sylvester equation

$$\frac{\partial \pi}{\partial w}(w)s(w) = f\big(\pi(w),\, l(w)\big), \quad (3.11)$$

for all w in a neighborhood of the origin. This equation generalizes the classical Sylvester equation:

- In the linear case, it is an algebraic matrix equation;
- In the nonlinear case, it becomes a first-order partial differential equation.

3.2.2.5 Definition of Nonlinear Moments

Once the invariant manifold is known, the nonlinear moment associated with the generator is defined as

$$\mu(w) = h\big(\pi(w)\big). \quad (3.12)$$

This function represents the steady-state output of the nonlinear system when driven by the generator. In contrast to linear moments, which are finite-dimensional objects, nonlinear moments are generally infinite-dimensional functions. Importantly, this definition is coordinate-free and does not rely on linearization, making it suitable for strongly nonlinear systems.

3.2.2.6 Reduced-Order Model Construction

A reduced-order nonlinear model is sought in the form (3.2) where:

- $x_r(t) \in \mathbb{R}^v$ is the reduced state vector;
- $u(t) \in \mathbb{R}^m$ is the input, generated by the same signal generator used for the full-order system;
- $y_r(t) \in \mathbb{R}^p$ is the reduced output;
- $f_r : \mathbb{R}^v \times \mathbb{R}^m \to \mathbb{R}^v$ defines the reduced dynamics;
- $h_r : \mathbb{R}^v \to \mathbb{R}^p$ defines the reduced output map.

The reduced model matches the nonlinear moments of the original system if there exists a mapping

$$\pi_r : \mathbb{R}^v \rightarrow \mathbb{R}^v$$

such that

$$\frac{\partial \pi_r}{\partial w}(w)s(w) = f_r\big(\pi_r(w),\, l(w)\big), \quad (3.13)$$

and

$$h\big(\pi(w)\big) = h_r\big(\pi_r(w)\big).$$

A natural and widely adopted choice is

$$\pi_r(w) = w,$$

which directly identifies the reduced state with the generator state. Under this assumption, the reduced model can be written as

$$\dot{x_r}(t) = s(x_r) - g(x_r)l(x_r), \quad (3.14)$$
$$y_r(t) = h\big(\pi(x_r)\big).$$

Where $g : \mathbb{R}^v \rightarrow \mathbb{R}^{v\times m}$ is a free design function. The function $g(\cdot)$ represents a degree of freedom analogous to the matrix G in linear matching and can be exploited to:

- Impose local stability of the reduced model;
- Shape transient dynamics;
- Improve numerical robustness.

The choice of function $g(\cdot)$ can be trivial, for example, equal to 1, or it can be a choice made considering the dynamic characteristics of the model being reduced. Regarding the cases that will be considered in this thesis, the choice of function will be made trying to consider the dynamic characteristics of the full model. In general, it is a degree of freedom that must be used with physical criteria with respect to the model considered.

3.2.2.7 Geometric Interpretation

From a geometric viewpoint, nonlinear moment matching consists in constructing a reduced model whose invariant manifold induces the same output mapping as the original one when driven by the same generator, The nonlinear Sylvester equation (3.11) defines the invariant manifold of the full-order system, while (3.13) defines the manifold of the reduced system. Moment matching is achieved when these manifolds are output equivalent. This interpretation highlights that reduced-order models are exact steady-state realizations of selected behavior, rather than approximations in classical error-norm sense [10], [21].

3.3 Data-Driven Moment Matching

In the previous sections, moment matching has been presented within a model-based framework, where the full knowledge of the system equations allows the explicit computation of moments through Sylvester equations or invariant manifolds constructions. However, in many practical scenarios, such as large-scale thermo-fluid or electrochemical systems, the availability of an accurate state-space representation cannot be assumed. In these contexts, system behavior is often accessible only through input-output data generated by simulations or experiments, motivating the development of data-driven formulations of moment matching. The term data-driven moment matching refers to a class of approaches in which reduced-order models are constructed without explicit access to the system matrices, relying instead on measured or simulated data. Despite sharing this common objective, the existing data-driven methodologies differ substantially in their conceptual foundations, mathematical formulation, and computational interpretation. Two distinct paradigms can be identified, which will be clearly distinguished in this thesis.

The first paradigm, introduced in [22], can be interpreted as a response-based data-driven moment matching approach. In this framework, moments are not explicitly computed or estimated as algebraic quantities. Instead, they are defined implicitly through the steady-state response of the unknown system when interconnected with suitably designed signal generators. The key idea is that, under appropriate conditions, the steady-state behavior of the interconnection encodes the same information that classical moments capture in the model-based setting. As a consequence, moment matching is enforced dynamically, by ensuring that the reduced-order model reproduces the same steady-state response induced by the signal generator. This approach preserves a strong conceptual continuity with the classical theory presented in Section 3.2 while removing the need for explicit

knowledge of the system equations, and the necessity to solve (3.4) or (3.11). The second paradigm, developed more recently in [11] and [12], adopts a fundamentally different perspective by interpreting moments as unknown functions to be learned directly from data. In this case, moment matching is reformulated as a learning problem, where the objective is to estimate moment-related mappings using regression techniques in RKHS. This kernel-based formulation allows moments to be reconstructed explicitly from noisy input-output data, providing increased flexibility and robustness. Unlike the response-based approach of [22], the kernel-based framework introduces an explicit statistical learning layer, enabling regularization, noise handling, and generalization properties that are particularly attractive for large-scale and data-rich applications.

While both paradigms operate in a data-driven setting and aim at constructing reduced-order models that reproduce the dominant input-output behavior of the original system, they differ profoundly in their interpretation of moments, their reliance on dynamical versus learning-based principles, and their computational structure. For this reason, they will be treated separately in the remainder of this chapter. Section 3.3.1 focuses on the response-based data-driven moment matching framework of [22], highlighting its theoretical connection with classical moment matching. Section 3.3.2 then introduces the kernel-based learning approaches of [11] and [12], which represent a more recent extension of the data-driven paradigm.

3.3.1 Response-Based Data-Driven Moment Matching

The response-based data-driven moment matching framework proposed in [22] extends the classical moment matching theory to scenarios in which the explicit state-space representation of the system is unavailable. Unlike learning-based approaches, this methodology preserves a strong dynamical interpretation of moments and relies on the steady-state behavior of an interconnected system to implicitly encode moment information.

3.3.1.1 Unknown System Description

Consider an unknown nonlinear dynamical system described by (3.1). The only accessible information about (3.1) is its input-output response, obtained either from experiments or simulations. No assumptions are made on the availability of linearization or Jacobian matrices.

3.3.1.2 Signal Generator and Interconnection Structure

To extract moment information from data, the system (3.1) is interconnected with a signal generator of the form (3.8), which is designed by the user and plays a role analogous to the interpolation points and directions in classical moment matching. The choice of $s(\cdot)$ and $l(\cdot)$ determines the class of moments being matched The interconnection between the system and the signal generator yields the composite system (3.10).

3.3.1.3 Definition of Data-Driven Moments

Under suitable assumption on stability, regularity, and well-posed of the interconnection, the interconnected system admits an invariant manifold. In particular, the following conditions are assumed to hold:

1. Regularity of the vector fields: the functions $f(\cdot,\ \cdot)$, $h(\cdot)$ and $l(\cdot)$ are assumed to be locally Lipschitz continuous, ensuring existence and uniqueness of solution of the interconnected system for given initial conditions.
2. Existence of an equilibrium of the signal generator: the signal generator admits an equilibrium at the origin $s(0) = 0$ and its trajectories remain bounded for all admissible initial conditions $w(0)$ in a neighborhood of the origin.
3. Asymptotically stability of the generator dynamics: the equilibrium $w = 0$ of the autonomous system (3.8) is assumed to be (locally) asymptotically stable. This guarantees the existence of steady-state trajectories and allows the defintion of moments through asymptotic responses.
4. Input-to-state stability of the interconnected system: The system (3.10) is assumed to admit bounded trajectories and to converge asymptotically to a steady-state behavior induced by the generator.

Under this assumption, the interconnected system admits a locally invariant manifold (3.9), where $\pi : \mathbb{R}^v \to \mathbb{R}^n$ is a sufficiently smooth mapping satisfying the invariance condition (3.11). The existence of such a manifold implies that, asymptotically, the system state satisfies

$$x(t) \to \pi\big(w(t)\big) \quad as \quad t \to \infty.$$

The data-driven moment associated with the signal generator (3.8) is then defined as (3.12). Crucially, $\mu(w)$ can be evaluated directly from steady-state input-output data, without computing $\pi(w)$ explicitly, so without solving the partial differential equation in Sylvester form. This represents the key data-driven aspect of the approach.

3.3.1.4 Reduced-Order Model Structure

A reduced-order model of order $v \ll n$ is constructed as (3.14). By construction, the reduced model reproduces the same steady-state response of the original system when driven by the signal generator. The reduced-order model (3.14) is said to match the data-driven moments of (3.1) if

$$\lim_{t\to\infty}\big(y(t) - y_r(t)\big) = 0,$$

for all trajectories generated by the signal generator. This condition generalizes the classical moment matching property:

- Classical moments are matched algebraically via Sylvester equations;
- Data-driven moments are matched dynamically, through steady-state equivalence.

3.3.1.5 Relationship with Model-Based Moment Matching

When the system is linear and known, the invariant manifold condition reduces to a Sylvester equation of the form (3.4), where Π corresponds to the linearization of $\pi(w)$. In this case, the response-based data-driven framework recovers the model-based moment matching theory presented in Section 3.2. Therefore, the method presented in [22] can be interpreted as a nonlinear, data-driven generalization of model-based moment matching, preserving its geometric and dynamical structure while avoiding the need for explicit model knowledge. Some remarks are needed related to applicability and limitations. While the response-based data-driven approach avoids explicit model identification, it still relies on:

- The design of suitable signal generators;
- The existence of invariant manifolds;
- Asymptotic convergence to steady-state.

Moreover, moments are not estimated explicitly, which limits post-processing flexibility and motivates the kernel-based extensions introduced in Section3.3.2

3.3.2 Kernel-Based Data-Driven Moment Matching

While the response-based data-driven framework introduced in Section 3.3.1 preserves a strong dynamical interpretation of moment matching, it defines moments implicitly through the steady-state behavior of an interconnected system. As a consequence, moments are not explicitly available as algebraic or functional objects, but only through asymptotic input-output responses. Although this formulation is

theoretically consistent with the classical theory, it presents several practical limitations when applied to large-scale systems and noisy datasets. In particular, the response-based approach:

- Requires the system to reach steady-state for each excitation;
- Does not provide an explicit parametric representation of the moments;
- Does not naturally accommodate noisy measurements;
- Does not allow for regularization or statistical robustness.

These limitations motivate the development of a fundamentally different data-driven paradigm, in which moments are explicitly reconstructed from data as unknown functions. This is the perspective adopted in [11] and [12], where moment matching is reformulated as a learning problem in a functional space. Rather than encoding moment information in steady-state trajectories, the kernel-based framework aims to directly approximate the moment mapping (3.12) from input-output data, where $\pi(w)$ denotes the invariant manifold associated with the interconnection between the system and the signal generator, as introduced in Section 3.3.1. In this sense, the goal is to learn the mapping $\mu : \mathbb{R}^{v} \rightarrow \mathbb{R}^{p}$ without explicitly computing $\pi(w)$, and without relying on asymptotic convergence. This perspective shifts the focus from dynamical equivalence to functional approximation: the reduced-order model is constructed to reproduce the learned moment function $\mu(w)$ rather than the asymptotic response of the original system.

3.3.2.1 Problem Formulation and Learning-Based Moments

In the kernel-based data-driven framework proposed in [11], [12], the concept of moment is no longer defined implicitly through the steady-state response of an interconnected system but is instead interpreted as an unknown functional object to be learned from data. This shift in perspective represents a fundamental conceptual departure from both classical model-based moment matching and from the response-based data-driven formulation of [22]. Recall from Section 3.3.1 that, under suitable regularity and stability assumptions, the steady-state behavior of the interconnected system is characterized by the invariant manifold $x = \pi(w)$, where $w \in \mathbb{R}^{v}$ denotes the state of the signal generator. The associated moment mapping is defined as (3.12), which describes the steady-state output of the original system when driven by the generator. In the [10] and [22], $\mu(w)$ is either computed analytically via Sylvester equations or invariant manifold conditions or accessed implicitly through asymptotic responses. In contrast, the kernel-based approach assumes that neither the system dynamics nor the manifold mapping $\pi(w)$ is known. Instead, one is given access only to a finite dataset of the form

$$D = \{(w_i,\ y_i)\}_{i=1}^{N}\ ,\ (3.15)$$

where:

- $w_i\ \in\ \mathbb{R}^v$ denotes the generator state at which the system is excited;
- $y_i\ \in\ \mathbb{R}^p$ is the corresponding measured output of the original system.

The central objective of kernel-based data-driven moment matching is to reconstruct an approximation $\mu^*(w)$ of the true moment mapping $\mu(w)$ directly from dataset D, without requiring any explicit model of the original system. Within this framework, moments are no longer interpreted as coefficients of a Taylor expansion or as algebraic quantities associated with interpolation conditions. Instead, they are viewed as samples of an unknown nonlinear function $\mu: \mathbb{R}^v \rightarrow \mathbb{R}^p$, whose structure must be inferred from data. This interpretation allows the moment matching problem to be recast as a regression problem: given a finite set of noisy measurements $\{(w_i,\ y_i)\}$, find a function $\mu^*(w)$ such that

$$\mu^*(w)\ \approx\ y_{i,}\quad for\ i = 1, \ldots, N\ .$$

In contrast to classical identification methods, the goal here is not to reconstruct a full dynamic model of the original system, but only the specific functional mapping $\mu(w)$ that encodes the relevant moment information. This distinction is crucial: the reduce-order model will be constructed so as to reproduce the full input-output behavior of the original system. To formalize this objective, the unknown moment function (3.12) is assumed to belong to a suitable function space H, which will later be chosen as a RKHS. The learning problem can then be stated as follows: Given a dataset (3.15) , find a function $\mu^\star\ \in\ H$ that minimizes a regularized empirical cost function of the form

$$\min_{\mu^* \in H}\quad \sum_{i=1}^{N}\left|\left|y_i - \mu^*(w_i)\right|\right|^2 + \lambda\left|\left|\mu^*\right|\right|_H^2.\ (3.16)$$

Where $\lambda\ >\ 0$ is a regularization parameter. This formulation introduces a statistical dimension to moment matching: approximation accuracy, regularization, and generalization properties now play a central role. These aspects are completely absent in model-based and response-based formulations. It is important to emphasize that, although both the response-based and kernel-based operate in a data-driven setting, they differ in several fundamental aspects:

1. Implicit vs explicit moments: in response-based, moments are encoded implicitly through steady-state trajectories. In kernel-based moments are explicitly reconstructed as functions.
2. Dynamical opposite learning-based formulation: the response-based enforces moment matching dynamically through interconnections. Kernel-based approach enforces it statistically through regression.
3. Deterministic opposite stochastic robustness: the kernel-based framework naturally accommodates noise, regularization, and generalization, whereas the response-based framework is purely deterministic.
4. Flexibility of the reduced-order model: in the kernel-based approach, the reduced model is built directly from the learned moment function, enabling a richer class of approximants.

These conceptual differences motivate the introduction of the RKHS-based methods, which will be introduced in the next subsection

3.3.2.2 Reproducing Kernel Hilbert Space for Moment

The kernel-based data-driven framework introduced in [11], [12] relies on the mathematical structure of RKHS to reconstruct the unknown moment mapping $\mu(w)$ from finite and possibly noisy data. This choice is motivated by the need for a flexible, well-posed, and regularized function approximation framework capable of handling nonlinear dependencies between the generator state w and the system output y. Before formalizing the learning problem, it is therefore necessary to introduce the basic concepts of RKHS theory and to explain their relevance in the context of data-driven moment matching. Let $X \subseteq \mathbb{R}^v$ denote the domain of interest for the generator state w. A Hilbert space H of functions $f : X \to \mathbb{R}$ is a function space equipped with an inner product $\langle \cdot, \cdot \rangle_H$ and an associated norm

$$||f||_H = \sqrt{\langle f, f \rangle_H}.$$

In the present context, H is chosen such that it contains sufficiently regular functions to represent the unknown moment mapping (3.12), while simultaneously penalizing overly complex solutions. A Hilbert space H of functions defined on X is called a Reproducing Kernel Hilber Space (RKHS) if there exists a function

$$k : X \times X \to \mathbb{R},$$

called the reproducing kernel, satisfying the following properties:

1. For every fixed $w \in X$, the function $k(\cdot, w)$ belongs to H.
2. The reproducing properties $f(w) = \langle f(\cdot), k(\cdot, w)\rangle_H \quad for\ each\ f \in H$ holds.

The kernel function implicitly defines the inner product in H and therefore the geometry of the function space. The kernel $k(w, w')$ can be interpreted as a measure of similarity between two points w and w' in the input space. Common choices include:

- Gaussian (RBF) kernels;
- Polynomial kernels;
- Laplacian kernels.

In the context of moment matching, the kernel determines how the learned moment function generalizes from the observed generator states w_i to unseen values of w .One of the most important results in RKHS theory is the Representer Theorem, which states that the solution of (3.16) admits a finite-dimensional representation

$$\mu^*(w) = \sum_{i=1}^{N} K(w, w_i)\alpha_i, \text{ (3.17)}$$

where $\alpha_i \in \mathbb{R}^p$.

This result is crucial for computational tractability: although H may be infinite-dimensional, the optimal solution lies in the finite-dimensional subspace spanned by the kernel sections $k(\cdot, w_i)$. In the present context, the moment mapping (3.12) is generally vector-valued, i.e.,

$$\mu : \mathbb{R}^v \to \mathbb{R}^p.$$

To handle this case, the framework of vector-valued RKHS is employed. In this setting, the kernel becomes an operator-valued function, where in this case $X \in R^v$

$$K : X \times X \to \mathbb{R}^{p\times p},$$

the choice of RKHS in [11], [12] is motivated by several key properties:

1. Flexibility: RKHS can approximate a wide class of nonlinear functions.
2. Regularization: the RKHS norm naturally penalizes too complex functions.
3. Well-posed: regularized learning problems in RKHS are convex and admit unique solutions.
4. Noise robustness: the framework accommodates noisy measurements.
5. Finite-dimensional representation: the representer theorem ensures computational feasibility.

These properties are essential for large-scale and data-driven applications, where noise, modeling uncertainties, and limited data availability are unavoidable. Within the kernel-based data-driven moment matching framework, RKHS provides a principled way to approximate the unknown moment mapping (3.12) while enforcing smoothness and regularity constraints. Once an approximation $\mu^*(w)$ has been learned, it can be directly embedded into a reduced-order model, whose structure will be discussed in the subsequent sections.

3.3.2.3 Kernel-Based Estimation of Moment Function

In this section, the formalization of the kernel-based estimation of moment mapping (3.12) introduced in section 3.3.2.1 is given. The objective is to construct an explicit approximation $\mu^*(w)$ of the unknown moment function using a finite set of input-output data, without relying on any explicit model of the original system. Let the original unknown system be interconnected with a known signal generator of the form (3.8). For a given set of initial conditions $w(0)$, the generator produces trajectories $w(t)$, and the system is excited accordingly. Assuming that the system reaches steady-state for each excitation, one can collect a dataset of the form (3.15). The steady-state assumption ensures that

$$y_i = \mu(w_i) + \epsilon_i,$$

where $\mu(w_i)$ is the (3.12), true moment valued in the sampled generator state, and ϵ_i accounts for measurement noise, numerical errors, or unmodeled dynamics. The kernel-based framework formulates the estimation of (3.12) as a regularized regression problem in a Reproducing Kernel Hilbert Space H . Specifically, one seeks a function $\mu^* \in H$ that solve the optimization problem (3.16). Where:

- The first term enforces data fidelity;
- The second term penalizes complexity via the RKHS norm;
- $\lambda > 0$ is a regularization parameter.

This formulation balances accuracy and smoothness of the learned function. By the Representer Theorem, the solution of (3.16) admits the finite-dimensional representation (3.17), and this expansion highlights that the learned moment function is a linear combination of kernel evaluations centered at the data points. Let us define the kernel matrix **K** associated with the dataset (3.15). In the vector-valued case, **K** is a block matrix of size $Np \times Np$ whose (i, j)-th block is given by

$$\mathbf{K}_{ij} = K(w_i, w_j) \in \mathbb{R}^{p \times p}.$$

Thus:

$$\mathbf{K} = \begin{bmatrix} K(w_1, w_1) & K(w_1, w_2) & \cdots & K(w_1, w_N) \\ K(w_2, w_1) & K(w_2, w_2) & \cdots & K(w_2, w_N) \\ \vdots & \vdots & \ddots & \vdots \\ K(w_N, w_1) & K(w_N, w_2) & \cdots & K(w_N, w_N) \end{bmatrix}.$$

The kernel matrix is:

- Symmetric;
- Positive semidefinite;
- Typically dense.

In the scalar-output case ($p = 1$), this reduces to the classical Gram matrix.

Let us define the stacked coefficient vector

$$\boldsymbol{\alpha} = \begin{bmatrix} \alpha_1 \\ \vdots \\ \alpha_N \end{bmatrix} \in \mathbb{R}^{Np},$$

and the stacked output vector

$$\mathbf{y} = \begin{bmatrix} y_1 \\ \vdots \\ y_N \end{bmatrix} \in \mathbb{R}^{Np}.$$

The optimization problem reduces to solving the linear system in order to compute α

$$(\mathbf{K} + \lambda I)\alpha = \mathbf{y}, \quad (3.18)$$

where I denotes the identity matrix of dimension $Np \times Np$.

This linear system is always well-posed for $\lambda > 0$, even if **K** is not full-rank matrix. The coefficients α_i determine the contribution of each kernel basis function to the learned moment mapping. Each term $K(w, w_i)\alpha_i$ can be interpreted as a localized influence centered at the sample w_i, whose shape is governed by the kernel. Once the moment function $\mu^*(w)$ has been estimated using (3.17) with coefficient α computed from (3.18), the reduced-order model is constructed explicitly in the form (3.14). The reduced state is chosen as

$$x_r(t) \in \mathbb{R}^v, \text{ where } v \ll n,$$

i.e., it has the same dimension as the signal generator. The reduced dynamics are defined as

$$\dot{x_r}(t) = s\big(x_r(t)\big),$$

where $s(\cdot)$ is the same function used in the signal generator. This choice is not arbitrary: it ensures that the reduced state evolves on the same latent space used to parametrize the moment function. The reduced output is defined as

$$y_r(t) = \mu^*\big(x_r(t)\big).$$

Substituting the (3.17) yields

$$y_r(t) = \sum_{i=1}^{N} K(x_r(t),\, w_i)\alpha_i.$$

This representation is explicit, nonlinear in $x_r(t)$, and fully data-driven. The complexity of the reduced model evaluation is $0(Np)$ per time step. The reduced-order model construction ensures moment matching in the following sense: By definition,

$$\mu^*(w_i) \approx y_i \approx \mu(w_i),$$

which implies that the reduced model reproduces the steady-state response of the original system at the sampled generator states. Since $\mu^*(w)$ approximate (3.12) over the domain of interest, the reduced model matches the moments of the original system in a functional sense. The kernel-based formulation introduces several tunable degrees of freedom:

1. Kernel type (Gaussian, polynomial, etc.).
2. Kernel hyperparameters (bandwidth, etc.).
3. Regularization parameter λ .
4. Choice of generator dynamics $s(\cdot)$.

These choices determine the approximation class and the generalization properties of the reduced model. The main computational bottleneck is the inversion of the kernel matrix **K.** For large N , this becomes expensive, motivating the use of:

- Low-rank approximations;
- Sparse kernels;
- Inducing point methods.

These aspects are addressed in [11], [12]. The kernel-based framework transforms moment matching into a function learning problem, where the reduced-order model is explicitly constructed by embedding the learned moment mapping into a low-dimensional dynamical system. This yields:

- Explicit reduced models;
- Robustness to noise;
- Generalization,
- Nonlinear approximation capability.

3.3.2.4 Specialization to the SISO Case

The formulation presented above is fully general and applies to MIMO systems. The SISO case can be recovered as a particular instance of this framework by setting $p = 1$. In this case, the moment mapping reduces to a scalar-valued function

$$\mu : \mathbb{R}^{v} \rightarrow \mathbb{R},$$

while the kernel becomes a scalar-valued function

$$k : \mathbb{R}^{v} \times \mathbb{R}^{v} \rightarrow \mathbb{R}.$$

As a consequence, the vector-valued RKHS employed in the MIMO case reduces to a standard scalar-valued RKHS, and the associated operators admit simpler representations. In the SISO case setting, the Representer Theorem yields the scalar form of (3.17) where:

- $\alpha_i \in \mathbb{R}$ are scalar coefficients;
- $k(\cdot, \cdot)$ is a scalar-valued positive definite kernel;
- $\{w_i\}_{i=1}^{N}$ are the sampled generator states.

In the SISO case the kernel matrix **K** reduces to the classical Gram matrix

$$\mathbf{K} = \begin{bmatrix} k(w_1, w_1) & k(w_1, w_2) & \cdots & k(w_1, w_N) \\ k(w_2, w_1) & k(w_2, w_2) & \cdots & k(w_2, w_N) \\ \vdots & \vdots & \ddots & \vdots \\ k(w_N, w_1) & k(w_N, w_2) & \cdots & k(w_N, w_N) \end{bmatrix} \in \mathbb{R}^{N \times N}.$$

The coefficient vector is given by

$$\boldsymbol{\alpha} = \begin{bmatrix} \alpha_1 & \cdots & \alpha_N \end{bmatrix}^\top \in \mathbb{R}^N,$$

And the stacked output vector is

$$\mathbf{y} = \begin{bmatrix} y_1 & \cdots & y_N \end{bmatrix}^\top \in \mathbb{R}^N.$$

The estimation problem reduces to solving (3.18) in SISO case. The reduced-order model in the SISO case takes the form (3.14), where

$$y_r(t) = \mu^*\big(x_r(t)\big),$$

and μ^* is defined as (3.17). Model (3.14) is a nonlinear dynamical system of dimension v, whose output is given by a kernel expansion learned from data. In the SISO case, the kernel matrix has dimension $N \times N$, which is significantly smaller than the $Np \times Np$ matrix arising in the MIMO case. This yields:

- Lower memory requirements;
- Faster linear solves;
- Simple hyperparameters tuning.

3.4 Problem Formulation

In this section, the specific problem addressed in this thesis is formally stated. The aim is to contextualize the theoretical developments presented in the previous sections within the class of systems of interest and to precisely define the scope, assumptions, and objectives of the proposed data-driven moment matching framework when applied to large-scale models.

3.4.1 Class of Models Under Consideration

We consider nonlinear, continuous-time dynamical systems of the form (3.1). The focus of this thesis is on large-scale systems, characterized by a high-dimensional state vector, with $n \gg 1$ typically ranging from several tens to several hundreds or thousands of states. These systems arise from high-fidelity, physics-based modeling approaches commonly adopted in industrial simulation environments. In particular, the models considered in this work originate from first-principles formulations of thermo-fluid, electrochemical, and energy conversion processes.

3.4.2 Data-Driven Setting and Imperial College Tool

In many applications:

- The governing equations may be unavailable;
- The model may be proprietary;
- The system may be implemented in a simulation environment without direct access to internal variables;
- The computational complexity of the full-order model may prevent its explicit manipulation.

For these reasons, the systems considered in this thesis are treated as black-box or grey-box systems, accessible only through input-output simulations. Model order reduction is performed using a dedicated data-driven moment matching tool development at Imperial College London, which implements recent theoretical advances in data-driven moment matching. This tool provides a numerical framework for estimating system moments directly from input-output data, without requiring explicit knowledge of the underlying system equations. Consequently, the full-order models are never explicitly projected or linearized in closed form; instead, the reduction procedure relies on steady-state or transient response data generated by the full-order model.

3.4.3 Problem Statement

The central problem addressed in this thesis can now be formally stated. Given a nonlinear large-scale dynamical system (3.1) accessible only through input-output simulations, and given a signal generator (3.8) together with a dataset (3.15) the objective is to construct, using the Imperial College London data-driven moment matching tool, a reduced-order model (3.2) with the following features:

1. Dimensionality reduction.

2. Preservation of frequency-domain behavior: the reduced-model should accurately reproduce the linearized frequency-domain dynamic characteristics of the original system around different operating points, to guarantee good generalization performance. The emphasis is placed on the similarity of the Bode magnitude and phase diagrams over a prescribed frequency range.
3. Moment matching consistency: the reduced model should match a prescribed set of moments of the original system, as defined in Section 3.2 and 3.3.
4. Fully data-driven construction: the reduced-order model must be constructed using only input-output data, without explicit access to the system equations.
5. Computational efficiency: the reduced model must enable fast simulation and be suitable for real-time control applications.

3.4.4 Motivation for Control Applications

A key motivation for model reduction in this work is the enabling of advanced control strategies, such MPC. By constructing accurate reduced-order models that preserve the dominant frequency-domain behavior, it becomes possible to apply MPC and other real-time control techniques.

4 OpenModelica and Models

4.1 Introduction to OpenModelica

In modern system engineering, modeling languages and simulation environments play a central role in the development, analysis, and validation of high-fidelity dynamical models. Among them, the Modelica language has emerged as a standard for multi-domain physical modeling due to its acausal component-based formulation, which allows complex system to be composed from reusable physical components and equations without requiring explicit causal ordering of variables. Modelica supports continuous and hybrid dynamics, and its equation-based nature enables intuitive representation of energy, mass, and information flows across interconnected submodels. The philosophy of Modelica contrasts with traditional signal-flow block diagrams by emphasizing physical structure, modularity, and reuse, which are particularly valuable for large-scale applications. A comprehensive treatment of the Modelica language and its principles is available in the literature [1], which covers language semantics, simulation techniques, and industrial usage.

4.1.1 Overview of OpenModelica

OpenModelica is an open-source environment for Modelica-based modeling, simulation, optimization, and analysis. It comprises:

- A Modelica compiler (OMC);
- A simulation runtime;
- Developments tools and APIs;
- Support for scripting and batch execution.

OpenModelica is widely used in academia and industry for

- Model prototyping;
- Large-scale system simulation;
- Linearization;
- Export of simulation data.

Unlike tools that require proprietary languages or numerical solvers, OpenModelica is fully open and extensible, allowing integration with custom toolchains.
The Modelica language itself, as implemented in OpeneModelica, supports:

- Acausal modeling with equations rather than blocks;
- Multi-domain modeling such as fluid, thermal, electrical, and mechanical;
- Unit consistency checking and symbolic processing;
- Automatic index reduction and DAE handling;
- Structured simulation output for data-driven workflows.

Such features make OpenModelica an ideal platform for generating the input-output data required by data-driven MOR techniques. The design principles and implementation of OpenModelica are discussed in detail in [23], while its role in industrial-grade simulation pipelines has been treated in [24].

4.1.2 OpenModelica as a Data Generation Platform

In the context of data-driven MOR, the role of the modeling environment is not limited to simulation but extends to systematic data generation. OpenModelica enables:

1. Reproducible simulations campaigns;
2. Automated parameter sweeps;
3. Time-domain excitation design with suitable signal generator;
4. Export of input-output trajectories;
5. Steady-state extraction.

These features are essential for constructing datasets of the form

$$D = \{(u(t),\, y(t)) : t \in [0,\, T]\},$$

or, in the moment matching setting in the form (3.15). Moreover, the support for Functional Mock-up Interface (FMI) standards allows interoperability between OpenModelica and other simulation and control platforms, facilitating the use of reduced-order models in heterogeneous environments [25].

4.1.3 Relevance for Large-Scale Model Reduction

The large-scale models considered in this thesis are implemented in OpenModelica due to its ability to manage:

- High dimensional state vectors;
- Stiff dynamics;
- Multi-physics couplings.

These features make OpenModelica particularly well-suited for generating the high-quality datasets required by data-driven moment matching approaches. In this work, OpneModelica serves as the primary simulation environment for both the heat exchanger-turbine system and the SOFC model described in Sections 4.2 and 4.3.

4.2 Heat Exchanger-Turbine Model

4.2.1 System Overview and General Architecture

The system considered in this chapter consists of a coupled heat exchanger-turbine configuration, representative of a simplified but physically meaningful thermo-fluid dynamic power conversion unit. The model is implemented in OpenModelica and is assembled by interconnecting a set of parametrized component classes that describe the main physical subsystems and their interactions. Despite its conceptual simplicity, the resulting dynamical model exhibits a high degree of complexity due to the strong nonlinearities, the presence of distributed-parameter elements, the use of detailed and accurate fluid models, and the tight coupling between thermodynamic and fluid-dynamic phenomena. From a high-level perspective, the system is composed of the following main elements:

1. A counter-current heat exchanger, implemented by the class *HX*, which models the thermal interaction between a hot and a cold fluid stream.
2. A gas turbine, implemented by the class *Turbine*, which converts the thermal and mechanical energy of the working fluid into electrical power.
3. Two ideal mass flow sources, implemented by the classes *SourceIdealMassFlow* and *SourceIdealMassFlow1*, which prescribe the boundary conditions at the inlets of the hot and cold sides of the heat exchanger.

4. Two ideal pressure sinks, implemented by the classes *IdealSinkPressure* and *IdealSinkPressure1*, which enforce fixed outlet pressure conditions at the boundaries of the system.
5. A global *System* object, which specifies ambient conditions.

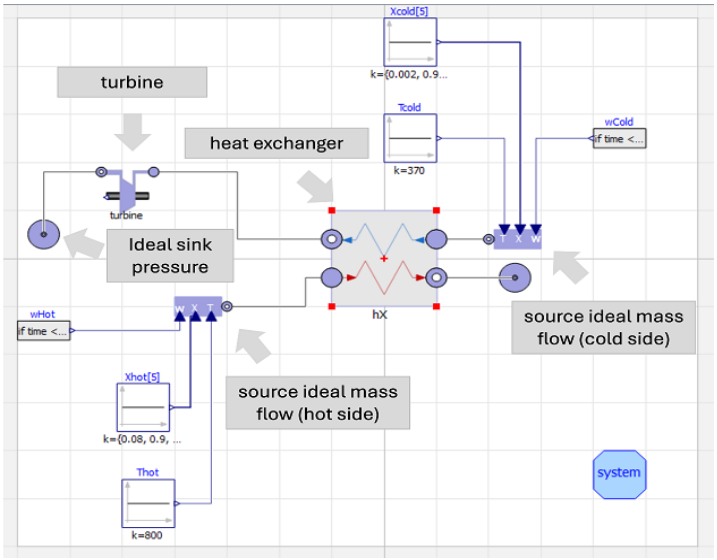


Figure 4.1: Heat Exchanger-turbine OpenModelica model

These components are interconnected to form the complete system model, implemented in the class *BaseTest*, which acts as a test bench for simulation and subsequent model reduction procedures. The physical structure of the system can be summarized as follows. Two fluid streams referred to as the hot side and the cold side enter the heat exchanger. Each stream is characterized by its mass flow rate, temperature, pressure, and chemical composition. Inside the heat exchanger, thermal energy is transferred from the hot stream to the cold stream through a finite-volume (FV) discretization of the wall and fluid domains, this kind of discretization is one of the most important contributions related to the model complexity. The cold stream, after being heated, is routed to the turbine inlet, where it undergoes an expansion process. This expansion produces mechanical power, which is subsequently converted into electrical power. The hot side of the heat exchanger is fed by the mass flow source *SourceIdealMassFlow*, which prescribes:

- The inlet mass flow rate $w_{hot}(t)$;
- The inlet temperature $T_{hot}(t)$;
- The inlet composition vector $x_{hot} \in \mathbb{R}^5$.

Similarly, the cold side is fed by *SourceIdealMassFlow1*, which prescribes:

- The inlet mass flow rate $w_{cold}(t)$;
- The inlet temperature $T_{cold}(t)$;
- The inlet composition vector $x_{cold} \in \mathbb{R}^5$.

The interconnection topology follows a physically meaningful layout and reflects the actual thermodynamic processes implemented by the model. In the present study, chemical compositions are treated as constants and are not regarded as relevant dynamic inputs. Although the employed dynamic model allow the compositions to change over time and take into account the corresponding dynamic behavior, this feature is not exploited in the MOR scenario, that assumes fixed inlet composition. In this case, the compositions can therefore be considered as "fake" inputs that remain fixed during the simulations, while the relevant control and excitation signals are the mass flow rates and temperatures of the two streams. From the perspective of system-theoretical analysis and model reduction, the interconnected model can be interpreted as a nonlinear MIMO dynamical system. The input vector is defined as:

$$\mathbf{u}(t) = \begin{bmatrix} w_{\text{hot}}(t) \\ w_{\text{cold}}(t) \\ T_{\text{hot}}(t) \\ T_{\text{cold}}(t) \\ \mathbf{x}_{\text{hot}} \\ \mathbf{x}_{\text{cold}} \end{bmatrix},$$

where the composition vectors are typically held constant. The outputs of interest are:

1. The electrical power generated by the turbine, denoted by Pe in OpenModelica.
2. The turbine inlet temperature, denoted by TIT in OpenModelica.

These outputs are selected because they are physically meaningful, measurable, and directly related to the performance of the system. Ultimately, they will have to be controlled to suitable set point, by acting on the two prescribed flow rate inputs.

The purpose of introducing this model is twofold. First, it serves as a realistic benchmark to test the theoretical reduction methods presented in Chapter 3. The model includes nonlinear constitutive relations, distributed thermal dynamics, stiff behavior, and strong coupling between subsystems, these features are typically absent in low-dimensional academic examples but are unavoidable in real engineering systems. Second, the model is used as a reference full-order system from which reduced-order model will be constructed in Chapter 5. The model, developed by DEIB, is implemented in OpenModelica using the library [27], which provides a set of high-fidelity thermo-fluid components validated in [28]. Each subsystem is defined by

a structured set of differential algebraic equations (DAEs), resulting in a large-scale nonlinear system. The use of an object-oriented modeling language such as Modelica allows for a modular and hierarchical construction of the system, in which each physical component is encapsulated in a class with well-defined interfaces.

4.2.2 Physical Interconnections and Energy Flows

This section describes in detail the physical interconnection of the components introduced in Section 4.2.1, focusing on the energy, mass, and information flow within the system. The aim is to clarify how the subsystems interact, how causality is implicitly defined by the Modelica formulation, and how the resulting coupled dynamics emerge from the component-level equations. The complete system is assembled in the class *BaseTest*, which acts as a top-level container for all the components. The topology treated in Section 4.2.1 induces a unidirectional flow of mass and energy, while allowing bidirectional exchange of information through algebraic constraints and constitutive relations.

The mass flow is imposed by the two ideal sources:

- *SourceIdealMassFlow* on the hot side;
- *SourceIdealMassFlow1* on the cold side.

These components prescribe the mass flow rate $w(t)$ at their respective outlets, enforcing the relation:

$$w_{flange} = -w_{input}.$$

This means that the source injects a prescribed mass flow into the connected component, independently of downstream conditions. This modeling choice effectively decouples the upstream thermodynamic boundary conditions from the internal dynamics of the system. The mass flows propagate as follows:

- Hot stream:
  SourceIdealMassFlow→ *HX.inhot*→ *HX.outhot* → *IdealSinkPressure*
- Cold stream:
  SourceIdealMassFlow1→ *HX.incold*→ *HX.outcold*→ *Turbine.inlet*→ *Turbine.outlet*→ *IdealSinkPressure1*

This configuration ensures that the turbine processes only the cold-side fluid after it has been thermally conditioned by the heat exchanger. The energy dynamics of the system are dominated by five main processes:

1. Convective energy transport due to mass flow.
2. Heat transfer inside the heat exchanger.

3. Mechanical-to-electrical energy conversion inside the turbine.
4. Variable mass storage in the working fluid side, due to the fluid compressibility.
5. Energy storage in the two fluids and in the in-between wall.

Heat exchanger:

The heat exchanger implements a counter-current configuration, meaning that the hot and cold streams flow in opposite directions. This arrangement maximizes the logarithmic mean temperature difference (LMTD) and therefore the efficiency of heat transfer. Thermal energy is exchanged between two streams through a discretized wall model, with FV elements. The governing principles are:

- Conservation of mass;
- Conservation of energy;
- Constitutive relations for heat transfer;
- Pressure loss relations.

The coupling between the hot and cold sides is mediated through the wall temperature field, which acts as an internal dynamic state.

Turbine:

The turbine converts part of the enthalpy drop of the fluid into mechanical power, and subsequently into electrical power. This conversion is modeled through:

- An isentropic efficiency parameter η_{iso};
- A mechanical efficiency η_{mech};
- An electrical efficiency η_{elec}.

The electrical power output is one of the key system outputs used later for model reduction. The system is closed by the use of ideal pressure sinks, in particular *IdealSinkPressure* on the hot outlet of the heat exchanger, representing the connection to a stack, and *IdealSinkPressure1* on the turbine outlet, representing the connection to a condenser at fixed low pressure. These components impose fixed outlet pressures, effectively removing pressure dynamics from the downstream side, and stabilizing numerical integration. From a modeling perspective, this choice ensures well-posed of the DAE system, and clear definition of thermodynamic reference conditions. In OpenModelica, causality is not imposed explicitly but emerges from the system of equations. Nevertheless, one can identify a conceptual signal flow:

- Prescribed signals: mass flow rates, inlet temperatures, inlet compositions;
- Propagated variables: pressures, enthalpies, densities;
- Computed outputs: turbine inlet temperature (*TIT*), electrical power (*Pe*).

Although OpenModelica uses an acausal formulation, the effective computational graph follows the physical direction of energy and mass propagation. The overall system is strongly nonlinear due to:

- Nonlinear fluid properties;
- Temperature-dependent material parameters;
- Nonlinear relations in the mass flow rate's law of the turbine;
- Nonlinear heat transfer coefficients.

Furthermore, the presence of FV discretization in the heat exchanger introduces many internal states, making the system high-dimensional and stiff. This characteristic makes the model particularly suitable as a benchmark for nonlinear model reduction techniques.

4.2.3 Detailed Description of the Main Components

This section provides a detailed description of the main components composing the heat exchanger-turbine system under investigation. The aim is to bridge the gap between the physical interpretation of each subsystem and its actual implementation in OpenModelica, highlighting how the governing equations and modeling assumptions are encoded into the simulation framework. The corresponding OpenModelica implementations are reported in Appendix B, where the full code is provided for reproducibility.

4.2.3.1 Turbine Model

The turbine is modeled as a nonlinear thermodynamic device, converting the enthalpy drop of a compressible fluid into mechanical and electrical power. The model combines a mass flow law (Stodola's law) an energy balance, and an isentropic efficiency relation. The full implementation of this component is provided in appendix B.2. Let the inlet and outlet thermodynamic states of the fluid be characterized by the following variables:

- $p_{in}(t) \in \mathbb{R}_{>0}$ is inlet pressure;
- $p_{out}(t) \in \mathbb{R}_{>0}$ is outlet pressure;
- $T_{in}(t) \in \mathbb{R}_{>0}$ is inlet temperature;
- $T_{out}(t) \in \mathbb{R}_{>0}$ is outlet temperature;
- $h_{in}(t)$ is inlet specific enthalpy;
- $h_{out}(t)$ is outlet specific enthalpy;
- $s_{in}(t)$ is inlet specific entropy;
- $\rho_{in}(t)$ is inlet density;
- $w(t)$ is mass flow rate.

The turbine produces:

- $P_t(t)$ is thermal power;
- $P_m(t)$ is mechanical power;
- $P_e(t)$ is electrical power.

The latter constitutes one of the main outputs of the global system.
The mass flow rate through the turbine is modeled using Stodola's law, which relates the flow rate to the upstream thermodynamic conditions and the pressure ratio across the turbine:

$$w(t) = K_t \sqrt{\rho_{in}(t) p_{in}(t)} \sqrt{1 - \frac{1}{PR(t)^2}},$$

where

$$PR(t) = \frac{p_{in}(t)}{p_{out}(t)},$$

is the pressure ratio, and $K_t > 0$ is a coefficient depending on the turbine geometry. This relation is nonlinear and introduces a strong coupling between thermodynamic states and flow dynamics. In the OpenModelica implementation provided in Appendix B.2, this equation appears as:

```
w = Kt*sqrt(pin*rho)*sqrtReg(1 - (1/PR)^2);
```

where a regularized square root function is employed to improve numerical robustness near singular operating conditions. The turbine is modeled as a steady-flow energy conversion device. The thermal power extracted from the fluid is given by:

$$P_t(t) = w(t)\big(h_{in}(t) - h_{out}(t)\big).$$

This quantity is converted into mechanical and electrical power through constant efficiencies:

$$\big(P_m(t) = \eta_{mech} P_t(t), \qquad P_e(t) = \eta_{elec} P_m(t)\big),$$

Where:

- $\eta_{mech} \in (0, 1)$ is the mechanical efficiency;
- $\eta_{elec} \in (0, 1)$ is electrical efficiency.

In the OpenModelica code provided in Appendix B.2, these relations are implemented as:

```
Pt = w*(fluidIn.h - hout);
Pm = Pt*eta_mech;
Pe = Pm*eta_elec;
```

To account for irreversibilities, the turbine model introduces an isentropic efficiency η_{iso}, defined as:

$$\eta_{iso} = \frac{h_{in} - h_{out}}{h_{in} - h_{out}^{iso}},$$

where h_{out}^{iso} is the outlet enthalpy corresponding to an ideal isentropic expansion. Rearranging, one obtains

$$h_{out} = h_{in} - \eta_{iso}\left(h_{in} - h_{out}^{iso}\right).$$

The isentropic outlet state is defined by the constraint:

$$s_{out}^{iso} = s_{in}$$

where s_{out}^{iso} is the isentropic outlet entropy, this formulation appears in the OpenModelica code provided in Appendix B.2 as:

```
delta_h_iso = fluidIn.h - fluidOutIso.h;
hout = fluidIn.h - eta_iso_nom*delta_h_iso;
fluidOutIso.s = fluidIn.s;
```

The turbine is strongly coupled to the surrounding components through:

- Time-varying inlet conditions;
- Nonlinear algebraic relations;
- Implicit thermodynamic state equations.

As a consequence, the turbine contributes to the global system as a nonlinear algebraic component. The full interconnection structure is reported in Appendix B.5.

4.2.3.2 Heat Exchanger Model

The heat exchanger is modeled as a distributed-parameter system describing the thermal interaction between two fluid streams (hot and cold side) and a separating solid wall. The model accounts for spatial temperature gradients, convective heat transfer, and thermal storage effects. To obtain a FV representation suitable for simulation and control-oriented analysis, the underlying partial differential equations are discretized in space. This modeling choice results in a high-order, nonlinear dynamical system, whose dimension scales with the number of spatial cells adopted for the discretization. This feature is a primary source of the large-scale nature of the overall heat exchanger-turbine model. The full OpenModelica implementation of the counter-current heat exchanger component, including the FV discretization and wall coupling, is reported in Appendix B.3.
Let

$$x \in [0, L],$$

denote the spatial coordinate along the longitudinal direction of the heat exchanger, where $L > 0$ is the total length of the device. The exchanger is partitioned into

$$N \in \mathbb{N},$$

control volumes of equal length

$$\Delta x = \frac{L}{N}.$$

For each cell $i = 1, \ldots, N$, three lumped temperatures are defined:

- $T_{h,i}(t)$ is the hot fluid temperature;
- $T_{c,i}(t)$ is the cold fluid temperature;
- $T_{w,i}(t)$ is the wall temperature.

These variables constitute the main state components of the heat exchanger subsystem. Their numerical realization in the OpenModelica environment is implemented through the *Flow1DFV* and *MetalWall1FV* classes, whose definitions and numerical structure are reported in Appendix B.3. For each control volume i, the

thermal dynamics of the hot fluid, wall, and cold fluid are governed by three coupled energy balance equations.

Let:

- $\rho_{h,i}(t) = \rho\left(T_{h,i}(t), p_{h,i}(t), X_i(t)\right)$ be the hot fluid density of the i -th volume, and $X_i(t) = \left[x_{1,i}(t), \ldots, x_{N,i}(t)\right]$ the fluid composition of the i -th volume, where N is the total number of species into the composition;
- $c_{p,h,i}(t) = \sum_{k=1}^{N} X_{k,i}(t) c_{p,k}\left(T_{h,i}(t)\right)$ be the mean specific heat capacity of the mixture, and $c_{p,k}(T)$ is the specific heat of the k -th species;
- A_h be the cross-sectional area;
- $\dot{m}_h(t)$ be the mass flow rate;
- U_{hw} be the heat transfer coefficient between hot fluid and wall.
- $h_{h,i}(t) = \sum_{k=1}^{N} X_{k,i}(t) h_k\left(T_{h,i}(t)\right)$ be the mean specific enthalpy of the mixture and $h_k(T)$ is the specific enthalpy of the k -th species.

The energy balance for the hot fluid side considering variable composition of the mixture reads:

$$\rho_{h,i}(t)\,\bar{c}_{p,h,i}(t)\,A_h \Delta x \frac{dT_{h,i}(t)}{dt} = \dot{m}_h(t)\left(\bar{h}_{h,i-1}(t) - \bar{h}_{h,i}(t)\right) - U_{hw} A_{hw}\left(T_{h,i}(t) - T_{w,i}(t)\right) + \Phi_{comp,i}(t)$$

Where the function $\Phi_{comp,i}(t)$ takes into account the energy contribution of the time varying mass fractions, and density

$$\Phi_{comp,i}(t) = -\rho_{h,i}(t)\,A_h \Delta x \sum_{k=1}^{N} h_k(T_{h,i}(t))\,\frac{dX_{k,i}(t)}{dt} - \bar{h}_{h,i}(t)\,A_h \Delta x \frac{d\rho_{h,i}(t)}{dt}$$

In the OpenModelica code, these dynamics are embedded in the *hotside* channel component, which is instantiated within the *HeatExchangerCounterCurrent* class provided in Appendix B.3.

Let:

- ρ_w be the wall density;
- $c_{p,w}$ be the wall specific heat;
- V_w be the wall volume in cell i ;
- U_{hw},U_{wc}be the heat transfer coefficients.

Then the energy balance for the wall:

$$\rho_w c_{p,w} V_w \frac{dT_{w,i}(t)}{dt} = U_{hw} A_{hw} \left(T_{h,i}(t) - T_{w,i}(t)\right) + U_{wc} A_{wc} \left(T_{c,i}(t) - T_{w,i}(t)\right).$$

This equation is realized in the *MetalWallFV* component, which explicitly models the thermal inertia of the separating wall. The corresponding OpenModelica implementation is reported in Appendix B.3.

Let:

- $\rho_c, c_{p,c}, A_c$ be the corresponding quantities for the cold stream;
- $\dot{m_c}(t)$ be the mass flow rate.

The energy balance for the hot fluid side without considering variable composition of the mixture reads:

$$\rho_c c_{p,c} A_c \Delta x \frac{dT_{c,i}(t)}{dt} = \dot{m}_c(t) c_{p,c} \left(T_{c,1+1}(t) - T_{c,i}(t)\right) - U_{wc} A_{wc} \left(Tc, i(t) - T_{w,i}(t)\right).$$

The cold-side dynamics are implemented in the *coldside* channel of the *HeatExchangerCounterCurrent* model, whose full source code is reported in Appendix B.3. The model adopts a counter-current configuration, meaning that the hot and cold fluids flow in opposite directions. This is reflected using $T_{h,i-1}$ as upstream value for the hot side and $T_{c,i+1}$ as upstream value for the cold side. This structure introduces a nontrivial coupling pattern between the spatially distributed states, in the OpenModelica implementation, the counter-current topology is enforced through the *HeatExchangerTopologyFV* object, which defines the wall-to-fluid connectivity (see Appendix B.3). Let the full state trasposed vector of the heat exchanger be defined as:

$$x_{HX}(t) = \left[T_{h,1}(t) \ldots T_{h,N}(t)\ T_{w,1}(t) \ldots T_{w,N}(t)\ T_{c,1}(t) \ldots T_{c,n}(t)\right] \in \mathbb{R}^{3N}$$

Then the heat exchanger dynamics can be written as

$$\dot{x_{HX}}(t) = f_{HX}(x_{HX}(t), u_{HX}(t), \theta),$$

where:

- $u_{HX}(t)$ collects the inlet temperatures and mass flow rates;
- θ is the vector of physical parameters;
- f_{HX} is a nonlinear vector field induced by the convective and conductive terms.

This structure corresponds to the internal representation generated by the OpenModelica compiler from the component-based formulation reported in Appendix B.3. The dimension of the heat exchanger subsystem scales linearly with the number of finite volumes:

$$\dim(x_{HX}) = 3N.$$

In practical implementations, N is chosen large enough to accurately resolve spatial temperature gradients, resulting in tens or hundreds of dynamic states. Moreover, the presence of time-varying mass flow rates, nonlinear thermophysical properties, and nonlinear heat transfer correlations, makes the subsystem strongly nonlinear. These nonlinearities are directly visible in the source code through the property evaluations and heat transfer models included in the *Flow1DVF* components reported in Appendix B.3. The heat exchanger subsystem:

- Introduces the majority of the system's dynamic states;
- Dominates the overall model order;
- Generates stiff dynamics;
- Contributes to differential-algebraic coupling.

As such, it plays a crucial role in determining the computational complexity of the full boiler-turbine model and is a primary target for model order reduction.

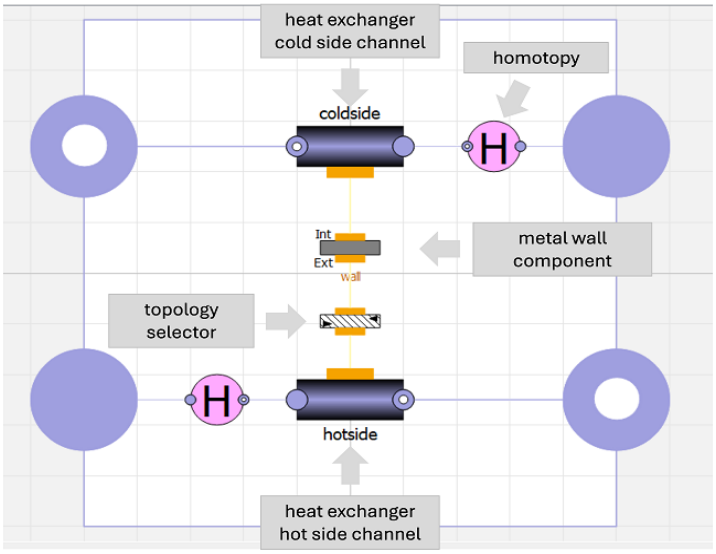


Figure 4.2: Components of the heat exchanger model

4.2.3.3 Boundary Components

The boundary components of the heat exchanger-turbine system play a crucial role in defining the thermodynamic operating conditions and in closing the system of equations arising from the interconnection of the physical subsystems, In the present model, two idealized boundary elements are employed:

- An ideal mass flow source implemented with *SourceIdealMassFlow* class, prescribing the inlet mass flow rate and thermodynamic state;
- An ideal pressure sink implemented with *IdealSinkPressure* class, imposing a fixed outlet pressure condition.

Although these components do not introduce additional dynamic states, they strongly influence the system's behavior by enforcing algebraic constraints on the flow variables. Their implementation is reported in Appendix B.4. Now a mathematical description of the boundary components within the system will be provided.

Ideal Mass Flow Source:

The *SourceIdealMassFlow* components represent idealized inlet boundary conditions. Contrary to simplified flow sources that only impose a mass flow rate, these components enforce a complete set of thermodynamic variables at the corresponding fluid port. More precisely, each mass flow source imposes:

- The mass flow rate:

$$\dot{m}_p(t) = \dot{m}_{ref},$$

- The temperature:

$$T_p(t) = T_{ref},$$

- The specific enthalpy:

$$h_p(t) = h_{ref},$$

- The fluid composition:

$$X_p(t) = X_{ref},$$

Where the subscription p denotes the corresponding fluid port and the subscription ref quantities are prescribed input values. These relations act as algebraic constraints on the port variables. No additional dynamic equations are introduced by the source.

Ideal Pressure Sink:

The ideal pressure sink enforces a fixed pressure condition at the outlet of the fluid network. Let $p(t)$ denote the pressure at the connected port. The defining equation is:

$$p_p(t) = p_{ref},$$

where $p_{ref} > 0$ is a constant parameter. In contrast to the mass flow source, the pressure sink does not prescribe the flow rate or enthalpy; these quantitates are determined by the upstream components. The pressure sink introduces the algebraic constraint:

$$p_p(t) - p_{ref} = 0,$$

this equation eliminates pressure drift and ensures that the system admits a well-defined thermodynamic equilibrium. The ideal pressure sink is implemented as the class *IdealSinkPressure*. Its full definition is reported in Appendix B.4. Let the global system be represented as semi-explicit DAEs form:

$$\dot{x}(t) = f\big(x(t), z(t), u(t)\big),$$
$$0 = g\big(x(t), z(t), u(t)\big).$$

where:

- $x(t)$ are dynamic states;
- $z(t)$ are algebraic variables;
- $u(t)$ are external inputs.

The boundary components primarily affect the algebraic equations $g(\cdot)$ by imposing constraint on pressure, enthalpy, and flow variables. As such, they influence the solvability of the algebraic loop, the index of the DAE system, the numerical stiffness, and the admissible steady-state configurations. So, the boundary components:

- Enforce physically meaningful inlet and outlet conditions;
- Introduces algebraic constraints into global DAE system;
- Influence the equilibrium configurations;
- Close the thermodynamic network.

Their OpenModelica implementations are provided in Appendix B.4, and their role in the top-level interconnection is detailed in Appendix B.5. Moreover, an overview of how OpenModelica simulates models in Appendix A.

4.2.3.4 System Object and Global Environment

The heat exchanger-turbine model is embedded within a global system object that defines the thermodynamic environment. This object does not represent a physical device but acts as modeling infrastructure layer that ensures environment conditions. In the OpenModelica implementation, this role is fulfilled by the *System* class, whose complete definition is reported in Appendix B.1. The system object defines a set of reference environmental quantities, such as:

- Ambient pressure:

$$p_{amb} > 0,$$

- Ambient temperature:

$$T_{amb} > 0,$$

- Reference enthalpy:

$$h_{ref},$$

- Reference entropy:

$$s_{ref}.$$

These values are used to initialize thermodynamic states, and to compute relative quantities. In the OpenModelica implementation, these quantitates are declared as *parameter* fields of the *System* class.

4.3 Solid Oxide Fuel Cell (SOFC) Dynamic Model

This section presents a detailed physical and mathematical description of the Solid Oxide Fuel Cell (SOFC) model developed by DEIB, and adopted in this thesis. The formulation follows the control-oriented, one-dimensional modeling framework proposed in [26] and implemented within the library [29]. The objective is to provide a clear and rigorous interpretation of the internal thermochemical and electrochemical processes that ultimately lead to electrical power generation, while emphasizing the strong nonlinear coupling and distributed nature of the resulting dynamical system. Unlike conventional power generation systems, the SOFC directly converts chemical energy into electrical energy through electrochemical reactions occurring at high temperatures. This feature enables high efficiency, fuel flexibility, and internal fuel processing, but also results in complex multi-physics dynamics involving mass transport, chemical kinetics, charge transfer, and heat exchange.

4.3.1 Electrochemical Conversion Cycle

The operation of the SOFC can be interpreted as a closed electrochemical cycle involving the continuous circulation of oxygen ions and electrons between the cathode and anode. At the cathode inlet, a hot oxidant stream composed primarily of oxygen and carbon dioxide is supplied to the cell. At the anode inlet, a methane-rich fuel stream is introduced, through Water Gas Shifting (WGS) and Steam Reforming (SR) chemical reactions hydrogen is generated from a methane-rich fuel stream.

The interaction between these two streams, mediated by the electrolyte and the external electrical circuit, is responsible for the generation of electrical power. At the cathode side, molecular oxygen undergoes electrochemical ionization. This process consumes electrons arriving from the external circuit and produces negatively charged oxygen ions. These ions are the only species allowed to pass through the dense ceramic electrolyte, which is selectively conductive to oxygen ions while remaining impermeable to electrons and neutral molecules. As a result, charge separation is enforced, and electrons are compelled to flow through the external circuit rather than through the electrolyte. Once the oxygen ions cross the electrolyte and reach the anode side, they participate in oxidation reactions with the fuel-derived species present in the anode channel, in particular Carbon Oxidation Reaction (COR) and Hydrogen Oxidation Reaction (HOR). These reactions release electrons, which are then transported back to the cathode through the external circuit.

In this way, the electrochemical loop is closed: electrons produced at the anode sustain the oxygen ionization reaction at the cathode, while the ions flux through the electrolyte balances the electronic current in the external circuit. From the electrons flux from the anode to the cathode there is a current generation, and so an electrical power generation. The continuous repetition of this cycle is the fundamental mechanism by which the SOFC converts chemical energy into electrical power.

4.3.2 Cathode Electrochemistry and Oxygen Ions Generation

At the cathode, molecular oxygen undergoes electrochemical reduction according to the oxygen reduction reaction (ORR):

$$\frac{1}{2}O_2 + 2e^- \rightarrow O^{2-} \quad (4.1)$$

Where:

- O_2 is oxygen;
- $e^-$is electrons;

- O^{2-} is oxygen ions.

This reaction is enabled by the arrival of electrons from the external electrical circuit and is strongly temperature dependent. The cathode thus plays a dual role:

- It acts as a sink for electrons;
- It serves as the source of oxygen ions that sustain the electrochemical conversion.

The reaction (4.1) is activated by the high operating temperature of the SOFC and is facilitated by the catalytic properties of the cathode material. From a modeling perspective, this reaction establishes a direct relationship between the electrical current flowing in the external circuit and the rate of oxygen ion generation. Specifically, the molar flow rate of oxygen ions is proportional to the current density through Faraday's law

$$\dot{n} = \frac{1}{zF}.$$

Where:

- $\dot{n}$ is the molar flow rate of reacting species;
- z is the number of electrons transferred per mole;
- F is Faraday's constant.

The cathode thus acts simultaneously as an electrochemical reactor and as an interface between the electrical and ionic domains of the system. In the model presented in [26], the cathode behavior is embedded within distributed formulation that accounts for mass transport, electrochemical kinetics, and heat generation.

4.3.3 Oxygen Ions Transport Through the Electrolyte

The electrolyte, often referred to as part of the PEN (Positive Electrolyte Negative) structure, plays a central role in the SOFC operation. Its selective ionic conductivity ensures that only oxygen ions can pass from the cathode to the anode, while electrons are forced to travel through the external load. The ionic flux through the electrolyte is directly linked to the electrical current by

$$\dot{n_{O^{2-}}} = \frac{I}{2F}.$$

Where:

- $\dot{n_{O^{2-}}}$ is the molar flow rate of the oxygen ions;
- I is the total cell current;

- F is Faraday's constant.

This relation enforces charge conservation and couples electrochemical reactions to electrical power generation. In the SOFCPoliMi model described in [26], electrolyte transport losses are incorporated into the ohmic loss term, contributing to the overall voltage drop of the cell. The topic of voltage losses will be treated in this chapter further on.

4.3.4 Fuel Processing and Anode Thermochemical Reactions

A distinctive feature of high-temperature SOFCs is their ability to internally process hydrocarbon fuels. In the system considered in this thesis, methane is supplied to the anode inlet and undergoes two thermochemical reactions before participating in electrochemical oxidation. The two thermochemical reactions are:

- WGS;
- SR.

4.3.4.1 Steam Reforming

Due to the high temperature of the SOFC, methane can be internally reformed within the anode channel. The dominant reaction is SR

$$CH_4 + H_2O \rightarrow CO + 3H_2 \quad (4.2)$$

Where:

- CH_4 is methane;
- H_2O is water;
- CO is carbon monoxide;
- H_2 is hydrogen.

This reaction is strongly endothermic and is favored at high temperatures. In the SOFC environment, the heat required for SR is partially provided by the exothermic electrochemical reactions, creating a tight coupling between chemical kinetics and thermal dynamics. From a modeling standpoint, SR introduces nonlinear source terms in the species balance equations and contributes significantly to the overall energy balance of the anode channel. The reaction (4.2) serves a crucial role by converting methane which cannot be electrochemically oxidized directly into hydrogen and carbon monoxide, which are electrochemically active species.

4.3.4.2 Water-Gas Shift Reaction

In parallel with steam reforming, the WGS reaction occurs within the anode:

$$CO + H_2O \rightarrow CO_2 + H_2 \quad (4.3)$$

Where:

- CO is carbon monoxide;
- H_2O is water;
- CO_2 is carbon dioxide;
- H_2 is hydrogen.

This reaction is mildly exothermic and plays a key role in regulating the relative concentrations of H_2, CO ,H_2O, and CO_2. The reaction (4.3) redistributes chemical energy between carbon monoxide and hydrogen and determines the composition of electrochemically active species. In [26], the WGS reaction is modeled either through equilibrium constraints or through finite-rate kinetics, depending on the desired level of fidelity. Its inclusion ensures a realistic representation of the anode gas composition and its evolution along the cell length. Together, SR and WGS reactions enable methane, which cannot be directly oxidized electrochemically, to be transformed into hydrogen and carbon monoxide, which are the actual electrochemical fuels of the SOFC.

4.3.5 Anode Electrochemical Oxidation Reactions

At the anode side of the SOFC, the oxygen ions O^{2-} transported through the electrolyte participate in electrochemical oxidation reactions with the fuel-derived species. These reactions are the fundamental source of electron production and therefore of electrical power generation, Two electrochemical oxidation reactions dominate this stage:

- HOR;
- COR.

4.3.5.1 Hydrogen Oxidation Reaction

The HOR is given by:

$$H_2 + O^{2-} \rightarrow H_2O + 2e^- \quad (4.4)$$

Where:

- H_2 is hydrogen;
- $O^{2-}$is oxygen ions;
- H_2O is water;

- $e^-$is electron.

The (4.4) converts molecular hydrogen into water vapor while releasing two electrons per mole of hydrogen consumed. From an electrochemical perspective, HOR is typically characterized by fast kinetics at SOFC operating temperatures (700-1000 °C), making it the dominant contributor to current generation whenever hydrogen is available in sufficient concentration.

4.3.5.2 Carbon Monoxide Oxidation Reaction

The COR is given by:

$$CO + O^{2-} \rightarrow CO_2 + 2e^-.$$

COR produces carbon dioxide and releases the same number of electrons as HOR per mole of reactant. Although generally slower than HOR, COR plays a crucial role in SOFCs operating with internally reformed fuels, as it enables direct electrochemical utilization of carbon monoxide by SR and WGS reactions. In the model described in [26], the total current density is expressed as the combined contribution of HOR and COR, with the local reaction rates depending on species concentrations, temperature, and electrochemical overpotentials. This formulation ensures that both hydrogen and carbon monoxide oxidation contribute to the electrical output in a physically consistent manner. The coupling between HOR, COR, and the upstream reforming reactions creates a tightly interconnected reaction network:

- SR and WGS determine the availability of electrochemically active species;
- HOR and COR regulate electrical current and heat generation.

4.3.6 Closure of the Electrical Cycle and Power Generation

The SOFC operates as a closed electrochemical system in which ionic and electronic charge transport are strictly coupled. Oxygen ions generated at the cathode migrate through the electrolyte to the anode, where they are consumed by HOR and COR. The electrons released by these reactions flow through the external electrical circuit back to the cathode, sustaining the ORR. This closed-loop mechanism enforces global charge conservation and establishes a direct relationship between chemical reaction rates and electrical power generation. The electrical power delivered by the cell is given by:

$$P_e = V_{cell} I,$$

where:

- V_{cell} is the operating cell voltage;
- I Is the total cell current.

The cell voltage is modeled as:

$$V_{cell} = E_{Nernst} - \eta_{act} - \eta_{ohm} - \eta_{con}.$$

Where the three η terms represent irreversible losses. The activation losses η_{act} accounts for finite-rate electrochemical kinetics at the anode and cathode interfaces and depends nonlinearly on current density and temperature. The ohmic losses η_{ohm} captures ionic resistance in electrolyte and electronic resistance in electrodes and interconnections. This term decrease linearly with current increment, and is strongly temperature dependent. The concentration losses η_{conc} represents mass transport limitations due to finite diffusion rates in electrodes and gas channels. These losses reduce the actual operating voltage relative to the reversible thermodynamic limit and play a central role in shaping the cell's performance. The reversible open-circuit voltage is given by the Nernst equation

$$E_{Nernst} = E^0(T) + \frac{RT}{2F} \ln\left(\frac{p_{O_{2,cat}} p_{H_{2,cat}}}{p_{H_2O,an}}\right), \quad (4.5)$$

where:

- R is the universal gas constant;
- T is the temperature;
- F is Faraday's constant;
- $p_{O_{2,cat}}$is partial pressure of oxygen ions at the cathode;
- $p_{H_{2,cat}}$ is the partial pressure of hydrogen at the cathode;
- $p_{H_2O,an}$is the partial pressure of water at the anode.

The Nernst potential represents the maximum theoretical voltage obtainable from the electrochemical reactions under given thermodynamic conditions. Its dependance on temperature and partial pressures makes it a key link between chemical composition, thermal state, and electrical behavior. In [26], the Nernst equation provides the baseline voltage from which all irreversible losses are subtracted, ensuring thermodynamic consistency of the model. The relationship between cell voltage and current density defines the polarization curve, a fundamental characteristic of SOFC behavior. As discussed in [26], the polarization curve exhibits three main regions:

1. Activation-controlled region dominated by electrochemical kinetics at low current densities.
2. Ohmic-controlled region, characterized by an approximately linear voltage drop with current.
3. Concentration-controlled region, where mass transport limitations cause a sharp voltage decline.

The polarization curve defined in [26] encapsulates the combined effect of HOR, COR, transport phenomena, and losses. Note that this curve is not explicitly represented in the model, but it rather a result of all the detailed and interecting phenomena described therein.

4.3.7 Inputs and Outputs of the SOFC Model

From a system-theoretic perspective, the SOFC can be viewed as a nonlinear dynamical system with well-defined inputs and outputs. The inputs are:

- Anode inlet mass flow rate and composition (CH_4, H_2, CO, CO_2);
- Cathode inlet mass flow rate and composition (O_2, CO_2);
- Inlet temperature at anode and cathode;
- Electrical load or imposed current.

The main outputs are:

- Electrical power P_e,;
- Cell voltage V_{cell};
- Outlet gas composition at anode and cathode;
- Thermal power and temperature distributions.

4.3.8 Sources of Nonlinearity

The SOFC model exhibits strong nonlinearities and high dimensionality due to two primary contributions. First, electrochemical dynamics introduce nonlinear relationships through reactions kinetics of HOR and COR, exponential activation overpotentials, Nernst voltage dependance on partial pressures, and coupling between current density and species consumption. Second, spatial discretization of gas channels and solid structure leads to a large number of dynamic states, FV discretization of anode, cathode, and electrolyte domains results in distributed mass and energy balance whose dimensions scales with the number of control volumes. The combination of nonlinear electrochemical phenomena and spatially distributed dynamics yields a stiff, large scale DAE system, as emphasized in [26], making the SOFC an ideal candidate for model order reduction.

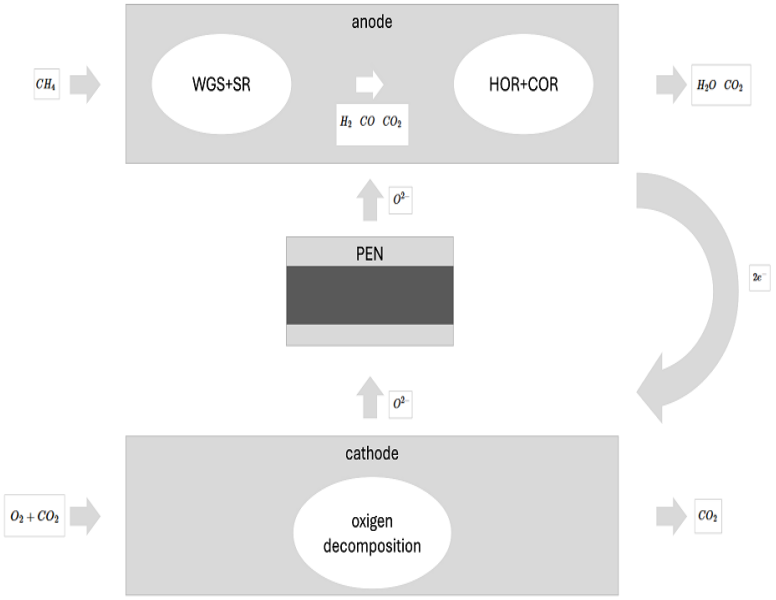


Figure 4.3: Conceptual schematization of SOFC operating cycle

4.4 Structural Overview of the SOFC Model Implementation

This section concludes the chapter by providing a structured overview of the OpenModelica implementation of the SOFC model adopted in this work. Rather than focusing on source code details, which are publicly available and extensively documented in the library [29], the emphasis is placed on the conceptual organization of the model, the role of its main classes, and the inheritance structure through which the physical phenomena described in the previous section are realized. The SOFC model developed by Politecnico di Milano is described in detail in [26]. The library follows a component-oriented and object-oriented modeling paradigm, in which complex multiphysics behavior emerges from the interconnection of well-defined subsystems.

4.4.1 Top-Level Model and System Integration

At the highest level, the SOFC model is instantiated through a benchmark model (e.g. *BenchmarkCammarata*), which serves as a complete test case for simulation and validation purposes. This model assembles the main fuel cell stack, boundary components, and electrical load, and defines the operating conditions for the simulations considered in this thesis. The core electrochemical device is represented by the class *StackCammarata*, which extends the more general class *Components.FuelCell.Stack*. This stack model encapsulates the distributed

representation of the SOFC and acts as the main interface between the electrochemical subsystem and the surrounding plant components. The top-level model connects the stack to ideal mass flow sources and pressure sinks on both anode and cathode sides, as well as to an externally imposed electrical load. In this way, the SOFC is embedded in a system-level context consistent with the energy conversion framework discussed in [26].

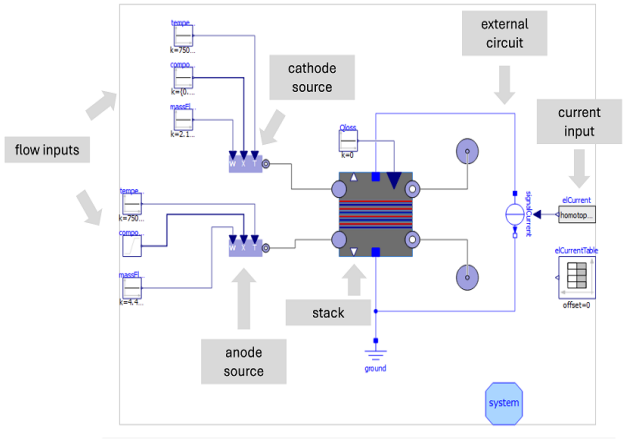


Figure 4.4: SOFC OpenModelica model (*BenchmarkCammarata)*

4.4.2 Stack Discretization and Modular Structure

The *StackCammarata* class represents the SOFC stack as a one-dimensional array of elementary repeating units, referred to as modules. Each module corresponds to a local section of the planar cell and captures the coupled thermo-fluid and electrochemical behavior at that spatial location. This discretization strategy allows spatial gradients of temperature, species composition, and current density to be resolved along the flow direction, while maintaining a level of complexity suitable for dynamic system analysis. The number of modules is a user-defined parameter and directly determines the order of the resulting dynamical system. Each spatial module is implemented through the class *Module*, which constitutes the fundamental building block of the stack.

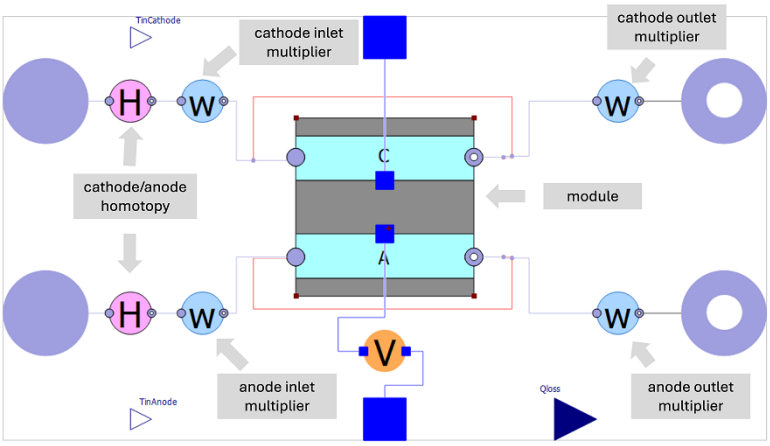


Figure 4.5: Stack OpenModelica model

4.4.3 Module Class and Physical Composition

The *Module* class implements the local coupling between gas channels, solid structures, and electrochemical interfaces. Rather than relying on deep inheritance hierarchies, the module is primarily constructed through composition, explicitly instantiating the subcomponents that represent the relevant physical domains.
In particular, each module contains:

- An anode gas channel implemented via the class *AnodeChannel;*
- A cathode gas channel implemented via the class *CathodeChannel;*
- A solid electrochemical structure represented by the *PEN* class;
- Anodic and cathodic interconnected plates modeled as thermal solids.

The *Module* class enforces mass, energy, and charge conservation at the local level and provides the interface through which neighboring modules exchange flow and thermal informations. This modular construction closely mirrors the physical structure of a planar SOFC and facilitates both scalability and model clarity.

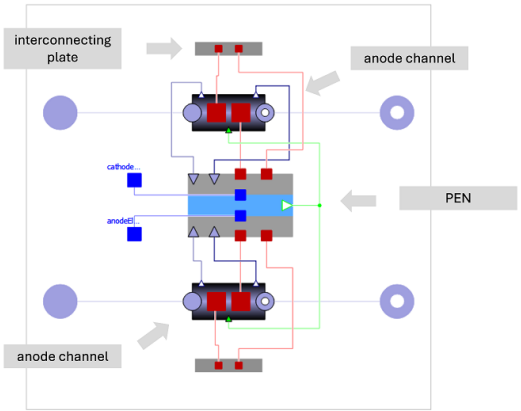


Figure 4.6: Module OpenModelica model

4.4.4 Anode Subsystem Class

The anodic side of the cell is modeled by the *AnodeChannel* class, which extends a common base class for gas channel named *BaseChannel*. This base class defines the generic structure of one-dimensional reacting flow models, including state variables, balance equations, and connectors. Within *AnodeChannel*, the gas mixture composition and temperature evolve according to mass and energy balances that account for both convective transport and chemical reactions. The model explicitly includes internal reforming phenomena, such as SR and WGS, which convert methane and carbon monoxide into hydrogen and carbon dioxide, The electrochemical oxidation reactions like HOR, and COR are coupled to the anode channel through the PEN interface. These reactions consume hydrogen and carbon monoxide using oxygen ions transported from the cathode and generate electrons that flow through the external circuit.

4.4.5 Cathode Subsystem Class

The cathodic side is represented by the *CathodeChannel* class, which also derives from *BaseChannel*. While its transport equations are structurally similar to those of the anode, the physical role of the cathode is fundamentally different. In the cathode channel, oxygen (typically diluted with carbon dioxide in high-temperature applications) is transported toward the electrochemical interface, where it undergoes ionization reactions producing oxygen ions O^{2-}. These ions are injected into the electrolyte and migrate toward the anode under the electrochemical potential gradient. The cathode model therefore provides the ionic input required to sustain the anodic oxidation reactions and closes the electrochemical cycle.

4.4.6 PEN Class and Electrochemical Coupling

The *PEN* class represents the Positive Electrolyte Negative assembly and constitutes the core of the electrochemical conversion process. This class couples anodic and cathodic subsystems by enforcing charge conservation and electrochemical equilibrium relations. Within the *PEN* model, the cell voltage is computed as the difference between the reversible Nernst potential and several loss contributions, including activation, ohmic, and concentration losses, as described in [26], and in previous sections. The (4.5) explicitly links temperature and species partial pressures to the open-circuit voltage, providing the fundamental thermodynamic driving force for power generation.

To improve numerical robustness, especially during initialization, the PEN equations make extensive use of the *homotopy*, allowing a smooth transition from simplified linear relations to the full nonlinear electrochemical model.

4.4.7 System-Level Behavior and Model Complexity

When all modules are assembled into the full stack, the resulting SOFC model exhibits a high-dimensional, strongly nonlinear dynamical structure. The overall system order scales linearly with the number of spatial modules, while nonlinearities arise from electrochemical kinetics, temperature-dependent material properties, and reacting flow dynamics. This combination of distributed parameters and nonlinear coupling makes the SOFC model representative of a challenging class of energy system models. At the same time, the clear modular structure and well-defined interfaces provided by [29] make the model particularly suitable for advanced analysis and model order reduction, as discussed in the following chapters.

5 Model Order Reduction of Heat Exchanger-Turbine System

This chapter presents the application of the MOR methodology introduced in Chapter 3 to the heat exchanger-turbine model described in Chapter 4. The focus is on the construction and validation of reduced-order models capable of accurately reproducing the frequency response of the original nonlinear full-order model in operating conditions of practical interest. The chapter is organized into two main parts. First, the reduction of the system in a SISO configuration is addressed, motivated by a control-oriented waste heat recovery (WHR) application. Then, the extension to the MIMO case is discussed, highlighting the additional challenges and design choices involved. This first model is a bench test to set up the correct approach in order to correctly reduce the SOFC model, which is more complex and larger than the heat exchanger-turbine one.

5.1 SISO Reduction: Problem Setup and Objectives

5.1.1 Control-Oriented Motivation

The SISO reduction experiment is motivated by WHR power generation problems, where the heat exchanger-turbine system is fed by the exhaust gases of an industrial process. In this context, the primary control objective is not directly control the electrical power output, which ultimately depends on the available waste heat power,

but rather to guarantee safe and reliable turbine operation by maintaining the turbine inlet temperature (TIT) at a prescribed nominal value. In industrial gas turbines, the TIT is a critical variable that directly affects mechanical stress, component lifetime, and overall system safety. A typical operating value for an industrial application is approximately

$$TIT_{ref} = 510\,°C\ (784\,k),$$

which is adopted in this study as the reference temperature to be regulated.

5.1.2 Selection of Input, Output, and Disturbances

In the SISO configuration considered here, the following variables are selected:

- Controlled input: $w_{cold}(t)$, the mass flow rate on the cold side of the heat exchanger.
- Controlled output: $TIT(t)$ the turbine inlet temperature.
- Disturbance: $whot(t)$, the mass flow rate on the hot side of the heat exchanger.

Physically, $whot$ represents the exhaust flow rate of the upstream industrial process. In a real WHR application, this quantity is not directly controllable and may vary depending on the operating conditions of the primary process. For this study, however, $whot$ is treated as a constant disturbance during each experiment and is used to define different operating points of the system. The cold-side mass flow rate $wcold$ is instead assumed to be actuated by a control valve and constitutes the manipulated variable used to regulate the turbine inlet temperature.

5.1.3 Definition of Operating Points

To assess the ability of the reduced-order model to generalize across a range of operating conditions, three representative steady-state operating points are selected. These points correspond to different load levels of the turbine, expressed as percentages of the nominal electrical power output from the turbine named Pe . The selected operating points are:

- Low load OP2 (30% of electrical power Pe respect to the nominal value):

$$w_{hot} = 45\ \frac{Kg}{s},\quad Pe \approx 6\ MW,$$

- Nominal load OP1 (100% of the electrical power Pe respect to the nominal value):

$$whot = 100\ \frac{Kg}{s},\quad P_e \approx 21\ MW,$$

- Intermediate load OP3 (65% of the electrical power P_e respect to the nominal value)

$$w_{hot} = 75\ \frac{Kg}{s},\quad P_e \approx 13.65\ MW,$$

5.1.4 Computation of Steady-State Conditions

For each value of w_{hot}, the corresponding equilibrium value of w_{cold} that guarantees

$$TIT(t) = TIT_{ref},$$

is computed numerically using a proportional-integral (PI) controller. The PI controller acts on the error

$$e(t) = TIT_{ref} - TIT(t),$$

and generates the cold-side mass flow rate command $w_{cold}(t)$. The integral action is characterized by an integral time constant of $T_i = 900\ s$, which ensures slow and smooth convergence to steady-state without inducing large transients in the thermal dynamics. For each operating condition, a simulation of duration $200000\ s$ is performed for OP2 and OP3, $300000\ s$ for OP1, allowing the system to fully settle to steady-state. Is important to note that OpenModelica uses a variable step stiff solver; in this way the time required to simulate the transient tail is small. The resulting equilibrium values are then extracted and used as operating points for subsequent analysis.

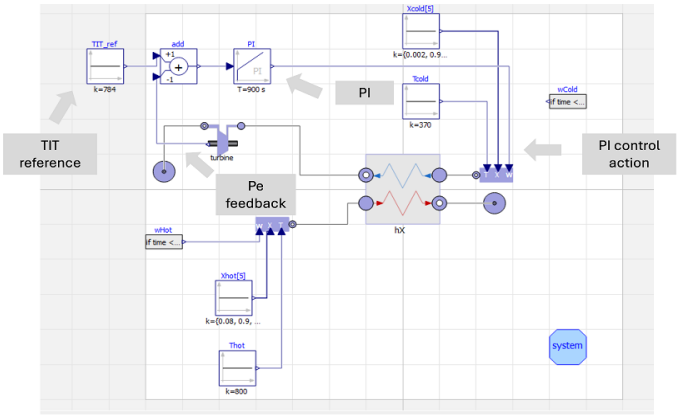


Figure 5.1: Heat exchanger-turbine model connected with PI

Using the PI controller the following results are obtained:

- Low load yielding an equilibrium value $w_{cold}^{*} = 45\ \frac{Kg}{s}$
  - $TIT = 784\ K,\ P_e = 6\ MW;$
- Nominal load yielding an equilibrium value $w_{cold}^{*} = 100\ \frac{Kg}{s}$
  - $TIT = 784\ K,\ P_e = 21\ MW;$
- Intermediate load yielding an equilibrium value $w_{cold}^{*} = 75\ \frac{Kg}{s}$
  - $TIT = 784\ K,\ P_e = 13.65\ MW;$

	OP2	OP1	OP3
TIT	784 K	784 K	784 K
w_{cold}^{*}	45 Kg/s	100 Kg/s	75 Kg/s
Pe	6 MW	21 MW	13.65 MW

Table 5.1: Equilibrium values of the three OPs

5.1.5 Dataset Generation for Model Reduction

Once the equilibrium points are identified, datasets for model reduction are generated around the OP1 and OP2 conditions. To this end, a signal generator of prescribed order, equal to the desired order of the reduced model, is connected to the cold side mass flow rate input w_{cold}. The signal generator starts from each of the previously obtained equilibria and produces a sinusoidal excitation centered around the equilibrium value w_{cold}^{*}, with a relative amplitude of $\pm 3.5\%$:

$$w_{cold}(t) = w_{cold}^{*}(1 + 0.035\sin(\omega t)). \quad (5.1)$$

This choice ensures that the system is sufficiently excited to capture its dominant dynamics; the trajectories remain close to the equilibrium point, and the validity of

local linear approximations and moment matching arguments is preserved. For each simulation, the following quantities are recorded:

- The internal states of the signal generator;
- The excitation input $w_{cold}(t)$;
- The output of the full nonlinear model $TIT(t)$.

These datasets have the following structure

$$D = \{w_{cold}(t_i), TIT(t_i), x_1(t_i), \ldots x_v(t_i)\}, \quad (5.2)$$

where:

- $w_{cold}(t_i)$ is the excitation input at simulation time t_i;
- $TIT(t_i)$ is the output of the full nonlinear model at simulation time t_i;
- $x_j(t_i)$ is the internal j -th internal state of the signal generator at simulation time t_i, where $j = 1, \ldots, v$ and v is the generator order.

These datasets constitute the input of the data-driven reduction algorithm, which will be described in detail in the following section.

5.1.6 Validation Strategy

The accuracy and performance of the reduce-order model is assessed in the frequency domain. Specifically, the nonlinear full-order model and the nonlinear reduced-order model are linearized around each OPs. The resulting linear models are then compared by analyzing their Bode diagrams. The comparison is performed:

- At the operating points used for data generation (OP1, OP2);
- At OP3, which is not used during the reduction procedure.

This validation strategy allows one to assess both interpolation accuracy and generalization capabilities of the reduced-order model.

5.2 SISO Reduction: Problem Setup

This section presents the data-driven nonlinear MOR algorithm developed and employed for the SISO heat exchanger-turbine model. The algorithm is designed to construct a low-order nonlinear model that accurately reproduces the frequency-domain behavior of the full-order system around multiple operating points, while retaining a compact structure suitable for control-oriented analysis. The methodology builds directly upon the nonlinear moment matching framework introduced in Chapter 3 and combines interpolation-based model reduction, data-driven identification, and numerical optimization. The complete MATLAB implementation is reported in Appendix D. In this section, the theoretical concepts underlying the

reduction algorithm used in this thesis are introduced, while the following section comprehensively explains the various sections of the MATLAB code that implement the SISO reduction algorithm. Note that the proposed model reduction method, which will later be validated in Section 5, has two main steps. The first applies moment matching theory to compute the moments μ^*; the second tries to further improve the quality of the MOR by minimizing the distance between the Bode plots of full and reduced models over a certain frequency range (not only the matching frequencies), by adapting the parameters α and β of the Rayleigh matrix defined in Section 5.2.1.

5.2.1 Signal Generator

One of the fundamental components of the reduction process, and of the ROM structure, is the signal generator. In this thesis a linear signal generator of the form

$$\dot{w}(t) = Sw(t), \quad (5.3)$$
$$u(t) = Lw(t).$$

The generator output used to excite the physical model is obtained by applying an offset and a relative amplitude scaling to the internal output:

$$y_{tot}(t) = y_0\big(1 + amp_{rel}u(t)\big),$$

where:

- y_0 is a constant offset defining the nominal operating level, and in this context is referred to the equilibrium value of the considered OP;
- amp_{rel} is a dimensionless parameter defining the relative oscillation amplitude around the offset, in this context $amp_{rel} = \pm 0.035$, in order to assure an oscillation of ± 3.5 % around y_0;
- $u(t)$ is the output given by the autonomous system (5.3).

This formulation allows the generation of smooth, bounded, and persistently exciting signals around a prescribed OP. The generator is characterized by two user-selectable parameters, the matrices (S, L) through which it is possible to characterize the dynamical structure of the generator. In particular, the matrices are considered degrees of freedom in the reduction process.

- Matrix S is the dynamical parameter; it defines the internal dynamics of the signal generator. In the configuration used in this thesis, S has purely imaginary eigenvalues, resulting in oscillatory behavior of the state vector. The eigenvalues determine the frequency where moment matching is done, and

determine the order of the reduced model. In this configuration, each match corresponds to two purely imaginary eigenvalues of opposite sign that introduces an order of two into the reduced model and matches the system at ω ;

- Matrix L is the output parameter, which selects a linear combination of the internal states to build the relative output signal. The parameters y_0 and amp_{rel} control the mean value and amplitude of the generated signal;
- Initial conditions values of the internal states are explicitly specified to ensure deterministic and reproducible signal generation.

The matrix $S \in \mathbb{R}^{v \times v}$is the state matrix of the generator and is constructed such that is spectrum satisfies

$$\sigma(S) = \{\omega_1, \omega_2, \ldots, \omega_v\},$$

where $\sigma(\cdot)$ denotes the spectrum of the matrix. In the most general case, S is chosen in canonical form and may be written as a block-diagonal matrix

$$S = \begin{bmatrix} S_1 & & \\ & \ddots & \\ & & S_r \end{bmatrix},$$

where each block S_i corresponds to a complex-conjugate pair $\omega_i \in \mathbb{C}$, yielding a 2×2 block of the form

$$S_i = \begin{bmatrix} \Re(\omega_i) & \Im(\omega_i) \\ -\Im(\omega_i) & \Re(\omega_i) \end{bmatrix}.$$

In the case of this thesis, the complex number

$$\omega_i = \Re(\omega_i) + \mathrm{j}\,\Im(\omega_i),$$

has the real part equal to zero and the imaginary part equal to the desired interpolation frequency. This choice defines that the matrix S_i has values equal to zero on the main diagonal, and values equal to $\pm\omega_i$ on the secondary diagonal. In the

reduction algorithm the overall matrix S employed is obtained by adding to the S matrix that contains the generator dynamics, named S_0, a damping coefficient obtained through a Rayleigh-type damping matrix. In this reduced-order modeling framework, the Rayleigh damping matrix is introduced as a structured way to model linear dissipation in the system. It is defined as

$$D_{Rayleigh} = \alpha I + \beta S_0, \quad (5.4)$$

where:

- α is a scalar coefficient representing uniform damping; it is uniform at all the system's modes and is dominant at low frequencies. It reduces the energy of the system in a homogeneous way;
- β is a proportional damping coefficient; it is proportional to the system's dynamic, and it dominates at high frequencies. The introduction of this damping parameter causes a drawback because β is proportional to the signal generator dynamics, the frequencies at which the matches occur are slightly different from the values chosen by the user;
- I is the identity matrix;
- S_0 is the nominal reduced dynamics imposed by the signal generator and determined by the interpolation points chosen by the user.

Rather than introducing an arbitrary damping matrix to decrease the resonance peaks, this formulation assumes that dissipation is partly uniform across all reduced states and partly proportional to the system's dynamics encoded in S_0. This makes damping physically structured and consistent with the ROM. In the reducing algorithm, the Rayleigh matrix is subtracted directly from the nominal dynamics

$$S_{damped} = S_0 - D_{Rayleigh}, \quad (5.5)$$

in this way the S matrix needed to build a part of the linear dynamics of the ROM becomes S_{damped}, and so

$$\dot{x_r}(t) = \left(S_0 - D_{Rayleigh}\right)x_r(t),$$

where $x_r(t)$ is the reduced state, and substituting the definition of $D_{Rayleigh}$, this can be written as

$$\dot{x_r}(t) = (1-\beta)S_0 x_r(t) - \alpha x_r(t).$$

This shows clearly how the two parameters act in the model:

- The parameter α introduces uniform dissipation and shifts all eigenvalues of the ROM to the left in the complex plane. It primarily affects the overall damping level and reduces the height of resonance peaks in the frequency response. It influences the overall quality of the ROM, but does not influence the interpolation frequencies;
- The parameter β scales the structural dynamics contained in S_0. As a result, it modifies both the effective stiffness behavior and the frequency-dependent damping characteristics. This parameter influences the shape and distribution of the resonance peaks across frequencies.

The purpose of introducing this damping is closely related to the frequency-domain optimization carried out by the reduction algorithm as a last step. Thanks to this kind of damping, the optimization procedure has two more parameters α, β available to improve the accuracy of the ROM frequency response.

The matrix $L \in \mathbb{R}^{1 \times v}$ determines which modes of the signal generator are effectively excited, the relative weighting among different interpolation points, and the amplitude distribution of the generated signal. The matrix L is, in general, a degree of freedom of the method. In this thesis, since sinusoidal signal generator have always been use, it structure is fixed and is

$$L = \left[l_1, l_2, \ldots, l_{\frac{v}{2}}\right],$$

where v is the reduced order (the dimension of S), and $l_i = [1,0]$. Consequently, by introducing the signal generator, two of the degrees of freedom that the algorithm makes available to the user are also defined with matrices (S, L). Provide that the pair is observable; the resulting signal generator is able to excite all modes associated with the selected interpolation points.

5.2.2 ROM structure

The second fundamental component of the reduction algorithm is the structure of the ROM implemented, the ROM is nonlinear and inspired by (3.14). The goal is to make the ROM preserve the input-output behavior of a full model of the form (3.1). In the context of this thesis the ROM is realized starting from a linear generator (5.3), consequently the nonlinearity in the states is introduced through the degree of freedom offered by the function $g(\cdot)$ of (3.14) introduced in Chapter 3, which in this

case is intentionally designed to be nonlinear. Further nonlinearities can be introduced into the ROM by the output equation, in case nonlinear kernel functions are used for moment matching in (3.17); otherwise, the output equation is linear and the nonlinear behavior of the system must be entirely captured by function $g(\cdot)$.Consequently, the ROM resulting from the reduction algorithm has the following structure:

$$\dot{x_r} = S_{damped}x_r - g(x_r,u)L^TLx_r + g(x_r,u)L^Tu, \quad (5.6)$$
$$y_r = \mu^*(x_r).$$

In this case, S_{damped} is the matrix (5.5) containing Rayleigh-like damping (5.4), $g(\cdot)$ is the function that introduced nonlinearities in state dynamics, and $\pi(\cdot)$ is the moment function (3.12). The ROM has order v equal to the dimension of S_{damped}. Function (3.12) is estimated in a data-driven way by solving an optimization problem (3.16), which, thanks to the Representer Theorem, takes the finite-dimensional form (3.17). Problem (3.17) is solved by obtaining the α weights by solving the matrix equation (3.18). The weights α introduces the nonlinearities into the estimated moment function (3.12), so they are of paramount importance. Another fundamental component of ROM is the function $g(\cdot)$, which is the third degree of freedom offered by the algorithm. From this consideration, it is possible to define two approaches to derive $g(\cdot)$:

1. Choosing $g(\cdot)$ in a trivial way, equal to 1 in the scalar case, or to the identity matrix in the multidimensional case, this choice generates in the case considered in this thesis a purely linear dynamics of the states, consequently all the nonlinearities, and the ability of the ROM to reproduce the input-output behavior of the full model is entrusted to the moment function, this can be defined as a black-box approach.
2. Choosing $g(\cdot)$ in a non-trivial way that can introduce nonlinearity in the dynamics of the states, contributing to the overall ROM reduction capabilities.

In the implemented reduction algorithm, approach 2 is chosen; that is, a particular structure of the function $g(\cdot)$ is defined. Two types of $g(\cdot)$ function are defined, the first one that does not take into account the a priori knowledge of the models to be reduced, the second one that tries to take it into account, in particular by introducing a state input product dynamics typical of the energy systems that are treated in this thesis, in a certain sense this second approach can be defined as a grey box.

1. Function $g(\cdot)$ with state input product:

$$g(x_r,u) = diag\left(p_1 + p_2\tanh(p_3x_r) + p_4x_r^{\circ 2} + p_5x_r\sum_j u_j\right), \quad (5.7)$$

where:

- $x_r \in \mathbb{R}^v$ is the reduced state;
- $u \in \mathbb{R}^m$ is the input;
- $x_r^{\circ 2}$ is the element-wise squaring;
- $diag(\cdot)$ is the diagonal matrix;
- $p_1, p_2, p_3 . p_4, p_5$ are parameters to be optimized;
- $x_r \sum_j u_j$ is the implementation of the state input product.

2. Function $g(\cdot)$ without state input product:

$$g(x_r, u) = diag(p_1 + p_2 \tanh(p_3 x_r) + p_4 x_r^{\circ 2}), \quad (5.8)$$

The $g(\cdot)$ function was chosen in this form to account for the dynamics of state saturation and state variation related to variations in the system's operating points; this is implemented by the hyperbolic tangent. A square element-wise function was then introduced to account for quadratic nonlinearities of the states, and finally, the state input product, typical of the energy systems considered in this thesis. To conclude the theoretical foundations of the reduction implemented in this thesis.

5.2.3 Optimization problem

To conclude the theoretical foundations of the reduction implemented in this thesis, it is important to note that the reduction algorithm optimizes the p_i parameters of the $g(\cdot)$ function, the α and β parameters of the matrix (5.5), and the equilibrium states of the reduced model. The optimization problem aims to minimize the distance between the magnitude and phase of the full model frequency response with respect to the ROM frequency response. Overall, the degree of freedom provided by the reduction algorithm are three:

1. Matrix S_0, with which to define the matching points.
2. Matrix L with which to define the type of signal generator and maintain fixed in this thesis.
3. Function $g(\cdot)$.

Once these three degrees of freedom are imposed by the user, the algorithm constructs the ROM by estimating the linear or nonlinear moment function, finally optimizing the reduced model so that it minimizes the frequency response with respect to the full model. This section describes in detail the optimization procedure that is performed on the MOR after moment matching. Mathematically the problem $\min_p J(p)$ is solved,

where $J(p)$ is a cost function, and p is the parameters vector $p = \left[\alpha, \beta, p_i, x_{r_j}^*\right]^T$, with $i = 1, \ldots, 5$, and $j = 1, \ldots, v$, x_r^* is the reduced equilibrium state, and v is the ROM order; the total dimension of the optimization problem is $2 + 5 + 3v$. for each vector p the optimization algorithm:

1. Reconstruct the Rayleigh matrix (5.4).
2. Construct the dynamic state, and output equation of (5.6), here named $f_r(x_r, u)$, and $g_r(x_r)$ for brevity.
3. Linearize the ROM (5.6) around the k-th OP of interest with Jacobians $A_k = \frac{\partial f_r}{\partial x_r}$, $B_k = \frac{\partial f_r}{\partial u}$, $C_k = \frac{\partial g_r}{\partial x_r}$.
4. Construct the linearized models around the OPs.
5. Compute the full-order model and ROM frequency response functions $G_{FOM,k}(j\omega)$, $G_{ROM,k}(j\omega)$.
6. Compare the two frequency response functions.

For each OPs, $k = 1,2,3$, and for each frequency ω , magnitude and phase errors are computed

- Magnitude error

$$e_{k,i}^{mag}(p) = 20\log_{10}|G_{ROM,k}(j\omega_i; p)| - 20\log_{10}|G_{FOM,k}(j\omega_i)|$$

- Phase error

$$e_{k,i}^{ph}(p) = \left(\angle G_{ROM,k}(j\omega_i, p) - \angle G_{FOM,k}(j\omega_i)\right)$$

For each OP the norm on the errors are computed

$$E_k^{mag}(p) = \left(\sum_{i=1}^{N_w} (e_{k,i}^{mag}(p))^2\right)^{1/2}$$

$$E_k^{ph}(p) = \left(\sum_{i=1}^{N_w} (e_{k,i}^{ph}(p))^2\right)^{1/2}$$

The complete cost function became

$$J(p) = \frac{1}{\sqrt{N_w}} \sum_{k=1}^{3} E_k^{mag}(p) + \frac{0.01}{\sqrt{N_w}} \sum_{k=1}^{3} E_k^{ph}(p) + 10^{-3} \sum_{i=1}^{4} |p_\delta^{(i)}|$$

Where the first term refers to the norm of magnitude error, the second term refers to the norm of phase error, and the last is a regularization term, and N_ω is the number of frequencies considered in the optimization. Division by $\sqrt{N_\omega}$ is useful to make the cost function independent with respect to the number of frequencies considered.

5.3 SISO Reduction: Model Order Reduction Algorithm

5.3.1 Section 0: Initialization of Damping Parameters

The algorithm begins with the initialization of two scalar damping coefficients α , and β :

```
alpha_damp=0.025;
beta_damp=0.75;
```

These parameters are introduced to construct a Rayleigh-type damping matrix (5.4) for the ROM. Their roles are twofold. First, they allow the designer to explicitly shape the dissipative properties of the reduced dynamics, which is particularly important when dealing with interpolation-based reduced models whose stability is not automatically guaranteed, moreover, to address oscillations issues. Second, these coefficients provide additional degrees of freedom during the optimization phase, enabling a better match between the frequency responses of the reduced and full-order models across a wide range of frequencies. At this stage, the values of *alpha_damp* and *beta_damp* are only initial guesses. Their optimal values are identified later through a frequency-domain optimization procedure.

5.3.2 Section 1: Linearization of the Full-Order Model

To assess the accuracy of the reduced model in the frequency domain, linearized representations of the full nonlinear system are required. Three full-order LTI models are constructed by linearizing the heat exchanger-turbine model around the operating points OP1, OP2, and OP3 identified in Section 5.1.

```
 sys=ss(A_full_1,B_full_1,C_full_1,D_full_1);
sys1= ss(A_full_2,B_full_2,C_full_2,D_full_2);
sys2= ss(A_full_3,B_full_3,C_full_3,D_full_3);
```

these linearized models are not used during the reduction procedure itself; they serve as high-fidelity references against which the reduced models are validated in the frequency domain. The direct feedthrough matrices are explicitly set to zero:

```
sys.D=0;
sys1.D=0;
sys2.D=0;
```

this choice is consistent with the physical structure of the system, where no instantaneous dependence between the manipulated mass flow rate and the turbine inlet temperature is present.

5.3.3 Section 2: Dataset Preprocessing and Steady-State Extraction

The datasets used for data-driven moment estimation are obtained from nonlinear simulations of the full-order model excited by a signal generator (5.3). These simulations typically include long transient phases before reaching a quasi-steady regime. Since moment matching aims to capture the local dynamical behavior around equilibria, it is essential to remove transient effects that do not reflect the steady-state dynamics. This motivates a preprocessing stage where:

- An initial transient window is discarded;
- The remaining values are subsampled to reduce redundancy;
- Data are reshaped into matrices compatible with the reduction algorithm.

A representative excerpt is:

```
idx=find(t_out>tSS,'first');
omega_1=omega_full_1(:,idx:fs:end);
y_1=y_full_1(idx:fs:end)/scale;
u_1=u_full_1(idx:fs:end);
```

This step significantly improves numerical robustness and ensures that the estimated moments reflect the steady-state input-output behavior of the full-order system.

5.3.4 Section 3: Nonlinear Dictionary and Moment Estimation

A key element of the data-driven nonlinear moment matching framework is the introduction of a dictionary of nonlinear functions:

```
dictionaryOfFunc={
@(z) z(1);
@(z) 1;
```

```
};
```

The dictionary defines the functional space used to approximate the output map of the reduced model. It serves two closely related purposes:

1. It determines how the generator states are mapped into a feature vector.
2. It implicitly defines the class of nonlinearities that can be captured in the estimated moments.

In this SISO case, a deliberately simple dictionary is employed, consisting of a linear term and a constant offset, this ensure the estimation of a linear moment function. The dictionary is evaluated on the generator states extracted from the datasets, and the corresponding weights, necessary to estimate moments, are computed using regularized least-square procedures (3.18) defined in Chapter 3.

```
weights = (nlValFromZeta_all' * nlValFromZeta_all +
lambda*eye(length(dictionaryOfFunc))) \ (nlValFromZeta_all' *
yNL_all);
```

where *nlValFromZeta_all* defines the contribution to the nonlinearity from the generator state, and *weights* correspond to α vector of (3.18).

```
for i=1:windowSize
NlValFromZeta_all(i,:)=modred.internal.computeNonlinearityFrom
State(real(omegaNL_all(i,:)), dictionaryOfFunc);
End
```

The function *modred.internal.computeNonlinarityFromState()* is provided from the Imperial College London moment matching MATLAB toolbox and is clear to see that to compute the nonlinear contribution of the state the function uses *dictionaryOfFunc*. From a theoretical standpoint, it is correct to state that the dictionary not only constructs the feature vector but also introduces nonlinear structure into the estimated moments. In a data-driven setting, moments are inferred from measured trajectories, and the dictionary directly determines how nonlinear dependencies are embedded in the reduced representation.

5.3.5 Section 4: Nominal Reduced-Order Model Construction

The reduced-order dynamics are driven by a linear signal generator (5.3) characterized by matrices S_0 and L

```
S0=computeSFromInterpolationPoints([...]);
L=ones(1,nu);
```

The matrix S_0 defined in Section 5.2.1 is constructed from a prescribed set of interpolation points and has a block-diagonal Jordan structure. The dimension *nu* of S_0 determines the order of the reduced model and the columns of matrix L .The mathematical structure of S_0 and L, as well as their relationship with the chosen interpolation points, are formally defined and discussed in Section 5.2.1, which introduces the signal generator concept used throughout this thesis. The nonlinear dynamic of the reduced model is implemented through four main mathematical entities:

1. Function *f_rom*, defined as $s(x_r) - g(x_r)l(x_r)$ in (3.14), represents the nonlinear state dynamics and is defined such that a damping effect is embedded into the function $s(\cdot)$, in the ROM (5.6) corresponds to the state equation.

   ```
   D_rayleigh = alpha_damp * eye(nu) + beta_damp * S0;
   ```

   This line of code defines a Rayleigh-type damping matrix (5.4), and the damping parameters defined into section 0 are employed

   ```
   S_damped = S0 - D_rayleigh;
   ```

   Definition of the damping matrix (5.5) to embed into *f_rom*

   ```
   f_rom = @(xi,u) ( S_damped*xi ) -
   delta_fun_opt(xi)*(L'*(L*xi)) + delta_fun_opt(xi)*(L'*u);
   ```

   Definition of *f_rom* is a function of the reduced state x, and input u.

2. Matrix L, represents the contribution of the output equation of the signal generator (5.3).

3. Function *build_delta*, that corresponds to function $g(\cdot)$ defined in Section 5.2.2, it represents the main degree of freedom of the nonlinear reduced model. It is defined in two ways:
   a. Function $g(\cdot)$ with state-input product embedded in

```
build_delta = @(p) ( @(xi,u) diag( ...
                p(1) + ...
      p(2) .* tanh( p(3) .* real(xi(:)) ) + ...
           p(4) .* (real(xi(:)).^2) + ...
p(5) .* ( real(xi(:)) .* sum(real(u(:))) ) ...
                      ) );
```

The mathematical structure of $g(\cdot)$ is (5.7).

b. function $g(\cdot)$ without state-input product embedded in

```
build_delta = @(p) ( @(xi) diag( p(1) + p(2) .* tanh( p(3) .*
     real(xi(:)) ) + p(4) .*       (real(xi(:)).^2) ) );
```

The mathematical structure of $g(\cdot)$ is (5.8).

In the linear moment matching framework, the matrix G plays the role of a free parameter used to impose additional properties on the reduced dynamics. In the nonlinear framework developed in Chapter 3, the function $g(\cdot)$ plays the exact same conceptual role: it generalizes the linear matrix G to a nonlinear, state, and input dependent operator. The only difference is that in this algorithm $g(\cdot)$ is named $\delta(\cdot)$.

The specific structure of $g(\cdot)$ is not dictated by the signal generator. Instead, it is chosen based on a combination of numerical consideration and physical insight defined in Section 5.2.2.

There is no universal or mechanical procedure to define $g(\cdot)$. Rather, its structure can be informed by prior knowledge of the system physics, as done here. It is important to say that *delta_fun_opt* in *f_rom*, is the function $g(\cdot)$ with the optimized parameters p_1, p_2, p_3, p_4, p_5.

4. Moment function computed via (3.17)

```
@(xi) (weights' * phi_col(xi));
```

Using the weights estimated with regression (3.18) and *dictionaryOfFunc* the moment function (3.12) is defined as a reduced states function. The data-driven logic is embedded here; moments are estimated in a data-driven way

because the weights are estimated in a data-driven way using the dictionary , the values of outputs *yNL_all,* and states from datasets.
By the end overall reduced order is defined using the function above

```
f_rom = @(xi,u) ( S0*xi - D_rayleigh*xi ) -
delta_fun_opt(xi)*(L'*(L*xi)) + delta_fun_opt(xi)*(L'*u);

g_rom = @(xi) reducedModel.output(xi);

reducedModel.output = @(xi) (weights' * phi_col(xi));
```

And the mathematical structure of the reduced model of order v is (5.6). The structure of the reduce-order model used in this thesis is slightly different from the form (3.14). It is a model built from a linear signal generator matrices (S, L), the nonlinearity is embedded in the function $g(\cdot)$. Furthermore, for the heat exchanger-turbine case, the moment is estimated as a linear function.

5.3.6 Section 5: Equilibrium Points of the Reduced Model

Equilibrium inputs for the reduced model are selected as the mean values of the input datasets:

```
u_eq1=mean(u_1);
u_eq2=mean(u_2);
```

This choice is physically meaningful and numerically robust. Numerical experiments indicate that the reduced model naturally preserves equilibrium input values of the full-order system. In contrast, no analogous property holds for the reduced states. Consequently, equilibrium states of the reduced model are treated as free variables and later optimized to improve frequency-domain matching as said in Section 5.2.2, and Section 5.2.3.

5.3.7 Section 6-8: Optimization of Damping, Nonlinearity, and Equilibria

The parameters defining the function $g(\cdot)$, the damping coefficients, and the reduced-state equilibria are jointly optimized by minimizing a frequency-domain cost function $J(p)$:

```
obj = (norm(mag_err1(:)) + norm(mag_err2(:)) +
norm(mag_err3(:))) / sqrt(Nw) + 0.01 * (norm(ph_err1(:)) +
 norm(ph_err2(:)) + norm(ph_err3(:))) / sqrt(Nw) + 1e-3 *
                  sum(abs(p_delta(:)));
```

where *mag_err1*, *mag_err2*, *mag_err3* are magnitude errors related to Bode plots of three operating points of interest (OP1, OP2, OP3), *ph_err1*, *ph_err2*, *ph_err3* are phase errors, and *norm()* is a MATLAB function to compute the norm. The objective function *obj* penalizes discrepancies between the Bode magnitude and phase of the reduced linearized models and those of the full-order linearized models at all operating points. The optimization problem is solved using a derivative-free algorithm, and all related routines are detailed in Appendix D.

5.3.8 Section 9-11: Linearization and Frequency-Domain Validation

After optimization, the reduced nonlinear model is reconstructed and linearized numerically around each operating point using complex-step Jacobian method. This approach provides high-accuracy numerical derivatives without the drawbacks of finite differences. The resulting reduced linear models are compared to the full-order models via Bode plots, enabling a direct and rigorous frequency-domain validation.

5.4 SISO Reduction: Numerical Experiments and Results

This section presents the numerical experiments and validation results obtained for the SISO heat exchanger-turbine system. The goal is to assess the effectiveness of the proposed data-drive nonlinear MOR framework and to analyze how specific modeling choices influence the quality of the reduced-order models. Two reduction strategies are investigated and compared:

1. Nonlinear reduced models without state-input coupling, with a function g depending only on the reduced state.
2. Nonlinear reduced models with state-input coupling, with a function g including the product between reduced states and input.

This distinction reflects the actual workflow followed during the research activity: the initial attempts focused on purely state-dependent nonlinear corrections, while subsequent refinements introduced state-input interaction terms motivated by physical insight into the heat exchanger, and in general thermo-fluid systems.

5.4.1 Experiment Setup

The reduction experiments are conducted on the SISO configuration of the heat exchanger-turbine model introduced in Section 5.1 and, more generally, in Chapter 4. Three operating points are considered to evaluate the performance and generalization capability of the reduced models, particularly the intermediate point OP3 is used to evaluate the generalization capability of the reduced model. The reason why the generalization capability is evaluated using the intermediate point is that once trained on the two extreme operating conditions OP1, and OP2, the model should also be able to adequately represent all intermediate conditions. This is why the reduction is performed using datasets from only 30% and 100% load conditions, and performance is then evaluated across all three conditions. This way, both the model's fitting and generalization capabilities are assessed. The values associated with the three OPs are defined in Section 5.1.4.

5.4.2 Case 1: Reduced Models without State-Input Product

5.4.2.1 Datasets Generation via Signal Generator

In the first set of experiments, the nonlinear function g depends solely on the reduced state. The full-order nonlinear model is excited by interconnecting it with a signal generator, treated in Section 5.2.1, and in Appendix C, whose output modulates the cold-side mass flow rate around its equilibrium value with the expression (5.1). The amplitude is limited to ± 3.5 % of the equilibrium value to avoid large deviations from the operating point and equilibrium. For each of the two extreme operating points, a simulation is performed, and a dataset (5.2) is created. The only thing that changes from one simulation to the other is the equilibrium value w_{cold}^{*} around the system is excited by the signal generator.

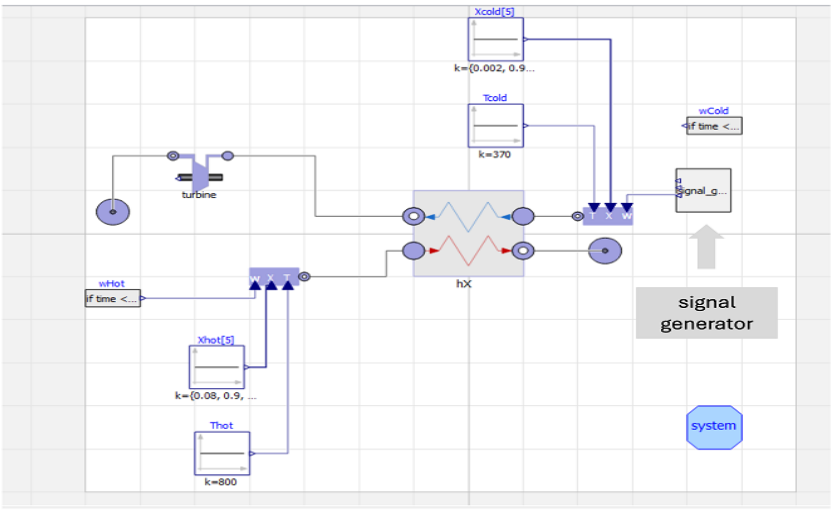


Figure 5.2 Interconnected model adopted for datasets generation

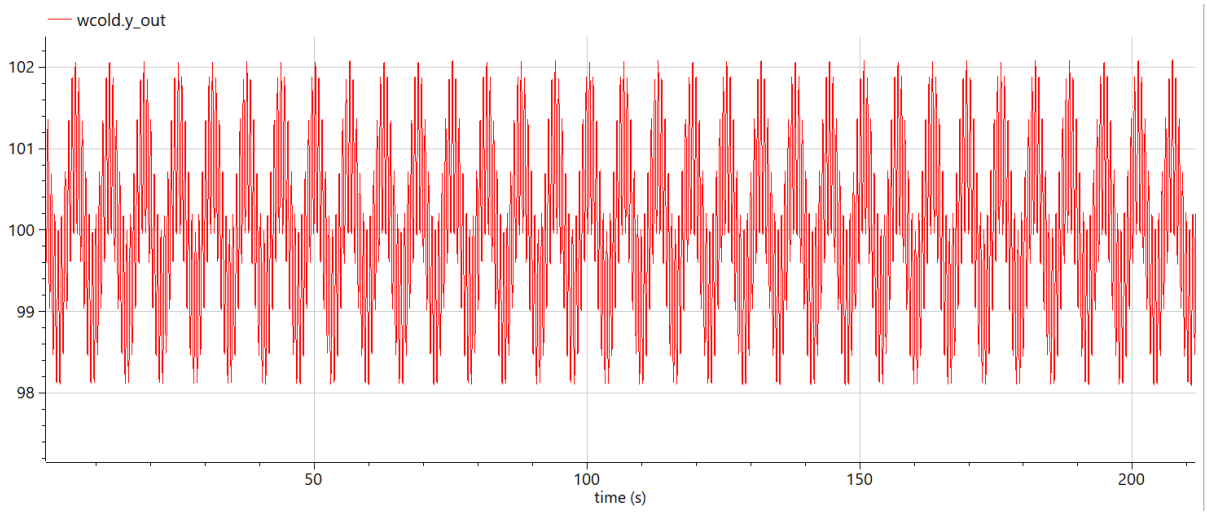


Figure 5.3: Signal generator w_{cold} (Kg/s) of order $v = 4$ around OP1

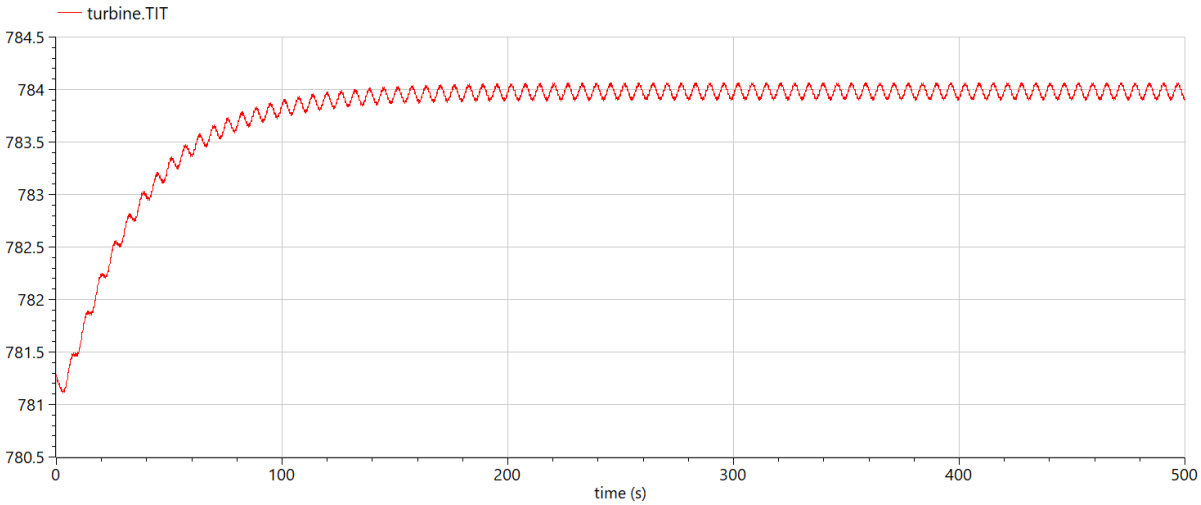


Figure 5.4: TIT (K) response to signal generator of order $v = 4$ around OP1

5.4.2.2 Execution of the Reducing Algorithm

For each dataset, the reduction algorithm described in Section 5.3 is executed. The dictionary of nonlinear function includes only two terms, one linear, and one constant. In that case, the function g does not contain state input coupling. These reduced models are constructed, corresponding to different reduced orders v starting from a full model of order $n = 80$. Three different models are realized:

- Order $v = 4$ and interpolation points in 0.1 and 10 rad/s;
- Order $v = 8$ and interpolation points in 0.1, 0.01, and 1 rad/s;
- Order $v = 12$ and interpolation points in 0.1, 0.01, and 1 rad/s.

The choice of interpolation frequency is made knowing that this system heat exchanger-turbine has main dynamics around 0.1 rad/s, so with the the frequency choice made above the fundamental dynamics of interest is covered. Outside of this band , in particular at higher frequencies, the dynamics is not relevant for control, given the typical control bandwidths associated to these kind of systems. For each case, the reduced nonlinear model is linearized around three operating points and compared against the corresponding linearized full-order model via Bode plots.

5.4.2.3 Frequency-Domain Validation

In this subsection, the results provided by the reduction algorithm are analyzed, the comparison between the Bode plots of the full-order model and the Bode plots of the reduced one around the OPs in the three cases shown above. FOM means full-order model, and ROM means reduced-order model.

1. Reduced model of order $v = 4$

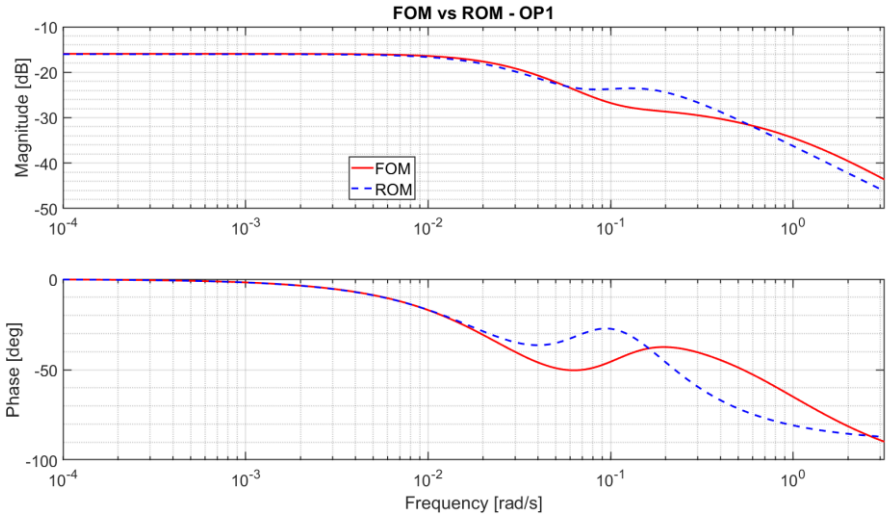


Figure 5.5: Bode comparison between FOM and ROM at OP1

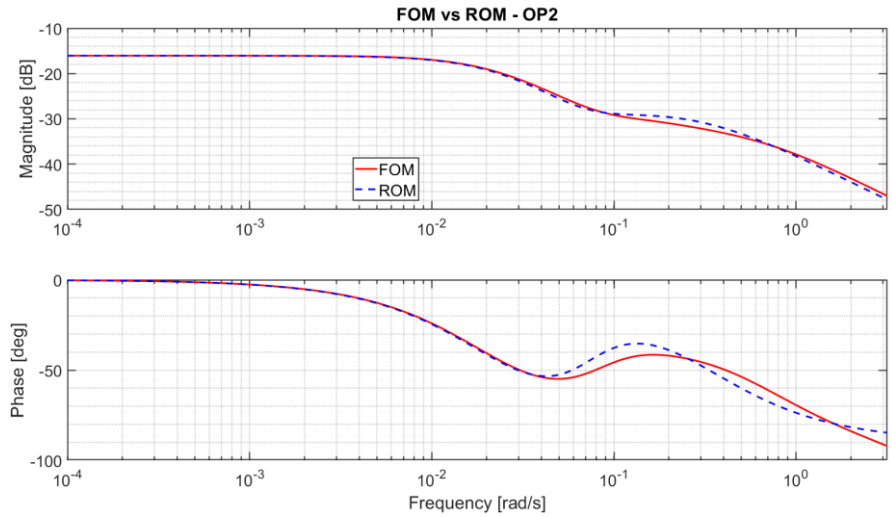


Figure 5.6: Bode comparison between FOM and ROM at OP2

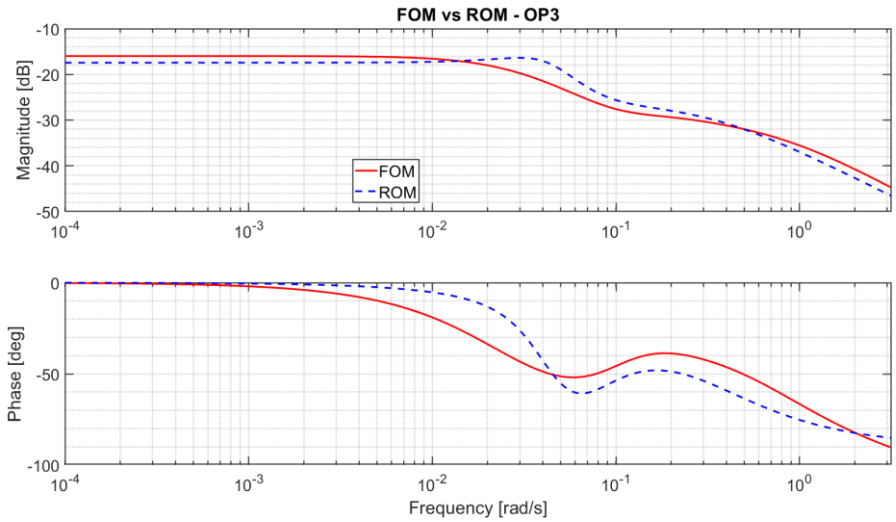


Figure 5.7: Bode comparison between FOM and ROM at OP3

2. Reduced model of order $v = 8$

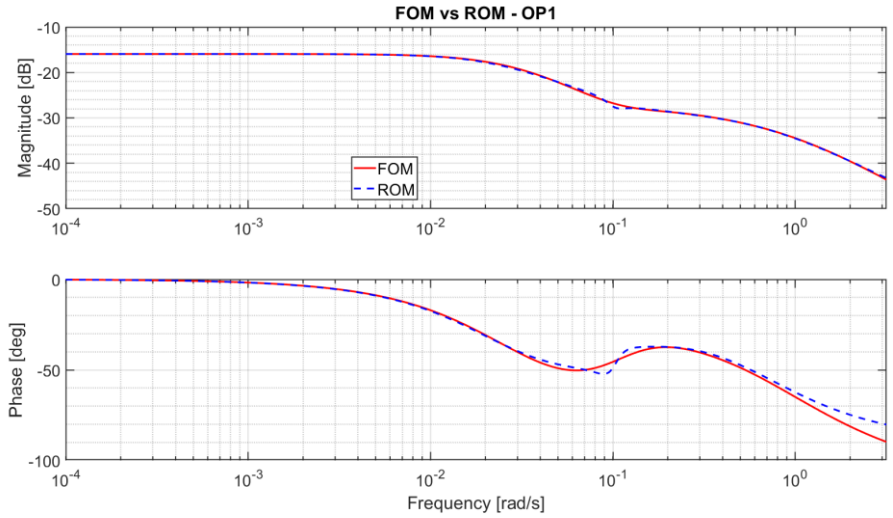


Figure 5.8: Bode comparison between FOM and ROM at OP1

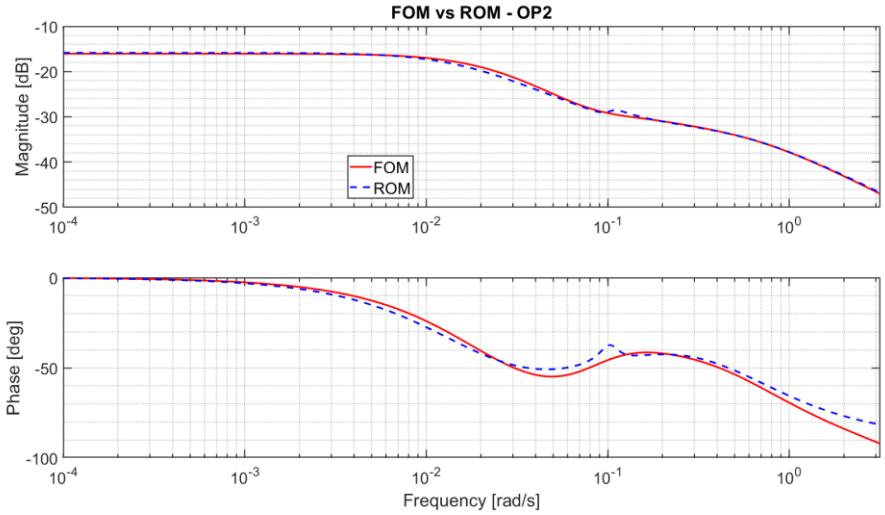


Figure 5.9: Bode comparison between FOM and ROM at OP2

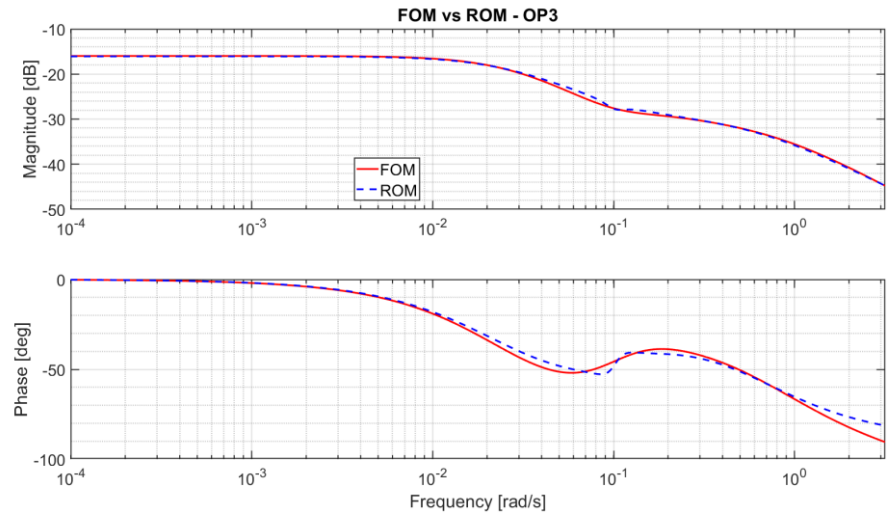


Figure 5.10: Bode comparison between FOM and ROM at OP3

3. Reduced model of order

$$v = 12$$

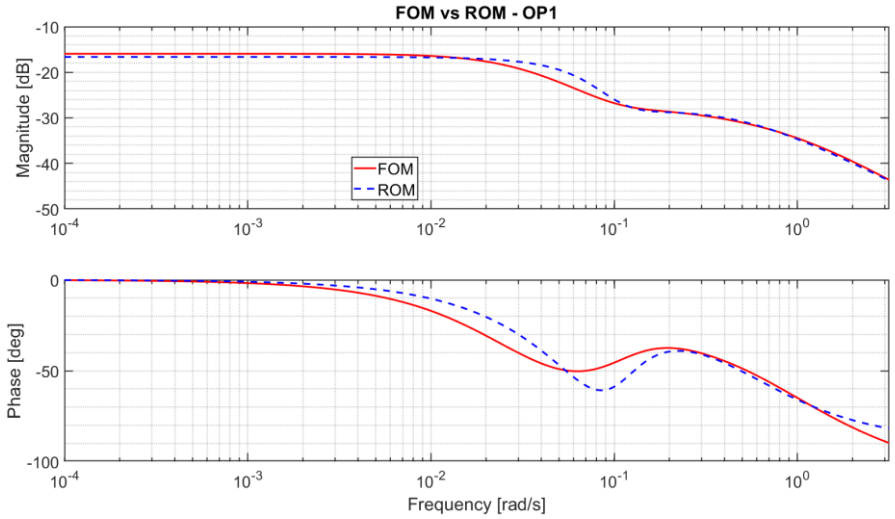


Figure 5.11: Bode comparison between FOM and ROM at OP1

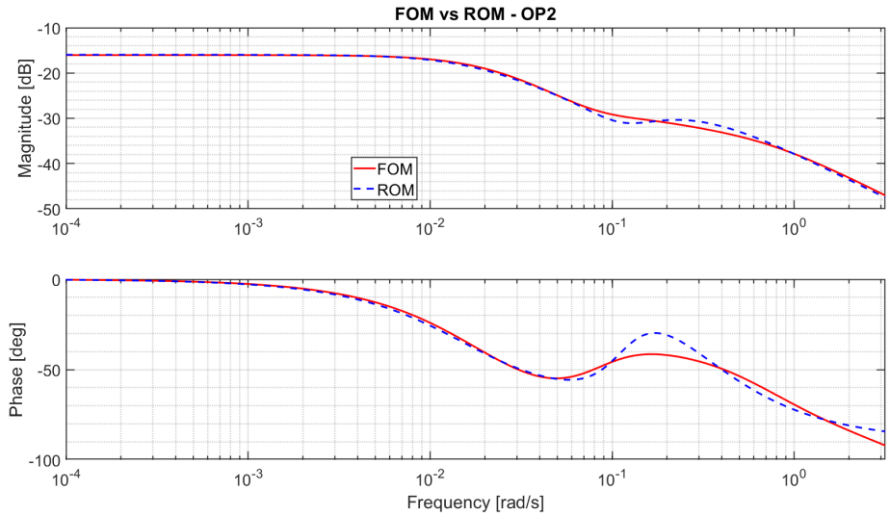


Figure 5.12: Bode comparison between FOM and ROM at OP2

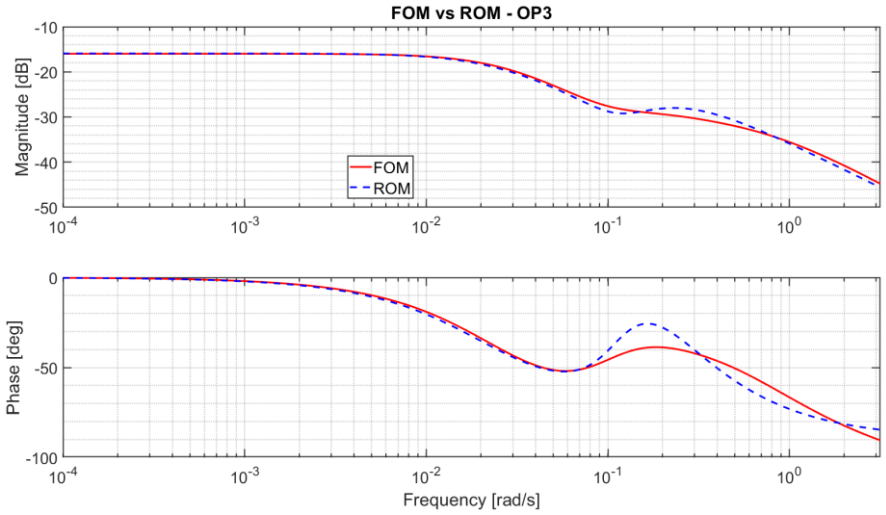


Figure 5.13: Bode comparison between FOM and ROM at OP3

5.4.2.4 Comparative Results and Discussion

The results show that all three reduced models (orders 4, 8, and 12) provide good performance in the low-frequency range, correctly reproducing the static gain and the dominant slow dynamics of the heat exchanger-turbine system. This behavior is particularly relevant from a control-oriented perspective, since the heat exchanger-turbine dynamics are inherently slow and largely governed by low-frequency phenomena. A clear improvement is observed when moving from order 4 to the order 8 reduced model. Order 8 exhibits a significantly better match in both magnitude and phase over a wider frequency range, indicating that the additional interpolation points allow the reduced dynamics to better capture the dominant modes of the system. Interestingly, the numerical experiments reveal that increasing the reduced order from 8 to 12 does not lead to further improvements. On the contrary, the order 8 reduced model outperforms the order 12, especially around 0.1 rad/s. This frequency range is of particular importance, as it contains the fundamental dynamics of the heat exchanger-turbine system, which dominate the system's response. The degradation observed for the order 12 model suggest that adding interpolation points beyond those associated with the dominant dynamics may introduce unnecessary complexity without improving, and in some cases slightly worsening, the quality of the frequency-domain approximations. This highlights an important practical insight: higher reduced order does not automatically guarantee better performance, and an appropriate selection of interpolation frequencies is often more critical than simply increasing model dimension. It is possible to state that for all three reduced models the generalization performances in the frequency domain are good, in fact the differences of the Bode plots referred to OP3 in the three cases are very similar compared to the frequency response of the full model in the band of interest. Overall, these results indicate that, in the absence of state-input product in function g , an order 8 reduced model represents the best trade-off between model complexity and accuracy for the SISO case. Another important remark is the fact that all the reduced models are asymptotically stable around the operating points. To verify this statement, the command *eig()* is used on the state matrix of the linearized reduced models. The eigenvalues are reported in table E.1 in Appendix E.

5.5.3 Case 2: Reduced Model with State-Input Product

5.5.3.1 Motivation and Dataset Generation

In the second set of experiments for heat exchanger-turbine model, the nonlinear function g is extended to include explicit state-input coupling. This choice is

motivated by physical considerations: the main nonlinear behavior of thermofluid systems is given by the flow rate enthalpy product, where the flow rate is an input and the enthalpy depends on the temperatures which are states. The dataset generation procedure is identical to that case 1, ensuring a fair comparison between the two approaches.

5.5.3.2 Reduced Model Construction

A single reduced model of order $v = 12$ is constructed using the same interpolation points as in the highest-order model of the previous case. The only difference lies in the definition of g, which now includes explicit state-input product. All other components of the reduction pipeline remain unchanged.

5.5.3.3 Frequency-Domain Validation

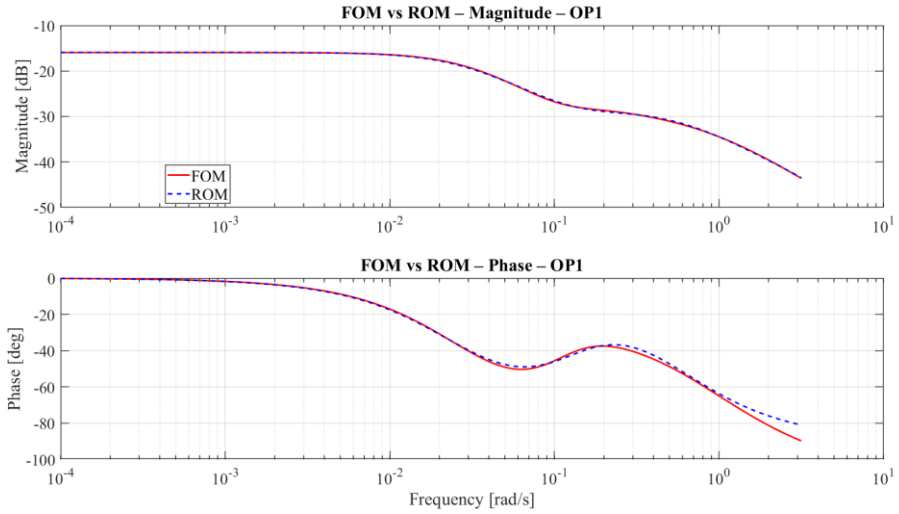


Figure 5.14: Bode comparison between FOM and ROM at OP1

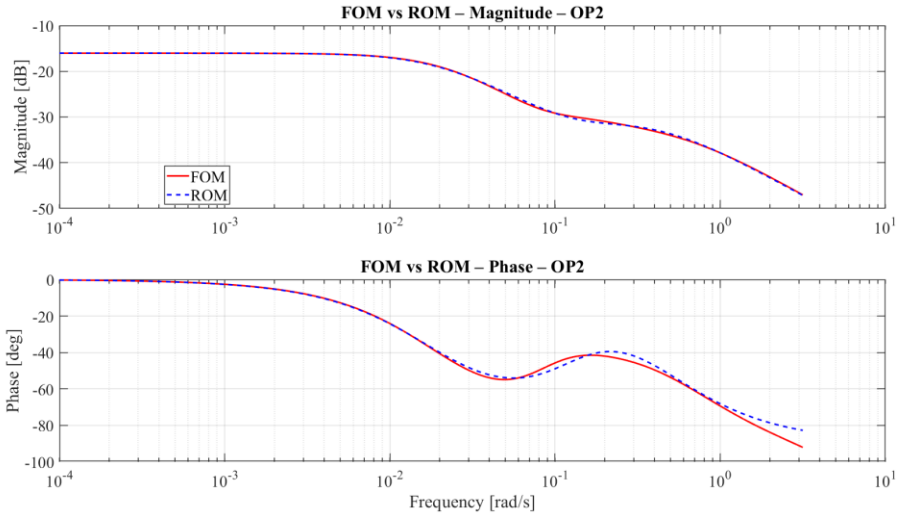


Figure 5.15: Bode comparison between FOM and ROM at OP2

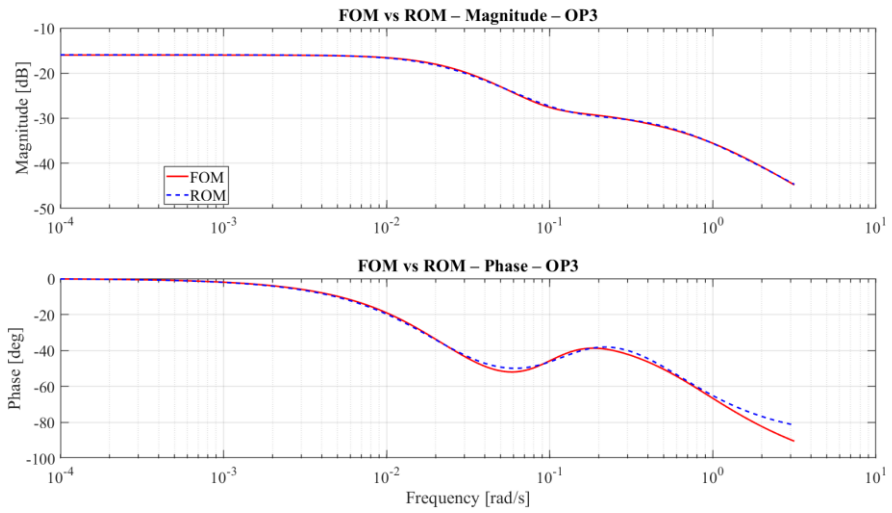


Figure 5.16: Bode comparison between FOM and ROM at OP3

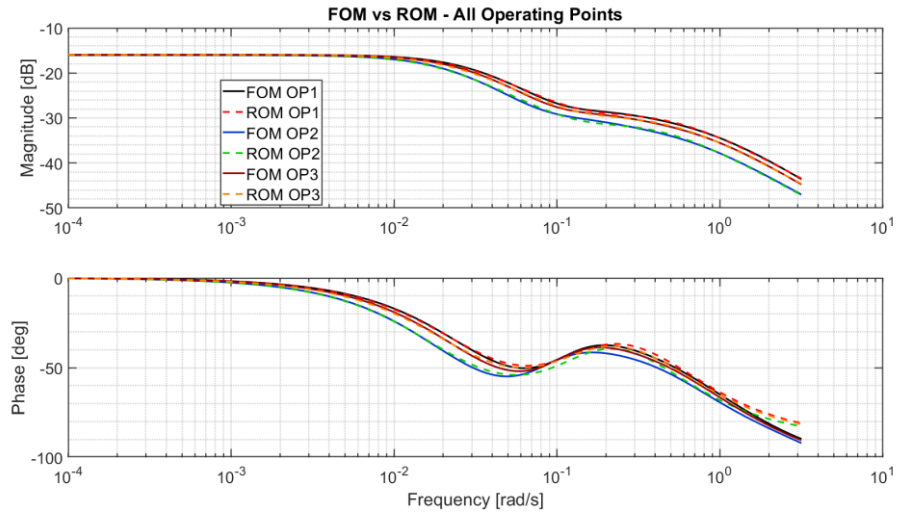


Figure 5.17: Bode comparison between FOM and ROM at all three OPs

5.5.3.4 Comparative Results and Discussion

The frequency-domain validation reveals a substantial improvement when state-input coupling is included:

- Bode magnitude and phase match closely across all three OPs, so the reduced model showed good performance in fitting and generalization;
- The mean square error between full-order and reduce-order frequency responses is significantly reduced; in particular the error is less than 0.5%;
- The reduced model exhibits better robustness with respect to operating point variations.

Importantly, the order 12 reduced model with state-input product outperforms the order 12 model without state-input product, demonstrating that structural consistency can be more important than increasing model order, moreover like the

previous case the obtained reduced model is asymptotically stable around the three operating points. The eigenvalues are reported in table E.2 in Appendix E. These results justify the choice of including state-input interaction term in the nonlinear function g, and motivate the adoption of the same structure in the reduction of the SOFC model presented in Chapter 6.

5.5.3.5 Concluding Remarks on the SISO Experiments

The SISO experiments highlight several key insights:

- The proposed reduction workflow is highly repeatable and systematic;
- Signal generator-based dataset generation enables consistent excitation across operating points;
- The structure of nonlinear function g plays a critical role in determining model accuracy;
- Incorporating physically motivated nonlinear terms allows achieving better accuracy at fixed reduced order.

5.6 MIMO Reduction: Problem Setup and Objectives

In this section, the model order reduction procedure is extended to the MIMO heat exchanger-turbine system, where both mass flow rates at the cold and hot sides of the heat exchanger are treated as manipulated variables, and multiple outputs are considered. Compared to the SISO case, the main conceptual and methodological difference lies in the computation of the moments of the full-order model, which directly impacts the construction of the reduced-order model (5.6).

5.6.1 Motivation and Methodological Differences Respect SISO Case

For SISO configuration, the Imperial College London MATLAB toolbox allows a data-driven estimation of the moments of the nonlinear system through time-domain simulations, by interconnecting the full-order model with a signal generator and constructing suitable datasets. However, this approach is not available for MIMO systems, as the toolbox does not provide data-driven tools for moment estimation in the multi-input multi-output case. As a consequence, the MIMO reduction procedure follows a non-data-driven approach, based on the analytical computation of moments of linearized models. The key idea is to exploit the toolbox functionality that computes the moment of a linear system by solving a Sylvester equation, given the state-space matrices of the system and the matrices defining the signal generator. This choice

represents the only substantial difference with respect to the SISO workflow. Once the moments are computed, the reduced-order model construction and optimization procedure remains unchanged, ensuring methodological consistency across the two cases. In particular, also in this case, the user characterizes the signal generator (5.3) by defining the interpolation points, and consequently the matrix S_0, the generator type (also in this case sinusoidal), which corresponds to imposing the matrix L. Subsequently, the function g is defined. In this way, the reduction algorithm will construct a ROM consistent with the choices made by the user, and the degree of freedom remains the same as the SISO case..

5.6.2 Operating Points and Linearization of the Full-Order Model

The system under investigation has two inputs, w_{cold} and w_{hot}, and two outputs, TIT and the electrical power output provided by the turbine P_e. The full-order nonlinear heat exchanger-turbine model is linearized around the same three OPs considered in the SISO analysis, ensuring comparability and allowing OP3 to be used as a benchmark for generalization. The three operating points correspond to different load conditions, the same of SISO case

- 30% load (OP2):

$$w_{cold} = w_{hot} = 45\ \frac{Kg}{s},\ \ Pe = 6MW\ ,\ \ TIT = 510.85\ °C\ (784\ K)$$

- 65% load (OP3):

$$w_{cold} = w_{hot} = 75\ \frac{Kg}{s},\ \ Pe = 13.65\ MW\ ,\ \ TIT = 510.85\ °C\ (784\ K)$$

- 100% load (OP1):

$$w_{cold} = w_{hot} = 100\ \frac{Kg}{s},\ \ Pe = 21\ MW\ ,\ \ TIT = 510.85\ °C\ (784\ K)$$

For each operating point, the nonlinear model is linearized, resulting in three linear MIMO systems characterized by their respective state-space matrices (A, B, C, D). These linearized models represent a local approximation of the full nonlinear

dynamics and are used both for moment computation and validation in the frequency domain.

5.6.3 Analytical Computation of Moments via Sylvester equation

Unlike the SISO case, the moments of the full-order nonlinear model are not obtained from datasets by solving (3.17). Instead, they are computed analytically from the linearized models by solving a Sylvester equation of the form (3.4). The signal generator matrices S_0 and L are constructed based on the same logic as the SISO case, following the structure introduced in Section 5.2.1. Like in the SISO case, interpolation points are selected to cover the frequency range of interest for the heat exchanger-turbine dynamics, and a small damping term is added through (5.4) to ensure numerical robustness. Moments are computed analytically for the linearized models at OP1 and OP2; the overall moment vector used for the nonlinear full-order model is then defined as the average of the moments computed at the two extreme operating points. This averaged moment representation is assumed to capture the dominant input-output behavior of the nonlinear system over the entire operating range.

5.6.4 Reduced-Order Model Structure

Once moments are available, the ROM (5.6) is constructed using the same structural framework adopted in the SISO case. The reduced dynamics are defined through a nonlinear state equation driven by a linear signal generation and a parametrized nonlinear function g.

Two different configurations are investigated, like in the SISO case:

1. Baseline case, where the function g depends only on the reduced states.
2. Extended case, where the function g explicitly includes a state-input product term.

The reduced-order dynamics are formulated in such a way that the linear input-output behavior matches the averaged moments of the full-order model, while the nonlinear function g is optimized to minimize the discrepancy the full-order and reduced-order model in the frequency domain.

5.6.5 Validation Strategy in the Frequency Domain

Since the system is MIMO, the validation of the reduced-order models is performed by comparing the singular values of the full-order and reduced-order linearized systems, rather than classical Bode plots. This choice provides a more meaningful

representation of the input-output behavior in the multivariable case. For each operating point, the reduced-order model is linearized around the equilibria, and its singular value frequency response is compared against that of the corresponding full-order linearized model. The comparison is carried out over a wide frequency range, encompassing both the slow thermal dynamics and the faster modes associated with the turbine subsystem.

5.7 MIMO Reduction: Model Order Reduction Algorithm

The overall reduction framework adopted for the MIMO case is conceptually identical to the one used for the SISO system. In particular, the structure of the reduced nonlinear dynamics, the delta-based formulation, and the optimization workflow remain unchanged. For this reason, only the main algorithmic differences are highlighted here, while the complete MATLAB implementation is reported in Appendix D. The first relevant difference concerns the definition of the signal generator matrices S and L . In the MIMO case, these matrices are constructed with a block-diagonal structure, obtained by replicating the single input generator for each input channel.

```
L=blkdiag(L,L);
S=blkdiag(S,S);
```

This construction allows the reduced model to properly account for the multi-input nature of the system while preserving the interpolation properties associated with the selected matching frequencies. The drawback is that the order of the reduced model scales with the number of inputs of the MIMO system, this fact will be highlighted in the following subsections. A second key difference is related to the computation of the moments. Unlike the SISO case, where moments are estimated in a data-driven manner, in the MIMO case the moments are computed analytically by solving a Sylvester equation using the matrices A and B of the linearized full-order models.

```
Pi=computeMoment(S,L,sys.A,sys.B);
Pi1=computeMoment(S,L,sys1.A,sys1.B);
```

The *computeMoment()* function is provide by Imperial College London MATLAB toolbox, the moments obtained from OP1, and OP2 are then averaged and used as a linear representation of the nonlinear full-order model

```
CPi_All = (CPi_Model + CPi_Model1) / 2;
```

Finally, the output map of the reduced-order model is explicitly defined as a linear function of the reduced states, based on the averaged moments, where *CPi_Model* corresponds to the linear output map $C\Pi$ treated in Chapter 3.

```
g_red=@(xi)CPi_All*xi
```

This represents a further difference with respect to the SISO case, where the output mapping is implicitly embedded in the data-driven identification procedure. Apart from these aspects, the reduction algorithm, the construction, and the structure of the reduced-order model (5.6) are identical to the SISO case. The interested reader is referred to Appendix D for complete implementation.

5.8 Case 1: Reduced Models without State-Input Product

5.8.1 Execution of the Reducing Algorithm

In the MIMO case, only one model is realized; it consists of a 12 order model, with interpolation points in 0.1 rad/s, 0.001 rad/s, 1 rad/s, to cover all band of interest. In the MIMO case there is a main drawback related to the dimension of the reduced-order model respect to the number of inputs associated to the MIMO system, the order of the reduced model scales with the number of inputs with this formula

$$reduced\ order = (interpolation\ points\ \times 2) \times number\ of\ inputs$$

This is motivated by the overall structure of the matrix S, the block diagonal structure introduces some fictitious states and so increases the order of the reduced model. In this particular case $number\ of\ inputs = 2$, $interpolation\ points = 3$, so $reduced\ order = 12$. This drawback could be strongly problematic in the cases where MIMO system with many inputs are addressed because this introduces strong limitations compared to the reduction that can be made.

5.8.2 Frequency-Domain Validation

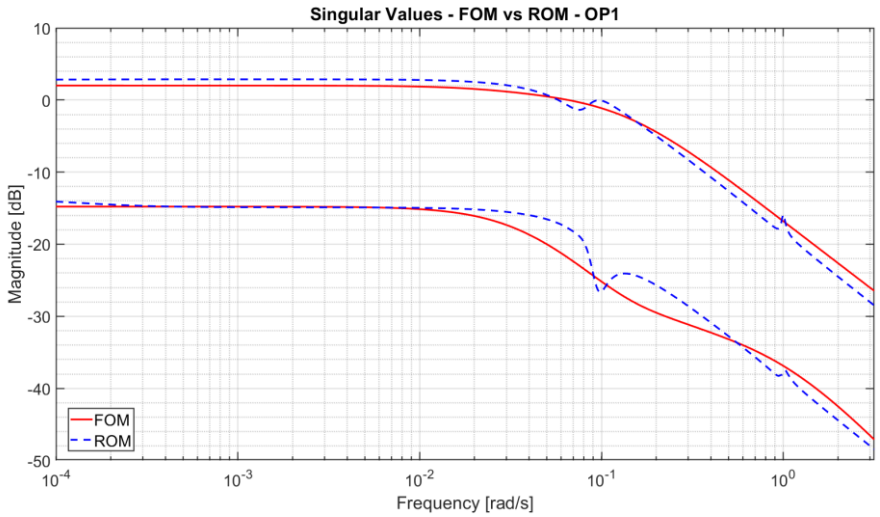


Figure 5.18: Singular values comparison between FOM and ROM at OP1

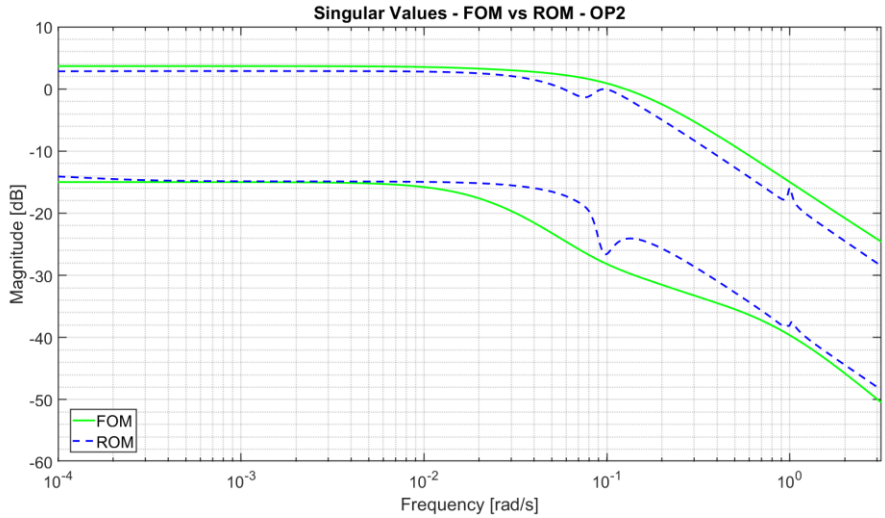


Figure 5.19: Singular values comparison between FOM and ROM at OP2

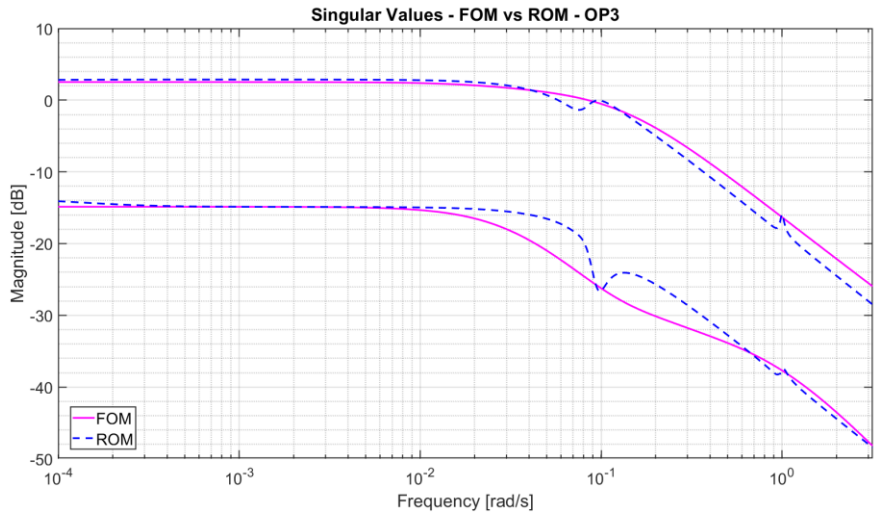


Figure 5.20: Singular values comparison between FOM and ROM at OP3

5.8.3 Comparative Results and Discussion

This subsection discusses the result obtained for the MIMO case using a reduce-order model of order 12 constructed without including the state-input product in the function g . The analysis highlights that the overall performance of the reduced model is unsatisfactory for control-oriented applications. Significant discrepancies emerge in the frequency domain, where pronounced resonance peaks appear in the singular values of the reduced model. These peaks are consistently observed at frequencies around 0.1 rad/s and 1 rad/s for all three operating points. Such resonant behaviors are especially critical, as these frequency ranges are directly associated with the dominant dynamics of the system. The presence of artificial amplification effects in the reduced model indicates that the nonlinear structure adopted in this case is unable to adequately capture the correct behavior of the full-order model. As a consequence, the reduced model exhibits a distorted dynamic response that is incompatible with the requirements of reliable controller design.

From a static perspective, the reduced model shows moderate/good accuracy at low frequencies, with steady-state gains that remain reasonably close to those of the full-order model. Although this level of precision may be considered acceptable for preliminary analysis, it is insufficient to compensate for the severe degradation observed in dynamic behavior.

Regarding stability, the reduced-order model remains asymptotically stable at all three operating points, as confirmed by the eigenvalues of the linearized reduced model. The eigenvalues are reported in table E.3 in Appendix E. However, stability alone is not sufficient to ensure suitability for control purposes when significant performance issues arise in the frequency response, moreover the presence of

resonance peaks could affect the Bounded Input Bounded Output (BIBO) stability of the reduced model. Overall, these results clearly indicate that, in the MIMO case, neglecting the state-input interaction in the function δ leads to a reduced-order model with poor dynamic fidelity. This outcome motivates the introduction of a state-input product in the function g , which is investigated in the following section.

5.9 Case 2: Reduced Models with State-Input Product

5.9.1 Execution of the Reducing Algorithm

Also, in this case, a reduced model of order 12 is generated; the only difference is the function g in which the state-input product is embedded.

5.9.2 Frequency-Domain Validation

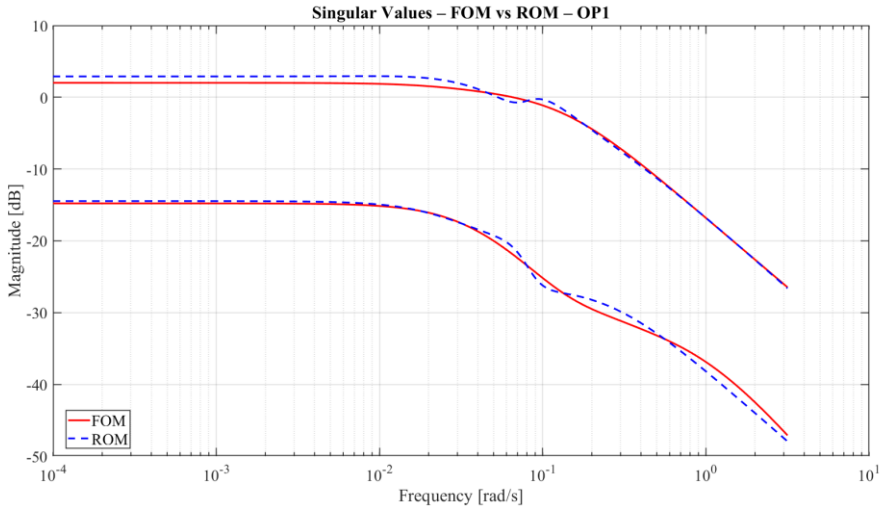


Figure 5.21: Singular values comparison between FOM and ROM at OP1

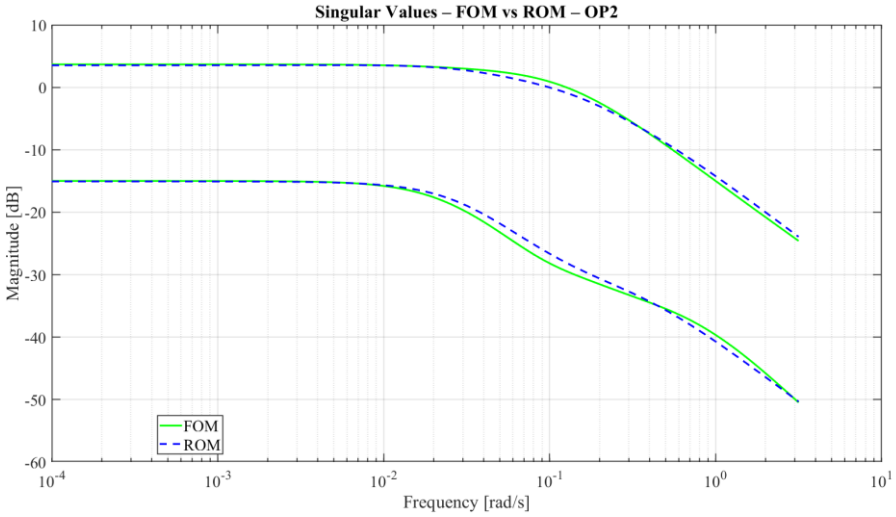


Figure 5.22: Singular values comparison between FOM and ROM at OP2

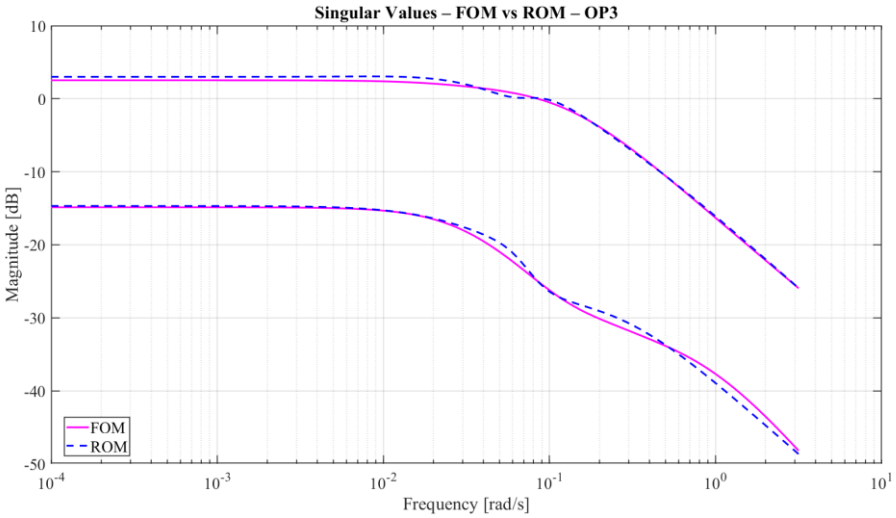


Figure 5.23: Singular values comparison between FOM and ROM at OP2

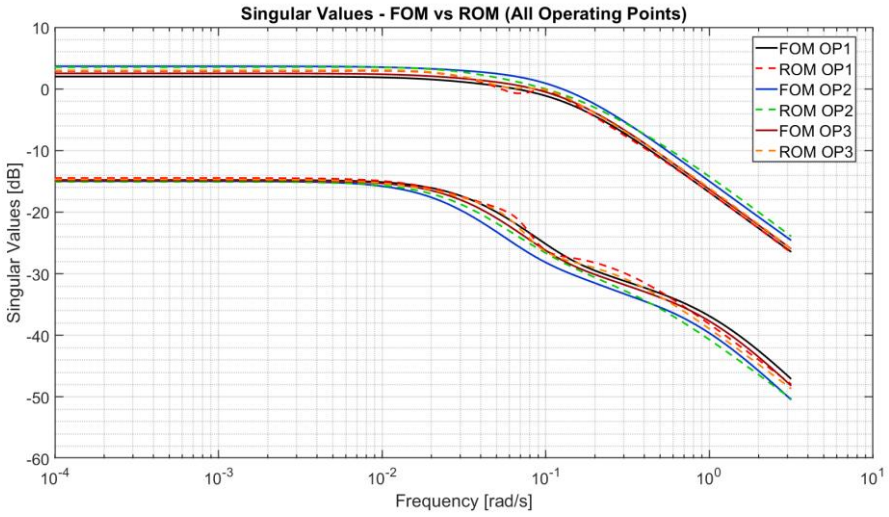


Figure 5.24: Singular values comparison between FOM and ROM at all three OPs

5.9.3 Comparative Results and Discussion

This subsection discusses the results obtained for the MIMO case using a reduced-order model of order 12. The inclusion of state-input products leads to substantial improvements in the quality of the reduced model. In contrast with the result obtained without the state-input product, the singular value plots now show an excellent agreement with those of the full-order model across the entire frequency range of interest. The resonance peaks previously observed at 0.1 rad/s and 1 rad/s are significantly attenuated or eliminated for all operating points. Quantitatively, the improvement is confirmed by the mean square error between the singular values of the reduced and full models, which remains below 1% in all considered cases. This level of accuracy demonstrates that the reduced model is capable of faithfully reproducing the dominant multivariable dynamics of the original system, especially in the frequency range relevant to control design.

From a steady-state perspective, the reduced model exhibits excellent low-frequency performance, accurately matching the static gains of the full-order model. This behavior is essential for regulations tasks, where correct steady-state input-output relationship must be preserved across different operating conditions.

With respect to stability, the reduced-order model remains asymptotically stable for two out of three operating points, as confirmed by the eigenvalue analysis of the linearized systems the operating point 1 has a couple of complex conjugate poles with positive real part. The eigenvalues are reported in table E.4 in Appendix E. To solve this issue is sufficient to add one more interpolating point at 0.1 rad/s; so, with a reduced model of order 16 the stability is guaranteed and performance are the same as order 12. The eigenvalues are reported in table E.5 in Appendix E. Importantly, improved dynamic behavior is achieved without compromising stability, which further supports the robustness of the proposed modeling approach.

Overall, these results clearly demonstrate that incorporating the state-input product into the function g is a key modeling choice for the MIMO case. This enhanced nonlinear structure allows the reduced-order model to capture the intrinsic coupling between inputs and internal states, leading to accurate frequency-domain behavior, reliable steady-state performance, and guaranteed asymptotically stability. As a result, the reduced model obtained with this approach can be regarded as fully suitable for control-oriented applications, providing a compact yet accurate representation of the original heat exchanger-turbine system for multivariable control design and analysis.

5.10 Concluding Remarks and Transition to the SOFC Case Study

This chapter has investigated the application of nonlinear moment matching MOR techniques to the heat exchanger-turbine system, considering both SISO and MIMO configurations. The results obtained throughout the chapter provide a comprehensive assessment of the strengths and limitations of the proposed reduction framework when applied to a complex thermo-fluid dynamic system. For the SISO case, the reduction workflow based on data-driven moment estimation proved effective in generating reduced-order models capable of accurately reproducing the dominant dynamics of the full-order system around multiple operating points. The systematic use of signal generators for dataset generation enables a reputable and flexible reduction procedure, well suited for control-oriented applications. Moreover, the comparative analysis across different reduced orders highlighted the trade-off between model complexity and dynamic accuracy, showing that increasing the order does not necessarily lead to better performance if the dominant physical dynamics are already captured. In the MIMO case, the limitations of a purely data-driven moment estimation approach motivated the adoption of an alternative strategy based on the analytical computation of moments from linearized models. This is a hybrid approach because the notion of moment matching is already valid and exploited, but the signal generator is not necessary in that case.

Despite this methodological difference, the reduced models retained the same structural form as in the SISO case, ensuring conceptual consistency across configurations. The results clearly showed that reduce-order models without state-input interaction suffer from poor dynamic behavior, particularly in the form of resonance peaks and reduce static accuracy, rendering them unsuitable for control-oriented purposes. A key outcome of this chapter is the identification of the state-input product in the function g as a crucial modeling ingredient. Its inclusion consistently led to significant improvements in frequency-domain accuracy, static performance, and robustness across operating points, both in SISO and MIMO settings. This observation suggests that incorporating prior physical insight into the structure of the nonlinear function g is essential to capture the intrinsic coupling mechanisms of energy systems governed by transport phenomena and power conversion processes.

Overall, the heat exchanger-turbine case study serves as a validation platform for the proposed nonlinear reduction framework. It demonstrates that accurate, low-order, and asymptotically stable reduced models can be obtained from large-scale, nonlinear,

and multivariable thermo-fluid systems, provided that the reduced dynamics are endowed with sufficiently rich nonlinear structure. These findings naturally motivate the extension of the proposed approach to more complex energy systems characterized by stronger nonlinearities, tighter multiphysics coupling, and higher intrinsic dimensionality. In particular, the SOFC system introduces additional challenges due to the coexistence of electrochemical, thermal, and chemical dynamics, as well as distributed effects arising from spatial discretization. In the next chapter, the reduction methodology developed and validated here is applied to the SOFC model. The focus will be on assessing the scalability and robustness of the approach when confronted with electrochemical reaction kinetics, transport processes across the PEN structure, and multivariable power generation dynamics, thereby further demonstrating its suitability for advanced control-oriented modeling of complex energy conversion systems.

6 Model Order Reduction of Solid Oxide Fuel Cell

6.1 SISO Reduction: Problem Setup and Objectives

This section addresses the application of the nonlinear moment matching model order reduction framework to the SOFC model in a SISO configuration. In line with the methodology adopted for the heat exchanger-turbine system, the reduction algorithm treated in Section (5.3), the structure of the ROM (5.6), and the nonlinear function g are preserved. The focus of this chapter is therefore on the definition of the control-oriented reduction problem for the SOFC, the selection of operating points, and the analysis of the resulting reduced models. The SOFC represents a significantly more complex system than the heat exchanger-turbine case, due to the overall order of 268 states, making it unsuitable for control design and real-time applications without prior reduction.

6.1.1 Control-Oriented Motivation

From a control-oriented perspective, the SOFC system is typically operated to regulate electrical output variables while respecting thermal and electrochemical constraints. In the SISO configuration considered in this chapter, the primary objective is to accurately capture the input-output dynamics between the electrical current drawn from the stack and the resulting output voltage. This choice is motivated by the fact that:

- The stack current directly governs the electrochemical reaction rates at both electrodes;

- The output voltage encapsulates the combined effect of reversible electrochemical potential and irreversible losses;
- Voltage regulation and prediction are key requirements for energy management strategies.

Consequently, a reduced-order model capable of accurately reproducing the voltage response to current variations over a wide operating range is of significant practical interest.

6.1.2 Selection of Input and Output Variables

In the SISO formulation adopted in this chapter, the variables of interest are defined as follows:

- Input stack current I ;
- Output stack voltage V_{cell}.

Moreover, the other inputs of the system

- Anode flow rate w_{anode};
- Anode temperature T_{anode};
- Anode composition X_{anode};
- Cathode flow rate $w_{cathode}$;
- Cathode temperature $T_{cathode}$;
- Cathode composition $X_{cathode}$.

Are considered constants. All other quantities, including temperature distributions, and internal electrochemical states, are treated as internal dynamics of the system and implicitly accounted for in the reduced-order model. This selection differs from the heat exchanger-turbine case only in the physical meaning of the variables involved; from a methodological standpoint, the reduction framework remains unchanged.

6.1.3 Definition of Operating Points

Unlike the heat exchanger-turbine case, steady-state operating points for the SOFC are not determined through close-loop simulations with a PI controller. Instead, they are identified directly from the polarization curve of the stack, which characterizes the steady-state voltage-current relationship. Three operating points are selected along the V-I polarization curve to capture the dominant nonlinear regimes of the SOFC:

- Low-current operating point OP2 (activation losses dominant region)

$$I = 2\ A \qquad V_{cell} = 0.94949\ V$$

- High-current operating point OP1 (concentration losses dominant region)

$$I = 9\ A \qquad V_{cell} = 0.715573\ V$$

- Intermediate-current operating point OP3 (ohmic losses dominant region)

$$I = 6\ A \qquad V_{cell} = 0.830759\ V$$

	OP2	OP1	OP3
I	2 A	9 A	6 A
V_{cell}	0.94949 V	0.715573 V	0.830759 V

Table 6.1: Equilibrium values of the three OPs

OP1, and OP2 are used for dataset generation and model reduction, while OP3 is reserved exclusively for the assessment of generalization capabilities, in direct analogy with the approach adopted in Chapter 5.

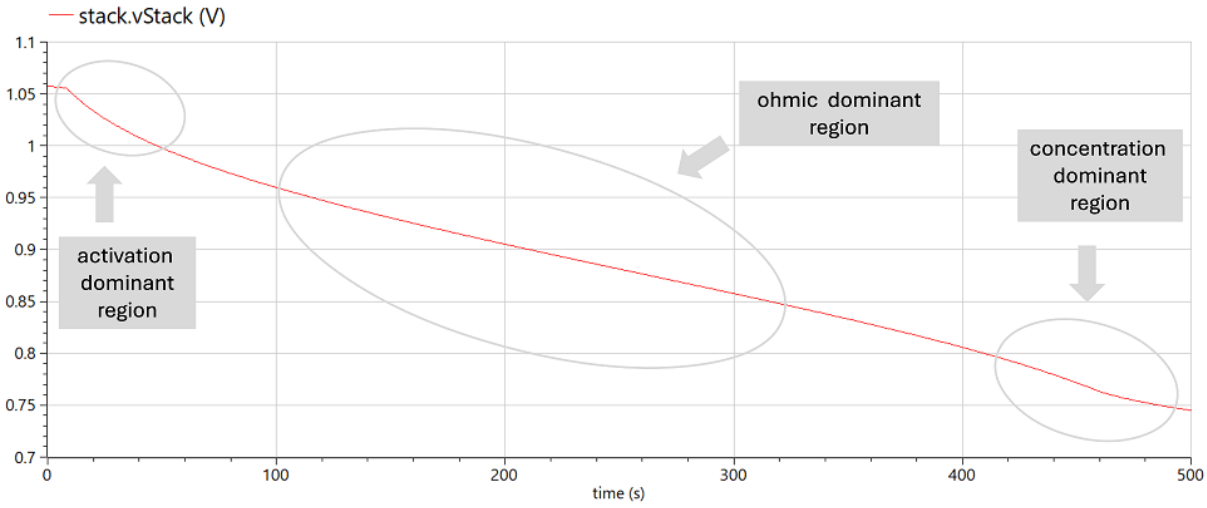


Figure 6.1: V_{cell} (V) response to ramp on I (A)

6.1.4 Computation of Steady-State Conditions

To define the values of the three operating point a simulation with a slow ramp on the input current I was used, this approach permits to identify with precision the value of current, and the corresponding voltage, related to a particular point on the polarization curve.

6.1.5 Dataset Generation for Model Reduction

Datasets are generated by exciting the full-order SOFC model with a signal generator (5.3) connected to the input current, as already introduced in Section 5.2.1 and in Appendix C. For each of the first two operating points the amplitude is perturbed around its equilibrium value using a bounded excitation signal of the form (5.1), the amplitude of the excitation is chosen to remain within a small neighborhood of the operating point, avoiding excursions that could move away the system from equilibrium point. After the simulation, the full-order output voltage and the internal states of the signal generator are recorded in a dataset (5.2).

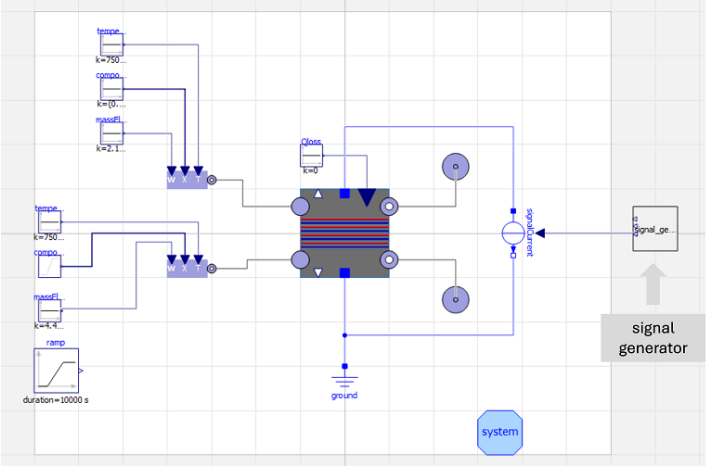


Figure 6.2: Interconnected model adopted for datasets generation

6.1.6 Validation Strategy

The validation of the reduce-order models is performed entirely in the frequency domain. Specifically:

- The full-order SOFC model is linearized around each operating point;
- The reduce-order model is linearized around the corresponding reduced equilibrium;
- Bode plots of the linearized models are compared.

The reduction algorithm optimizes, as in the previous case, the parameters of the function g by minimizing the frequency-domain discrepancy between the full-order and reduced-order linearization, following the same objective formulation used in Chapter 5.

6.2 SISO Reduction: Model Order Reduction Algorithm

The model order reduction algorithm employed for the SOFC case is identical to the one described in Section 5.3. The same reduce-order model structure, nonlinear

function g with state-input product, and optimization procedure are adopted. The only things that change are matching points and datasets. For completeness, the full implementation of the algorithm is reported in Appendix D. To avoid redundancy, no further algorithmic details are discussed in this chapter.

6.3 SISO Reduction: Numerical Experiments and Results

6.3.1 Experimental Setup and Control Objective

The numerical experiments aim at assessing the ability of the reduced-order models to reproduce the voltage dynamics of the SOFC over a wide range of operating conditions. All simulations are carried out using full-order SOFC model described in Chapter 4 and the reduced-order model obtained through the proposed reduction framework.

6.3.2 Reduced Models

In this section, the reduced-order models obtained for the SOFC system in the SISO configuration are presented and discussed. Two reduced models are considered, with dimensions equal to 12 and 16, respectively. The choice of these two orders is motivated by the need to investigate the trade-off between model complexity and accuracy, while avoiding excessive redundancy in the analysis. Both reduced models are constructed using the reduction algorithm described in Chapter 5, adopting the function g that explicitly includes the state-input product. The only difference in this case is in the structure of the moment function, which is now estimated as a nonlinear function; this is achieved by considering a nonlinear term of third power into the dictionary of functions *dictionaryOfFunc.* These choices are consistent with the conclusions drawn from the heat exchanger-turbine case, where the inclusion of the state-input coupling proved essential to accurately capture the system dynamics over a wide operating range, but in this case, due to the strong nonlinearities of the SOFC model is necessary to estimate a nonlinear moment instead of a linear one. The reduced models are obtained by matching moments at selected interpolation frequencies, chosen to capture both the dominant low-frequency dynamics and the faster transient behavior of the SOFC. For the model of order 12, the interpolation frequencies are set to 0.1, 0.01, and 1 rad/s, while for the model of order 16 the interpolation focuses on 0.1 and 1 rad/s, with an increased number of matched moments per frequency.

6.3.2.1 Dataset Generation via Signal Generator

Dataset generation follows the same procedure adopted in the SISO heat exchanger-turbine case. The signal generator structure (5.3), parameters, and degrees of freedom are unchanged, confirming the generality and repeatability of the proposed workflow. The only things that change are the equilibrium points of excitation.

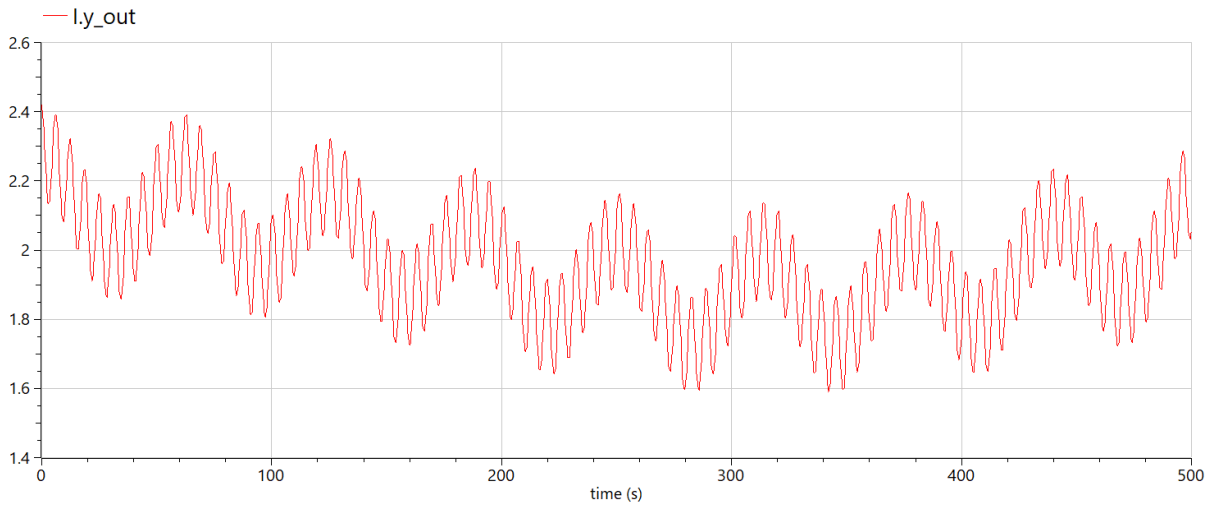


Figure 6.3: Signal generator I (A) of order $v = 12$ around OP2

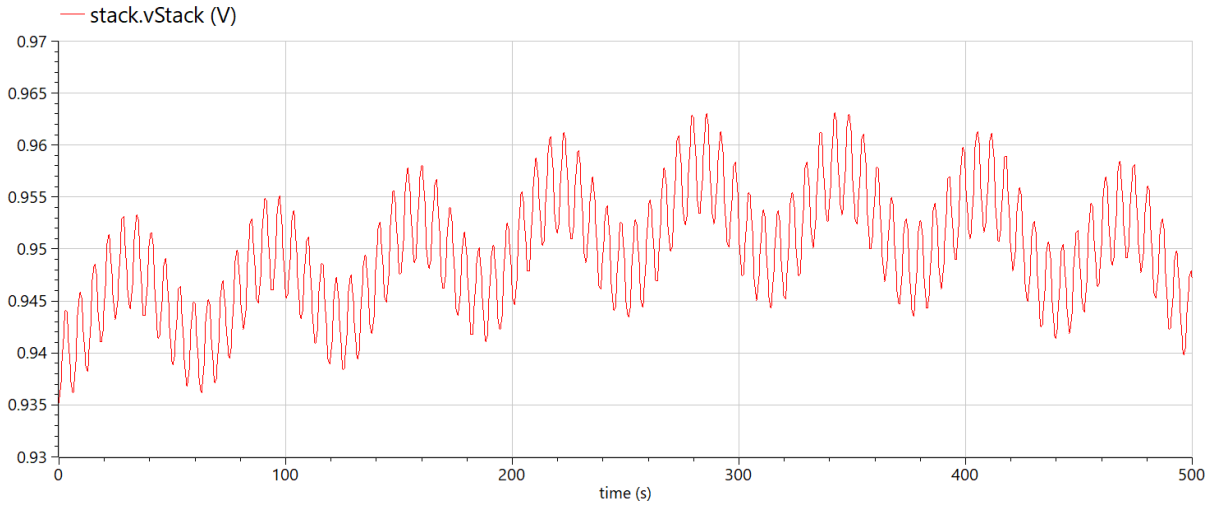


Figure 6.4: V_{cell} (V) response to signal generator of order $v = 12$ around OP2

6.3.2.2 Execution of the Reduction Algorithm

The reduction algorithm is executed using the datasets generated at the first two operating points. The optimization procedure identifies the parameters of the nonlinear function g , and in the section of datasets pre-processing, the voltage outliers values are removed.

6.3.2.3 Frequency-Domain Validation

In this subsection, the results provided by the reduction algorithm are analyzed, in particular the comparison between the Bode plots of the full-order model and the Bode plots of the reduced one in the three cases shown above.

1. Reduced model of order $v = 12$

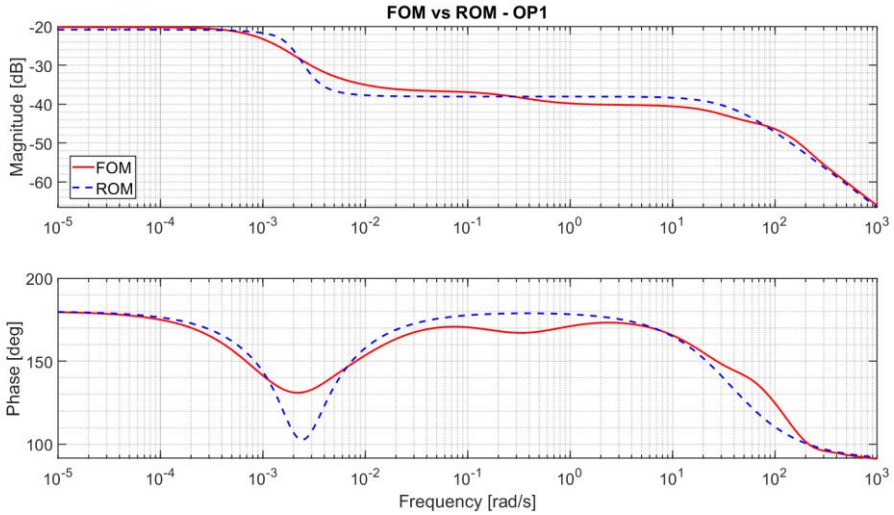


Figure 6.5: Bode comparison between FOM and ROM at OP1

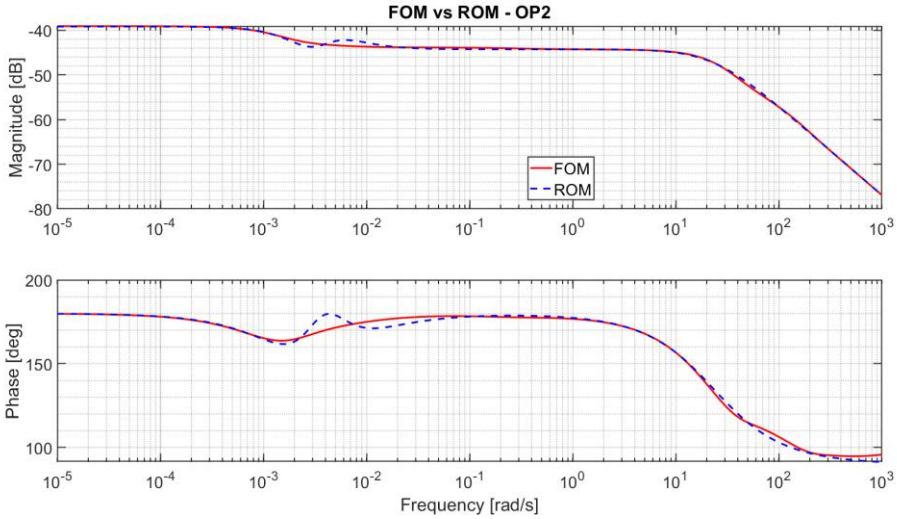


Figure 6.6: Bode comparison between FOM and ROM at OP2

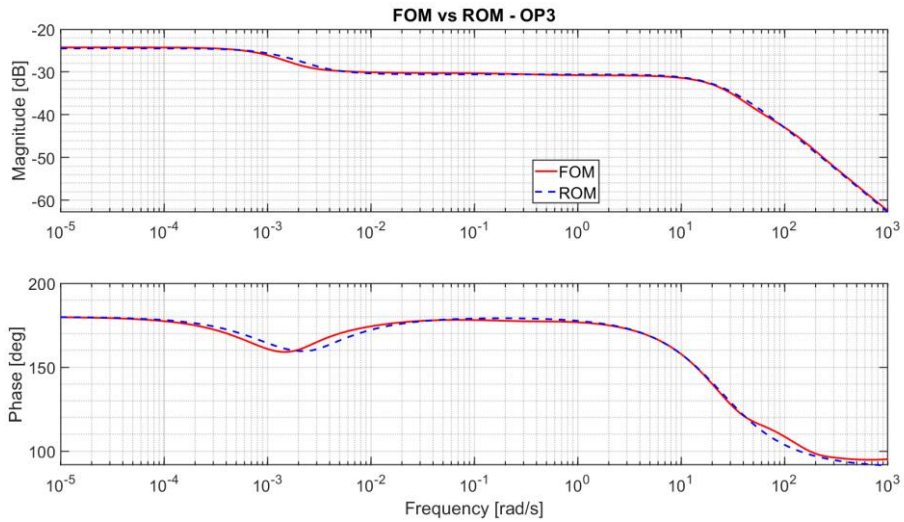


Figure 6.7: Bode comparison between FOM and ROM at OP3

2. Reduced model of order $v = 16$

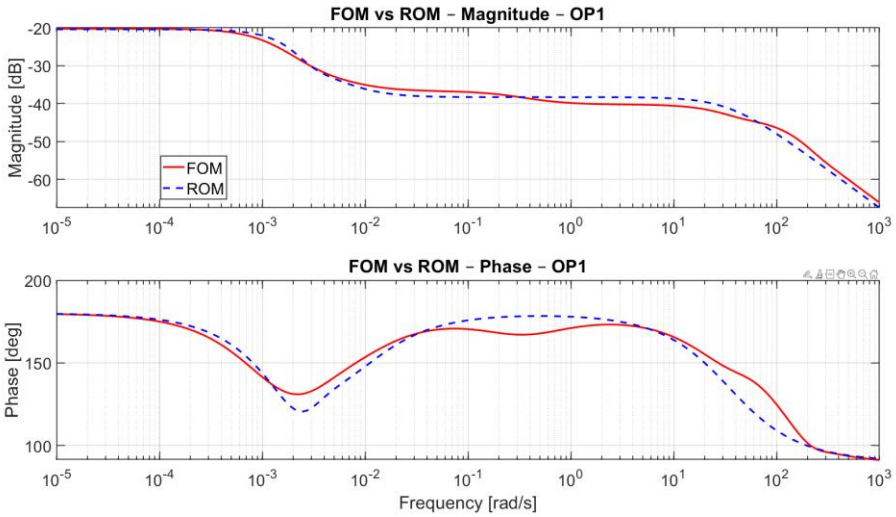


Figure 6.8: Bode comparison between FOM and ROM at OP1

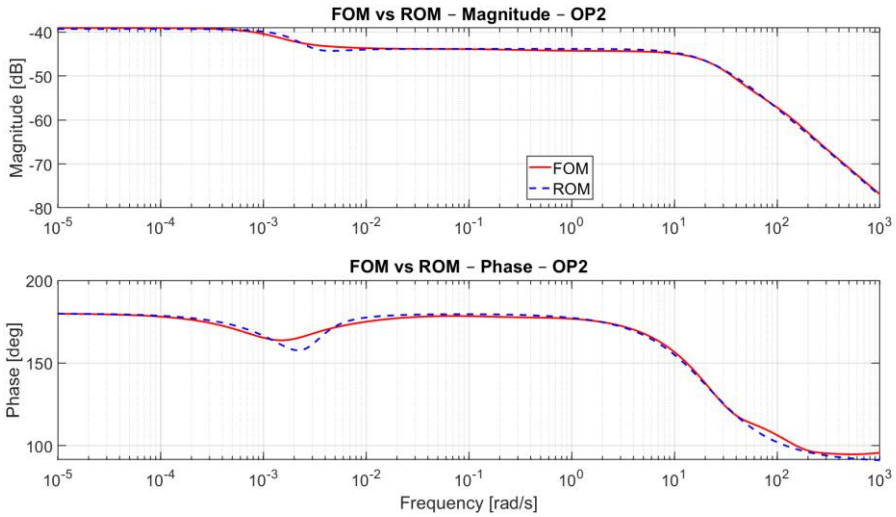


Figure 6.9: Bode comparison between FOM and ROM at OP2

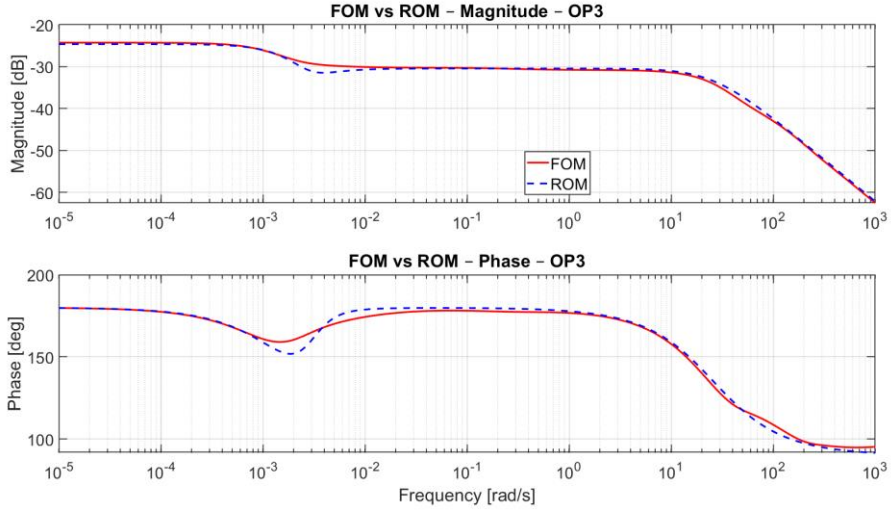


Figure 6.10: Bode comparison between FOM and ROM at OP3

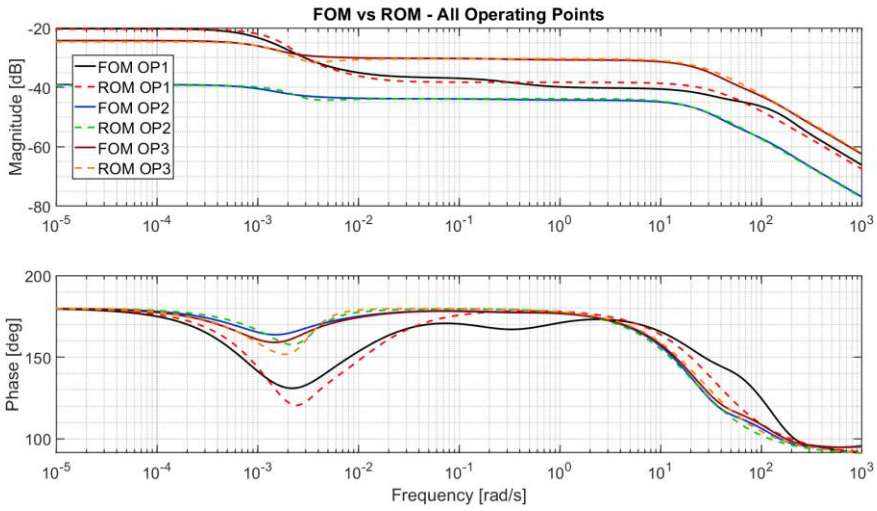


Figure 6.11: Bode comparison between FOM and ROM at OP1, OP2, and OP3

6.3.2.4 Comparative Results and Discussion

Both reduced models exhibit a good agreement with the full-order model in terms of magnitude response, particularly at low frequencies, which are associated with steady-state behavior and slow thermal dynamics. The agreement remains satisfactory over a wide frequency range, confirming that the dominant dynamic characteristics of the SOFC are well captured. More pronounced differences emerge in the phase response, especially around 0.002 rad/s. Nevertheless, the phase accuracy improves consistently as the model order increases from 12 degrees of freedom are effectively exploited by the reduction algorithm.

A comparative analysis of the reduced models highlights a clear improvement in performance when the order increases. The reduced model of order 12 achieves a mean square error below 5% across the frequency range of interest, which can already be considered satisfactory for many control-oriented applications. However, the model of order 16 further reduces the mean square error to values below 3%, providing a noticeably better match, particularly in the phase response. The improvement offered by the high-order model is especially evident when analyzing the frequency-domain behavior around the interpolation points, where the reduced model of order 16 closely follows both the magnitude and phase of the full-order system. This confirms the effectiveness of the nonlinear moment matching approach in exploiting additional model order to improve accuracy in a structured and meaningful way and preserving stability The eigenvalues are reported in tables E.6, and E.7 in Appendix E.

Considering that the full-order SOFC model has a state dimension (order) of 268, the reduced model of order 16 represents a reduction of more than 90% in model order, while still maintaining good static accuracy and a faithful representation of the relevant dynamics. This result demonstrates that the proposed reduction framework is well suited for complex electrochemical systems and can produce compact, reliable, and control-oriented reduced-order models.

6.4 MIMO Reduction: Problem Setup and Objectives

This section addresses the application of the nonlinear moment matching model order reduction framework to the SOFC model in a MIMO configuration. In line with the methodology adopted for the MIMO case of heat exchanger-turbine model, the reduction algorithm, the structure of ROM, and the nonlinear function g are preserved. In the following sections, priority will be given to defining the criteria used to select the OPs around which to build the ROMs, and to analyzing the results

obtained with a frequency approach by comparing singular values, like in the previous MIMO reduction case.

6.4.1 Control-Oriented Motivation

In the MIMO configuration of the SOFC model, two manipulated variables are considered: the anode mass flow rate w_{anode} and the stack current I. The selected outputs are the cell voltage V_{cell} and the generated electrical power. This configuration better reflects realistic operational scenarios, where both fuel supply and electrical load can be actively manipulated to regulate performance and efficiency. From a control-oriented perspective, this multivariable formulation is significantly more representative of practical energy management strategies. In real applications, the SOFC stack operates under varying load demands while fuel utilization must be carefully controlled to ensure efficiency, prevent fuel starvation, and maintain safe thermal conditions. The simultaneous regulation of voltage and power requires coordinated manipulation of both electrochemical and flow-related dynamics. The coupling between inputs and outputs is intrinsically nonlinear. Variations in current directly affect the electrochemical reaction rates, influencing voltage through activation, ohmic, and concentration losses. At the same time, the anode mass flow rate impacts fuel concentration, partial pressures, and thermal balances, indirectly affecting both voltage and power. As a consequence, the system exhibits strong cross-couplings and different time-scale dynamics, which significantly increases the complexity of controller design. Unlike the SISO case, where only the voltage response to current variations is considered, the MIMO configuration requires a faithful representation of the interaction structure between fuel dynamics and electrochemical dynamics. This makes the reduction problem more challenging and highlights the importance of selecting an appropriate nonlinear ROM structure. In particular, the presence of strong nonlinearities, and strong couplings suggests that a ROM relying solely on a linear moment function may be insufficient. The inclusion of a nonlinear moment function becomes crucial to preserve multivariable dynamic behavior. For these reasons, the MIMO SOFC reduction represents not only an extension of the SISO framework, but also a more demanding test of the proposed nonlinear moment-matching methodology in a realistic control-oriented context.

6.4.2 Operating Points and Linearization of the Full-Order Model

The system under investigation has two inputs, w_{anode}, and I , and two outputs, V_{cell}, and electrical power generated by the stack, named P_e. The main objective in defining the OPs for model reduction is to ensure that the ROM accurately captures the full range of relevant input-output interactions. In a multivariable nonlinear system such as the SOFC, dynamic behavior strongly depends on the operating region. Therefore, it is not sufficient to linearize around a single equilibrium. Instead, a structured selection of OPs is required to span the relevant combinations of manipulated variables. To systematically explore the operating domain, three representative values of the aode mass flow rate are selected. For each fixed value of w_{anode}, three different current values are chosen along the corresponding polarization curve. This procedure results in a total of nine OPs. The rationale behind this approach is twofold. First, by fixing w_{anode} and varying the current, one obtains a family of polarization curves parametrized by the fuel flow rate. This allows the reduction process to account for variations in fuel utilization and associated thermochemical dynamics. Second, for each curve, the three selected current values are chosen to represent the three characteristic regions of the SOFC operation:

- A low-current region dominated by activation losses;
- A high-current region dominated by concentration losses;
- An intermediate region where ohmic losses are predominant, and the V-I curve is approximately linear.

By repeating this selection for each of the three w_{anode} values, the nine OPs collectively cover a broad portion of the feasible operating domain. This structured grid-like selection ensures that the ROM is exposed to variations in both electrochemical loading and fuel supply, thereby capturing the essential multivariable nonlinear interactions. The resulting OPs reflects not only different electrical loading conditions, as in the SISO case, but also variations in mass flow dynamics and their coupling with electrochemical and thermal phenomena. This is particularly important in the MIMO setting, where cross-coupling between inputs and outputs plays a significant role in determining system behavior. The nine OPs will be reported below:

- 100% mass flow rate (OP1) $w_{anode} = 4.5e-7\ \frac{Kg}{s}$
  - OP1.1 $I = 2\ A$, $V_{cell} = 0.95\ V$, $P_e = 1.86\ W$;
  - OP1.2 $I = 9\ A$, $V_{cell} = 0.72\ V$, $P_e = 6.31\ W$;

    - OP1.3 $I = 6\ A$, $V_{cell} = 0.83\ V$, $P_e = 4.88\ W$;
- 35% mass flow rate (OP2) $w_{anode} = 1.57e-7\ \frac{Kg}{s}$
    - OP2.1 $I = 2\ A$, $V_{cell} = 0.94\ V$, $P_e = 1.84\ W$:
    - OP2.2 $I = 9\ A$, $V_{cell} = 0.70\ V$, $P_e = 6.17\ W$;
    - OP2.3 $I = 6\ A$, $V_{cell} = 0.76\ V$, $P_e = 4.47\ W$;
- 65% mass flow rate (OP3) $w_{anode} = 2.92e-7\ \frac{Kg}{s}$
    - OP3.1 $I = 2\ A$, $V_{cell} = 0.95\ V$, $P_e = 1.85\ W$;
    - OP3.2 $I = 9\ A$, $V_{cell} = 0.81\ V$, $P_e = 4.76\ W$;
    - OP3.3 $I = 6\ A$, $V_{cell} = 0.71\ V$, $P_e = 6.26\ W$;

6.4.3 Reduced Order Model Structure

Once the moments are computed analytically like in the previous MIMO case, the ROM is constructed using the same structural framework adopted in the previous cases. The reduced dynamics are defined through a nonlinear state equation driven by a signal generator and a parametrized nonlinear function g ; also, in that case, the moments are computed as a linear function.

6.4.4 Validation Strategy in the Frequency Domain

Like in the MIMO case of heat exchanger-turbine model, the validation of the ROMs is performed by comparing the singular values of the linearized full-order model and the linearized reduced one.

6.5 MIMO Reduction: Model Order Reduction Algorithm

The reduction algorithm adopted for the MIMO SOFC case follows the same conceptual framework introduced for the MIMO heat exchanger-turbine model. The nonlinear moment matching strategy, the structure of the reduced model, and the optimization-based parameter identification procedure remain unchanged. The reduction process is based on the computation of linear moments at multiple OPs defined in Section 6.4.2, and on the construction of a nonlinear ROM whose parameters are optimized to minimize the discrepancy between the frequency responses of the full and the reduced model. As in the previous MIMO case, the reduced model structure includes the nonlinear function g with state-input coupling, ensuring sufficient flexibility to capture multivariable nonlinear interactions. The main differences in the SOFC MIMO case lies in the way the overall linear moment information is aggregated. Since nine OPs are considered, the linear moments are

computed at each of these equilibria. The global linear moment used for the reduced model construction is then defined as the average of the moments obtained at the nine OPs. This averaging procedure allows the ROM to capture the input-output behavior across the entire selected operating domain. Another relevant difference concerns the presence of feedthrough in the SOFC MIMO model. Unlike the previous case, the full-order SOFC model exhibits a nonzero direct transmission matrix D . For this reason, the matrix D of the ROM is included among the optimization parameters. During the identification procedure matrix is optimized to ensure consistency in input-output behavior of the ROM. The complete implementation details are not repeated here and can be referred to in the corresponding sections and in Appendix D, where the code is reported.

6.6 MIMO Reduction: Reduced Models

6.6.1 Execution of the Reducing Algorithm

In the SOFC MIMO case, only one ROM is realized; it consists of a 12 order model, with interpolation points in 0.1 rad/s. In particular, 3 matches are enforced around 0.1 rad/s, and considering that the number of inputs is 2, the overall ROM order is 12.

6.6.2 Frequency Domain Validation

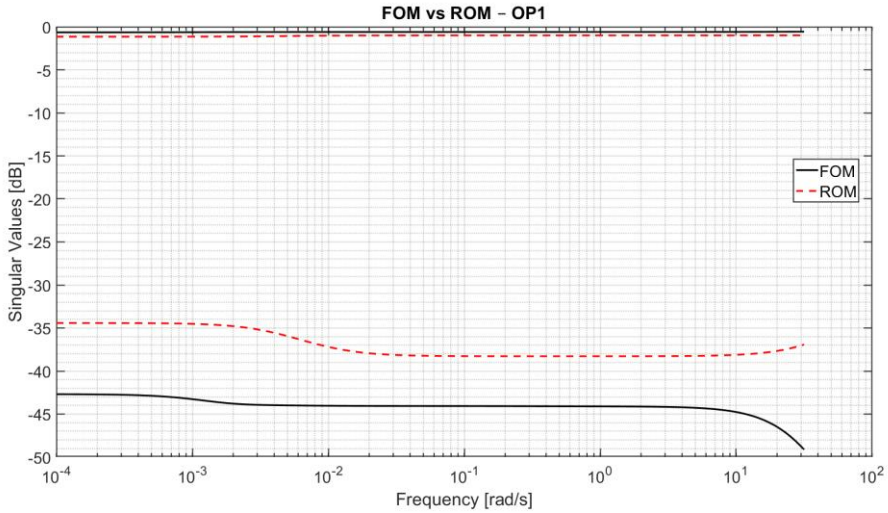


Figure 6.12: Singular values comparison between FOM and ROM at OP1.1

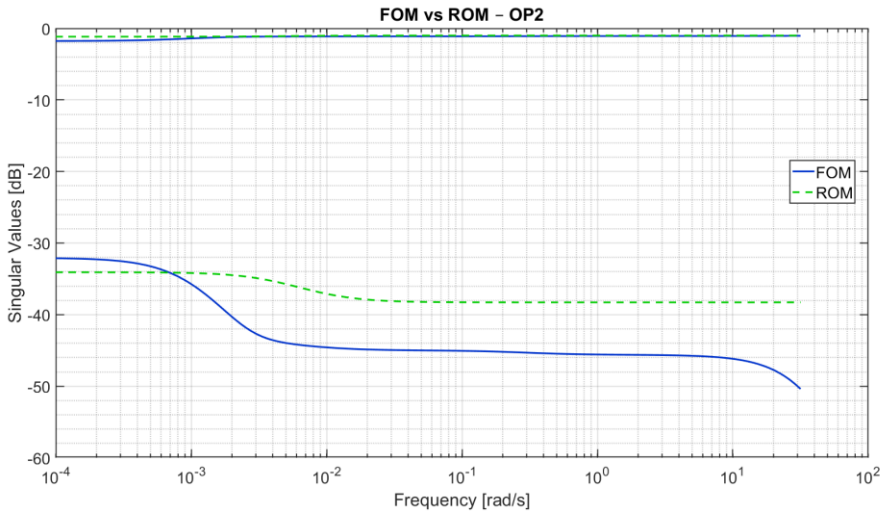


Figure 6.13: Singular values comparison between FOM and ROM at OP1.2

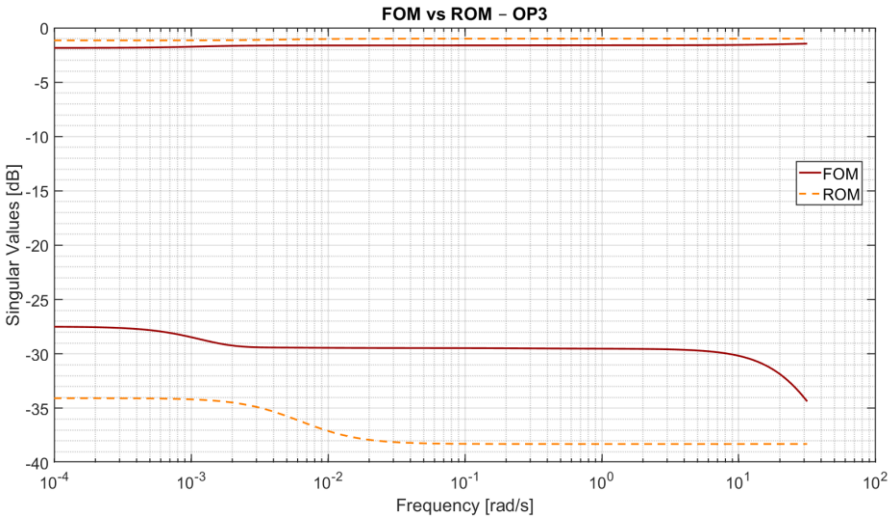


Figure 6.14: Singular values comparison between FOM and ROM at OP1.3

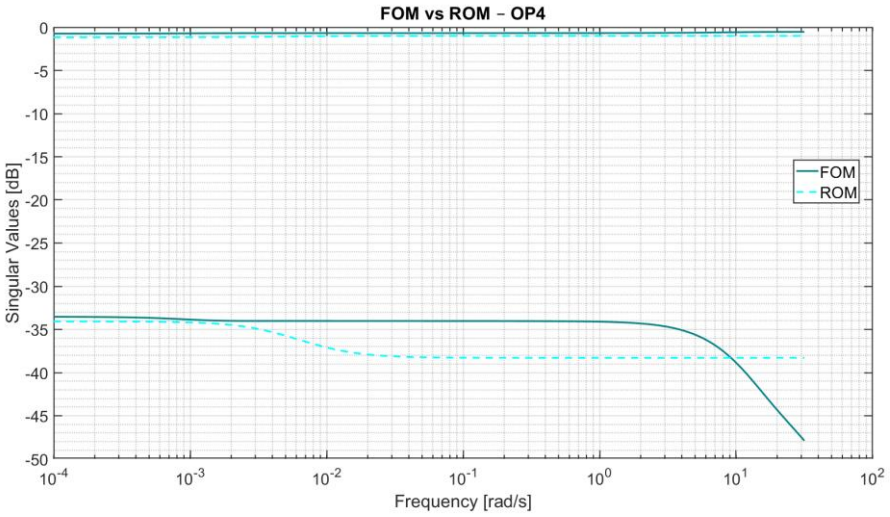


Figure 6.15: Singular values comparison between FOM and ROM at OP2.1

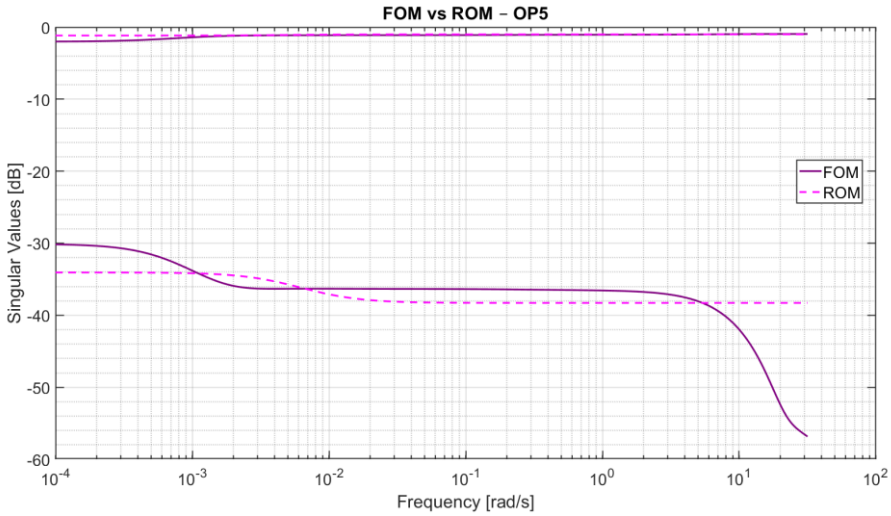


Figure 6.16: Singular values comparison between FOM and ROM at OP2.2

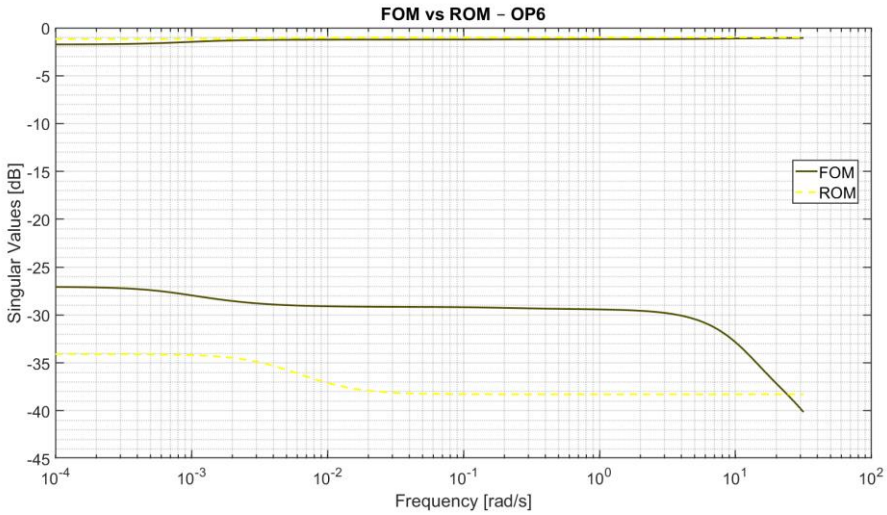


Figure 6.17: Singular values comparison between FOM and ROM at OP2.3

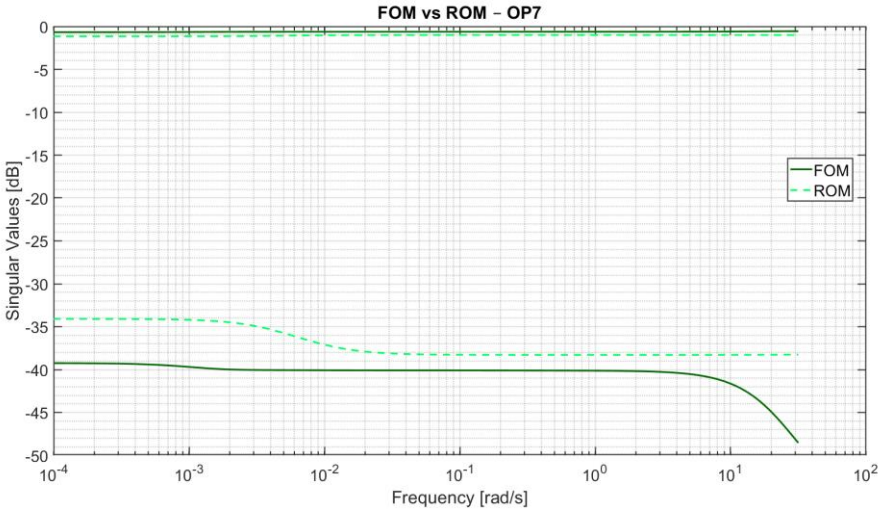


Figure 6.18: Singular values comparison between FOM and ROM at OP3.1

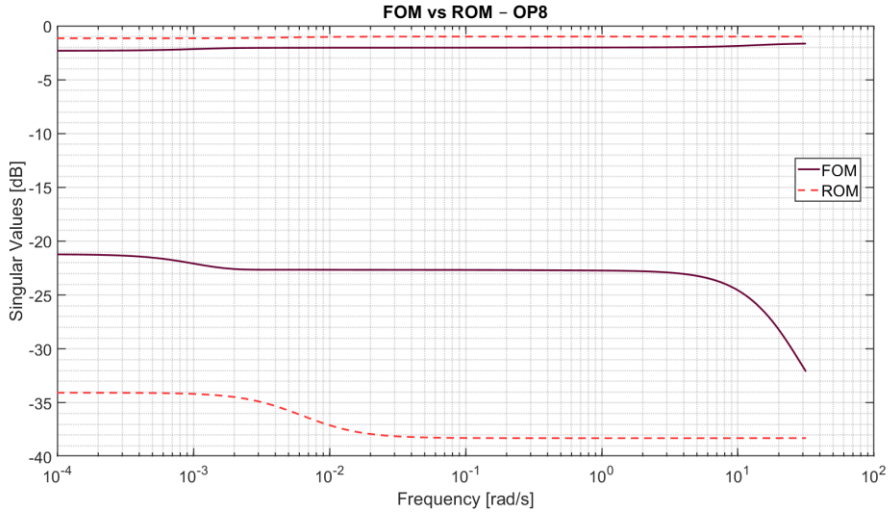


Figure 6.19: Singular values comparison between FOM and ROM at OP3.2

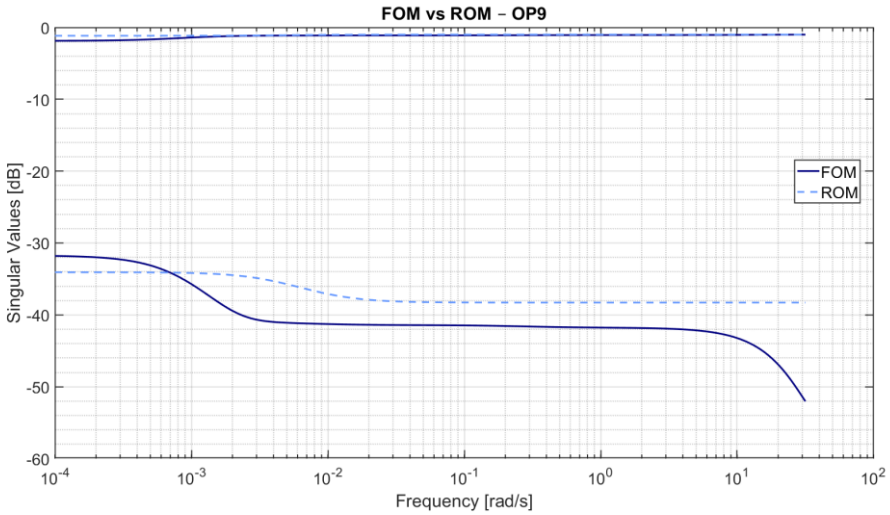


Figure 6.20: Singular values comparison between FOM and ROM at OP3.3

6.6.3 Comparative Results and Discussion

From the results about singular value plots, it clearly emerges that the reduced-order model is able to reproduce the dominant singular value with good accuracy across the entire frequency range of interest and for all OPs. The static gains are well preserved, and the slope of the dominant singular value curve follows that of the full-order model, indicating that the primary dynamic modes governing the system response are correctly captured by the reduction procedure. The reproduction of the second singular value is less accurate. Although the general trend is more or less maintained. In particular, the ROM does not perfectly replicate the magnitude evolution associated with the weaker dynamic channel of the system. Nevertheless, the discrepancy remains reasonable.

It is important to interpret this result in light of the adopted reduction strategy. The ROM relies on linear moment information averaged over the nine OPs. While this approach allows the model to capture the dominant dynamics across the entire operating domain, it does not explicitly incorporate nonlinear moment information. The second singular value is typically associated with secondary dynamic effects and cross-coupling phenomena, which are more sensitive to nonlinearities and operating-point variations. Therefore, it is reasonable that a reduction based solely on linear moment exhibits some limitations in reproducing this finer multivariable dynamics.

A potential improvement would consist in estimating and incorporating nonlinear moments information into the ROM structure, like in the SOFC SISO case. By enriching the ROM with nonlinear moment information, either computed analytically or estimated in a data-driven manner, it is expected that the accuracy in reproducing the secondary singular value would increase. This represents a natural extension of

the present work. Despite this limitation, the achieved performance must be evaluated in the context of the dimensionality reduction obtained. The full-order SOFC MIMO model has order 268, while the reduced one has order 12. Achieving a reduction of more than 95% while accurately reproducing the dominant singular value represents a significant result. The reduced model preserves the essential multivariable input-output structure and provides a compact representation suitable for control-oriented applications. Regarding the stability properties, linearized ROMs are simply stable because they have poles at the origin, so the ROM is simply stable, the eigenvalues are reported in Table E.8 in Appendix D. Overall, the order 12 ROM offers an experimentally acceptable compromise between complexity and accuracy. While further improvements are possible through nonlinear moment, the current results already demonstrate the effectiveness of the proposed nonlinear moment matching framework for high-dimensional multivariable models.

7 Conclusions and Future Developments

This work has presented and validated a nonlinear moment matching MOR framework specifically tailored for control-oriented applications in the field of power generation systems. The methodology has been applied to two representative and physically complex systems: a heat exchanger-turbine system and a SOFC system. In both cases, the objective was to construct reduced-order models capable of preserving the essential steady-state and dynamic input-output behavior of the original systems, while achieving a substantial reduction in dimensionality.

For the heat exchanger-turbine model, starting from a full-order nonlinear system of order 80, the reduction process has proven successful in both SISO and MIMO configurations. In the SISO case, a reduced model of order 12 provides an accurate approximation of the frequency response over the range of interest for control purposes. The Bode diagrams of the reduced model closely match those of the linearized full model in both magnitude and phase, and the static gain is preserved with satisfactory accuracy. The reduced model remains asymptotically stable, confirming the suitability of the adopted nonlinear structure for control-oriented modeling.

In the MIMO configuration, the reduction quality has been assessed through singular value analysis. A reduced model of order 12 achieves an excellent approximation of the singular values of the full-order model across the frequency band of interest. However, at this order, a pair of complex conjugate poles with positive real part appears in the reduced model. Although the frequency-domain approximation is

optimal in terms of singular values matching the presence of unstable poles makes this realization unsuitable for direct control implementation. Increasing the reduced order to 16 eliminates this instability while maintaining high accuracy in the singular value approximation. This confirms that the proposed framework can produce compact and accurate multivariables reduced models, if stability is explicitly verified. The reduction of the SOFC model, whose full-order formulation consists of 268 states, further demonstrates the scalability of the proposed approach. In the SISO configuration, a reduced model of order 16 accurately reproduces the Bode diagram of the linearized full model over a broad frequency range. Both magnitude and phase are well approximated, the static gain is preserved, and asymptotic stability is maintained. The mean square error remains low, while the dimensionality reduction exceeds 90%, highlighting the effectiveness of nonlinear moment matching for high-order models.

For the MIMO SOFC case, further investigation is required. The multivariable extension demands the computation of nonlinear moments for the reduced model, which entails a nonlinear Sylvester partial differential equation or developing an appropriate data-driven estimation strategy. The absence of a fully developed nonlinear moment computation framework in the multivariable setting currently limits the extension of the methodology. Addressing this issue represents a relevant and natural continuation of the research activity. Beyond the extension to the MIMO SOFC case, another important direction for future work concerns the practical control validation of the reduced models. In particular, the implementation and testing of an MPC scheme based on the developed reduced-order models would provide a direct assessment of their suitability for real-time advanced control applications.

Given the substantial reduction in model order achieved while preserving the dominant dynamic characteristics, the proposed reduced models are promising candidates for predictive control architectures, where computational efficiency and accurate dynamic representation are both essential. Testing MPC performance in terms of constraint handling, reference tracking, disturbance rejection, and robustness would allow a comprehensive evaluation of the control-oriented value of the proposed reduction framework. Overall, the results demonstrate that nonlinear moment matching combined with an appropriately structured reduced model and a careful selection of operating points provides a powerful and flexible tool for control-oriented model reduction. The methodology successfully bridges high-fidelity physical modeling and practical control implementation in complex energy systems. The achieved reductions in dimensionality, together with the preservation of dynamic and steady-state behavior, confirm the relevance of the approach for advanced control

applications and open promising perspectives for further theoretical and practical developments.

Bibliography

[1] P. Fritzson, *Principles of Object-Oriented Modeling and Simulation with Modelica*, 2nd ed. Hoboken, NJ, USA: Wiley, 2014.

[2] J. B. Rawlings and D. Q. Mayne, *Model Predictive Control: Theory and Design.* Madison, WI, USA: Nob Hill Publishing, 2017.

[3] A. Bemporad and M. Morari, "Robust model predictive control: A survey", *Automatica*, vol. 35, no. 3, pp. 407-427, 1999.

[4] B. C. Moore, "Principal component analysis in linear systems: Controllability, observability, and model reduction", *IEEE Trans. Autom. Control*, vol. 26, no 1, pp, 17-32, 1981.

[5] A. C. Antoulas, *Approximation of Large-Scale Dynamical Systems*. Philadelphia, PA, USA: SIAM, 2005.

[6] E. J. Grimme, "Krylov projection methods for model reduction", Ph.D. dissertation, Dept. Electr. Comput. Eng., Univ. Illinois at Urbana-Champaign, Urbana, IL, USA, 1997.

[7] A. C. Antoulas, D. C. Sorensen, and S. Gugercin, "A survey of model reduction methods for alrge-scale systems", in *Contemporary Mathematics,* vol. 280. Providence, RI, USA: AMS, 2001, pp. 193-219.

[8] A. C. Antoulas, "Model reduction of large-scale systems", SIAM News, vol. 43, no. 9, pp. 1-3, 2010.

[9] A. C. Ionita and A. C. Antoulas, "Data-driven parametrized model reduction in the Loewner framework", SIAM J, Sci. Comput., vol 36, no. 3, pp. A984-A1007, 2014.

[10] A. Astolfi, "Model reduction by moment matching for linear and nonlinear systems", IEEE Trans. Autom. Control, vol. 55, pp. 2321-2336, 2010.

[11] A. Moreschini, M. Scandella and T. Parisini, "Nonlinear Data-Driven Moment Matching in Reproducing Kernel Hilbert Spaces," *2024 European Control Conference (ECC)*, Stockholm, Sweden, 2024, pp. 3440-3445, doi: 10.23919/ECC64448.2024.10590737.

[12] A. Moreschini, M. Scandella, A. Astolfi and T. Parisini, "*Moment Matching by Kernel-Based Learning*", in IEEE Trans. Autom. Control, doi 10.1109/TAC 2025 3618165

[13] P. Brenner, S.Gugercin, and K. Willcox, "A survey of projection-based model reduction methods for parametric dynamical systems", *SIAM Review*, vol. 57, no. 4 pp. 483-531, 2015.

[14] R. W. Freund, "Krylov-subspace methods for reduced-order modeling in circuit simulation", *Journal of Computational and Applied Mathematics*, vol. 123, no. 1-2, pp. 395-421, 2000.

[15] P. Benner, J-R. Li, and T. Penzi, "Numerical solution of large-scale Lyapunov equations, and linear-quadratic optimal control problems, "*Numerical Linear Algebra with Applications*, vol. 15, no. 9, pp. 755-777, 2008.

[16] S. Gugercin and A. C. Antoulas, "A survey of model reduction by balanced truncation and some new results", *International Journal of Control*, vol. 77, no. 8, pp. 748-766, 2004.

[17] L. Ljung, *System Identification: Theory for the User*, 1999.

[18] R. Pintelon and J. Schoukens, *System Identification: A Frequency Domain Approach*, 2nd ed., 2012

[19] A. C. Antoulas, A. C. Ionita, and S. Lefteriu, "On the Loewner framework for model reduction", *SIAM Journal on Scientific Computing*, vol. 38, no. 5, pp. A3274-A3300, 2016.

[20] P. Van Overschee and B. De Moor, *Subspace Identification for Linear Systems: Theory---Implementation---Applications*, Boston, MAm USA: Kluwer Academic Publishers, 1996.

[21] G. Scarciotti and A. Astolfi, *Nonlinear Model reduction by Moment Matching*, Foundations and Trends in Systems and Control, vol. 4, no. 3-4, pp. 224-409, Aug. 2017, doi: 10.1561/2600000012.

[22] G. Scarciotti and A. Astolfi, "Data-driven model reduction by moment matching for linear and nonlinear systems", *Automatica*, vol. 79, pp. 340-351. May 2017, doi: 10.1016/j.automatica.2017.01.014

[23] P. Fritzson et al., "OpenModelica A free open-source environment for system modeling, simulation, and teaching," in Proc. *IEEE Conf. on Computer Aided Control System Design*, Munich, Germany, 2006, pp. 1588-1595.

[24] A. Pop et al., “OpenModelica: Modeling, Simulation, and Development Environment for Complex Systems,” *Mathematical and Computer Modelling of Dynamical Systems*, vol. 16, no. 4, pp. 327-351, 2010.

[25] T. Blochwitz et al., “The Functional Mock-up Interface for Tool Independent Exchange of Simulation Models,” in *Proc. 8th Int. Modelica Conf.*, Dresden, Germany, 2011, pp. 105-114.

[26] M. L. De Pascali, A. Donazzi, E. Martelli, and F. Casella, “A Control-Oriented Modelica 1-D Model of a Planar Solid-Oxide Fuel Cell for Oxy-Combustion Cycles,” in *Proceedings of the 2023 IEEE Conference on Control Technology and Applications (CCTA)*, Bridgetown, Barbados, Aug. 16-18, 2023, pp. 394-399, doi: 10.1109/CCTA54093.2023.10252534.

[27] *https://github.com/casella/ThermoPower.git*

[28] M. L. De Pascali and F. Casella, "Modeling Innovative Thermal Power Generation Systems for the Energy Transition with Modelica: an Open Source and Flexible Solution," *2024 IEEE Conference on Control Technology and Applications (CCTA)*, Newcastle upon Tyne, United Kingdom, 2024, pp. 533-538, doi: 10.1109/CCTA60707.2024.10666527.

[29] *https://github.com/looms-polimi/SOFCPoliMi.git*

Appendix

Appendix A

A.1 OpenModelica and Modelica Language Overview

This appendix introduces the software infrastructure and modeling paradigm of the models adopted in this thesis. In particular, it provides a concise but rigorous overview of the Modelica modeling language, the OpenModelica simulation environment, the underlying mathematical formulation, and the component-based and acausal modeling philosophy. This background is essential for interpreting the code reported in Appendix B, and for understanding how the models are constructed and simulated. Unlike traditional simulation environments based on imperative programming, Modelica is a declarative, equation-based language specifically designed for the modeling of complex physical systems. This feature is particularly well suited to multi-domain systems, such as the ones considered here, which

simultaneously involve fluid dynamics, heat transfer, thermodynamics, and energy conversions.

A.1.1 Declarative and Acausal Modeling

In Modelica, systems are described by equations rather than assignments. That is, the modeler specifies relations of the form

$$f(x, \dot{x}, z, u, \theta) = 0,$$

instead of writing explicit state-update equations of the form

$$\dot{x} = g(x, u).$$

This distinction is fundamental. In imperative languages, the causality (i.e., which variables are inputs and which are outputs) is fixed by the programmer. In Modelica, causality is inferred automatically by the compiler through symbolic manipulation. This acausal modeling approach provides several advantages:

1. Physical transparency: the equations correspond directly to physical laws such as mass balances, energy balances, and constitutive relations.
2. Modularity; components can be reused in different contexts without rewriting equations.
3. Automatic index reduction: high-index differential algebraic equations (DAEs) are transformed into solvable forms by the compiler.
4. Flexible interconnection: the same component can be used upstream or downstream without rewriting it.

This paradigm is crucial for large-scale systems such as heat exchanger-turbine or SOFC, where hundreds or thousands of equations interact in nontrivial ways.

A.1.2 Differential-Algebraic Equations

Most physical systems, modeled in Modelica do not naturally take the form of explicit ordinary differential equations (ODEs). Instead, they lead to DAEs:

$$F(\dot{x}(t),\ x(t),\ z(t),\ u(t),\ \theta) = 0, \quad \text{(a.1)}$$

where:

- $x(t) \in \mathbb{R}^n$ are dynamic states;
- $z(t) \in \mathbb{R}^m$ are algebraic variables;

- $u(t) \in \mathbb{R}^p$ are inputs;
- $\theta(t) \in \mathbb{R}^q$ are parameters.

OpenModelica automatically performs index reduction, symbolic differentiation, casualization, tearing, and equation sorting to convert the original model into a form suitable for numerical integration. This is essential for enabling the simulation of high-fidelity physical models without manual manipulation of the equations.

A.1.3 Component-Based Modeling

Modelica supports object-oriented physical modeling. Each physical component is represented by a class, which encapsulates:

- Parameters;
- Variables;
- Governing equations;
- Interfaces (or connectors).

For instance, a turbine is a class; a heat exchanger is a class. These components are then interconnected graphically or textually. Mathematically, this corresponds to assembling a global DAE system by concatenating the equations of each component and adding interconnection constraints. Let each component i be described by

$$F(\dot{x}_i, x_i, z_i, u_i, \theta_i) = 0.$$

The complete system is obtained by enforcing conservation laws at interfaces, equality constraints at connectors, and global parameters sharing. This yields a coupled system (a.1), where

$$x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix}.$$

This hierarchical construction is directly reflected in the OpenModelica code reported in Appendix B.

A.1.4 The OpenModelica Toolchain

OpenModelica is an open-source implementation of the Modelica language. It provides:

- A compiler for Modelica models;

- A symbolic manipulation engine;
- A code generator for the simulation code
- A ODE solver. The DAEs are turned into ODEs during code generation
- A graphical modeling environment;
- Scripting interfaces such as Python and OMJulia.

The typical workflow is:

1. Model definition: write or assemble the system using Modelica classes.
2. Flattening: the hierarchical model is transformed into a flat system of equations.
3. Symbolic processing: index reduction, simplification, and causalization.
4. Code generation: C/C++ code is generated for the solver.
5. Numerical simulation: time integration using a variable-step solver.
6. Data extraction: trajectories are exported for post-processing.

In this thesis, OpenModelica is used to simulate the full-order models, generate input-output datasets, and more in general to support data-driven model reduction.

A.1.5 Why Modelica for this Thesis

The choice of Modelica and OpenModelica is motivated by several factors:

1. Physical fidelity: the libraries used in this work (BoilerAndTurbine, and [29]) include real-fluid thermodynamics, phase-change effects, nonlinear transportation laws, and spatially distributed models.
2. Multi-physics integration: These models simultaneously involve fluid mechanics, heat transfer, thermodynamics, and energy conversion. Modelica is explicitly designed for such multi-domain problems.
3. Large-scale dynamics: the FV discretization of various components such as heat exchanger or fuel cell channels leads to many states. This makes the systems a natural benchmark for large-scale model reduction.
4. Direct link to industrial practice: Modelica is widely used in industrial contexts, especially in energy systems modeling. This increases the practical relevance of the proposed work.

A.2 Mathematical Structure of Modelica Models

A.2.1 From Component-Based Modeling to Global Systems

A Modelica model is inherently hierarchical and component-based. At the highest level, the user assembles a system by interconnecting pre-defined components. Each component contributes to its own set of equations, variables, and parameters. Let the

overall system consist of N_c components. Each component i is described by a local equation set of the form

$$F_i(\dot{x}_i(t), x_i(t), z_i(t), u_i(t), \theta_i) = 0,$$

where:

- $x_i(t) \in \mathbb{R}^{n_i}$ are the dynamic state variables,
- $z_i(t) \in \mathbb{R}^{m_i}$ are algebraic variables,
- $u_i(t) \in \mathbb{R}^{p_i}$ are external inputs,
- $\theta_i(t) \in \mathbb{R}^{q_i}$ are constant parameters.

The full system is obtained by concatenating all component equations and adding interconnection constraints enforced through Modelica connectors. Formally, the global model is described by (a.1) with

$$x = \begin{bmatrix} x_1 \\ \vdots \\ x_{N_c} \end{bmatrix}, \quad z = \begin{bmatrix} z_1 \\ \vdots \\ z_{N_c} \end{bmatrix}.$$

This formulation highlights a crucial aspect: Modelica models are not written in state-space form, but rather as large implicit systems of equations.

A.2.2 Model Flattening

Before numerical simulation can be performed, the hierarchical Modelica model must be transformed into a flat representation. Flattening is the process by which:

- All class hierarchies are expanded;
- All extends and redeclare constructs are resolved;
- All parameter values are propagated;
- All component instances are made explicit.

The result of flattening is a single system of equations (a.1) with no remaining hierarchy.

A.2.3 Causalization and Equation Sorting

Once flattened, the system of equations must be casualized, i.e., rewritten in a form suitable for numerical integration. Given a generic implicit DAE system (a.1), the compiler must determine which variables are treated as states, which variables are

algebraic, which equations define derivatives, which equations define algebraic variables. This process involves:

1. Structural analysis: identification of solvable equation-variable matchings.
2. Equation sorting: reordering equations to respect computational dependencies.
3. Tearing: partitioning the system into smaller subsystems to reduce computational cost.

A.2.4 Index of Differential-Algebraic Equations

Many Modelica models lead to high-index DAEs. The index of a DAE informally measures how many times algebraic equations must be differentiated to obtain an explicit ODE form. A typical index-1 DAE has the form

$$\dot{x} = f(x, z, u), \quad \text{(a.2)}$$
$$0 = g(x, z, u).$$

However, physical models often lead to index-2 or index-3 systems due to conservation laws, kinematic constraints, and thermodynamic equilibrium conditions.

A.2.5 Index Reduction

OpenModelica automatically performs index reduction to transform the original system into an equivalent index-1 DAE. This typically involves differentiating algebraic constraints, introducing auxiliary variables, and eliminating redundant equations. Formally, the original system (a.1) is transformed into

$$F^*(\dot{x}, x, z^*, u, \theta) = 0,$$

Where z^* includes additional variables introduced during differentiation. Index reduction is crucial for numerical stability, solver convergence, and accurate time integration.

A.2.6 From DAE to ODE: Numerical Integration

After index reduction and casualization, the system is typically transformed into state-space ODE form. This system is then integrated using implicit solvers such as:

- DASSL;
- IDA;
- CVODE.

These solvers internally perform Newton iteration, Jacobian evaluations, and linear system solves. The computational cost scales with the number of states. From the integration, the results are simulated trajectories of inputs, outputs, and state variables. So, the simulation of the system.

Appendix B

B.1 System Class

The *System* class provides a global simulation context shared by all components of the heat exchanger-turbine model. It specifies the ambient pressure and temperature, and how the model is initialized and simulated. In particular, the following aspects are controlled:

- Initialization strategy: the parameter *initOpt* selects the initialization mode. This is critical for large-scale nonlinear DAE systems, as inappropriate initialization can lead to solver failure or nonphysical transients;
- Flow reversal allowance: the Boolean flag $allowFlowReversal \in \{true, false\}$ determines whether bidirectional flow is permitted at fluid connectors. In the present configuration, flow reversal is allowed, enabling robust simulation under transient operating conditions.

These options influence the algebraic structure of the system equations generated by the Modelica compiler and directly affect numerical conditioning and convergence.
The System object does not contain state variables or dynamic equations. Instead, it acts as a shared parameter container and configuration hub. All major components of the heat exchanger-turbine model namely:

- The turbine;
- The heat exchanger;
- The boundary mass flow sources;
- The pressure sinks.

declare an outer *system* reference, which links them to the global system definition. This mechanism ensures that all components use consistent ambient references; initialization is performed coherently across subsystems, and moreover, that the resulting global model forms a single, well-posed differential-algebraic system.

B.1.1 System Class: OpenModelica Code

This subsection reports the OpenModelica implementation of the global *System* class used throughout the heat exchanger-turbine model. As discussed in Appendix B.1,

this class does not represent a physical subsystem, but rather provides a shared infrastructure for simulation settings, ambient conditions, and initialization policies. The complete OpenModelica code of the *System* class is reported below

```
within BoilerAndTurbine;
model System
parameter Boolean allowFlowReversal=true annotation(. . .);
parameter BoilerAndTurbine.Choices.Init.Options initOpt=
BoilerAndTurbine.Choices.Init.Options.steadyState annotation(. . .);
parameter BoilerAndTurbine.Types.Pressure p_amb = 101325
annotation(. . .);
parameter BoilerAndTurbine.Types.Temperature T_amb = 294.15
annotation(. . . );
parameter BoilerAndTurbine.Types.Temperature T_wb = 288.15
annotation(. . . );
Modelica.Blocks.Sources.RealExpression Amb_T(y=21 + 273.15)
annotation (Placement(transformation(extent={{-32,18},{-12,38}})));
annotation(. . . );
end System;
```

B.2 Turbine Model

This appendix reports the complete OpenModelica implementation references of the turbine component used in the heat exchanger-turbine system. The turbine is modeled according to Stodola's law and isentropic efficiency relations, as described in Section 4.2.3.1.

B.2.1 Inheritance Structure of Turbine Model

The turbine component used in this work is implemented through a multi-level inheritance hierarchy, which separates physical modeling assumptions from specialization choices. The inheritance chain is:

Turbine → *TurbineStodola* → *TurbineBase*

Each level has a well-defined modeling role. *TurbineBase* is an abstract (partial) model that defines the core physical structure of a generic turbomachine. It contains:

- Thermodynamic state variables;
- Mass and energy balance equations;
- Isentropic efficiency formulation;
- Mechanical and electrical power computation;
- Fluid connectors as *FlangeA, FlangeB.*

Crucially, *TurbineBase* does not specify the mass flow law. The design choice allows different turbine flow models to be implemented without duplicating the rest of the model. Because it is declared as *partial*, it cannot be instantiated directly. *TurbineStodola* extends *TurbineBase* by closing the model through the definition of the mass flow rate

equation using Stodola's law defined in Section 4.2.3.1. The top-level *Turbine* class is lightweight wrapper that extends *TurbineStodola*, fixes modeling options (e.g., entropy computation, transport properties) assigns numerical values to parameters such as:

- K_t;
- Nominal efficiency;
- Nominal pressure and temperatures;
- Initialization values.

This class does not introduce new equations, its purpose is purely parametric and configurational.

B.2.2 Homotopy-Based Initialization

The turbine model contains strongly nonlinear algebraic relations, including square roots, thermodynamic property inversions, and couple pressure-enthalpy relations. When embedded in a large DAE system (as is the case of heat exchanger-turbine and SOFC models), these nonlinearities can cause initialization failures or poor convergence of the nonlinear solver. To mitigate this issue, the model uses homotopy-based initialization. Homotopy is a numerical technique in which a difficult equation is replaced by a family of equations parametrized by a variable $\lambda \in [0.1]$:

$$F(x) = 0 \rightarrow (1-\lambda)F_{simple}(x) + \lambda F_{full}(x) = 0$$

- For $\lambda = 0$ the solver sees a simplified problem, easy to solve;
- For $\lambda = 1$ the original, full nonlinear problem is recovered.

The solver gradually increases λ, tracking the solution from the simple system to the full one. In the turbine implementation, homotopy is applied to the mass flow equation and the thermal power computation. For example, Stodola's law is written as:

```
w = homotopy(
        Kt*sqrt(pin*rho)sqrtReg(1 - (1/PR)^2),
        wnom*pin/pin_start
        )
```

here:

- The full expression is the nonlinear Stodola's law;

- The simplified expression is a linearized relation based on nominal conditions.

This guarantees that the model has a well-defined initial solution, and convergence is improved even for stiff or poorly scaled systems. An important thing to observe is that homotopy does not change the physical model; it only affects how the numerical solver reaches the solution. Once initialization is complete, the homotopy parameter is set to $\lambda = 1$, and the simulation proceeds using the full equations.

B.2.3 Turbine Class: OpenModelica Code

The turbine is implemented as a parametrized component extending a base turbomachine model. It exposes fluid inlet and outlet ports and provides electrical power and turbine inlet temperature as outputs. The *Turbine* and *TurbineBase* classes are inspired by the Thermopower library [27], whose validation is treated in [28].

B.3. Heat Exchanger Model

This appendix section documents the OpenModelica implementation of the counter-current heat exchanger used in the heat exchanger-turbine system. In contrast to the turbine component, the heat exchanger model relies much more on component composition than deep inheritance chains. Its complexity arises from the interconnection of multiple submodels representing fluid channels, thermal walls, and topology constraints. The physical modeling assumptions and governing equations have been discussed in detail in Section 4.2.3.2

B.3.1 Overall Class Structure

The main heat exchanger component instantiated in the system is the class *HX* which extends the class *HeatExchangerCounterCurrent*. This class comes from the *BoilerAndTurbine.HeatExchanger* package. Unlike the turbine, the heat exchanger does not rely on a strong inheritance hierarchy. Instead, inheritance is used once, to select the counter-current configuration, and most of the model complexity is introduced through composition of subcomponents. At high level, the structure is:

HX
└── *HeatExchangerCounterCurrent*
  ├── *hotside : Flow1DFV*
  ├── *coldside : Flow1DFV*
  ├── *wall : MetalWallFV*
  ├── *topology : HeatExchangerTopologyFV*
  └── *auxiliary initializers*

The design reflects the physical decomposition of the device into interacting subsystems.

B.3.2 Finite-Volume Fluid Channels

The hot and cold fluid streams are modeled using instances of the *Flow1DFV* class, which implements a one-dimensional FV discretization of compressible flow with heat transfer. Two separated instances are used:

- *hotside:* representing the hot fluid channel;
- *Coldside:* representing the cold fluid channel.

Each *Flow1DFV* instance internally contains:

- N finite volumes;
- Thermodynamic state variables such as pressure, temperature, enthalpy, composition;
- Mass and energy balance equations;
- Convective transport between neighboring volumes;
- Heat exchange terms with the wall.

From a software perspective, *Flow1DFV* is not further specialized by inheritance in this model. Instead, different behaviors are selected through replaceable models and parameters. This design allows to reuse of the same FV formulation for both streams, consistent treatment of thermodynamic on both sides, and a scalable system order through the parameter N. The full OpenModelica code for *Flow1DFV* instances, including parameterization for the hot and cold sides, is reported in Appendix B.3.6.

B.3.3 Wall Model

Thermal storage and conduction between the two fluid streams are captured by the *MetalWallFV* component. This class discretized the separating wall into N thermal volumes and introduces additional dynamic states corresponding to wall temperature, moreover couples to both fluid channels via convective heat transfer ports. The wall model is composed, not inherited, into the heat exchanger. Its presence is optional and controlled through configuration parameters (e.g., *WallRes*). From a modeling standpoint, this choice introduces thermal inertia, and increases system stiffness, in general significantly contributes to the overall system order. The internal equations of *MetalWallFV* are not modified in the *HX* class, but only parametrized.

B.3.4 Counter-Current Topology

The counter-current configuration is enforced through a dedicated topology component *HeatExchangerTopologyFV*. This object does not introduce additional states; its role is purely structural: it defines how wall ports and fluid volumes are interconnected. In particular, it ensures that the hot and the cold streams flow in opposite directions, and the correct wall segments are coupled to the corresponding fluid volumes.

B.3.5 Initialization and Homotopy Usage

Similarly to the turbine model, the heat exchanger implementation makes use of homotopy-based initialization, although in a more localized manner. Homotopy is primarily employed in the *Flow1DFV* channel models, and auxiliary *HomotopyInitializer* components connected to the fluid inlets. These initializers provide consistent starting values for pressure, temperature, composition, and enthalpy. The conceptual meaning of homotopy has already been discussed in Appendix B.2.2 and is not repeated here.

B.3.6 Heat Exchanger Class: OpenModelica Code

The *HX*, *Flow1DFV*, *MetallWallFV*, and *HeatExchangerTopologyFV* classes are inspired by the Thermopower library [27], whose validation is treated in [28]

B.4 Boundary Components

This section documents the boundary components used to impose inlet and outlet conditions in the heat exchanger-turbine model. These components do not introduce additional dynamic states but play a crucial role in defining the system's inputs, boundary constraints, and algebraic coupling structure. The boundary components considered are *SourceIdealMassFlow* and *IdealSinkPressure*, both are instantiated multiple times in the top-level structure. Their complete OpenModelica implementations are reported in this appendix.

B.4.1 Boundary Components Class: OpenModellica Code

The complete OpenModelica code of the *SourceIdealMassFlow* class is reported below

```
1.    within BoilerAndTurbine.Components.Sources;
2.    model SourceIdealMassFlow
3.    parameter Integer nX = 10 "Number of species in the mixture";
4.    parameter BoilerAndTurbine.Types.Pressure p_start=27.5e5 "";
5.    parameter BoilerAndTurbine.Types.Temperature T_start=920 "";
6.    parameter BoilerAndTurbine.Types.MassFraction X_start[nX] "";
```

```
7.    parameter BoilerAndTurbine.Types.Density rho_start "";
8.    Replaceable model Medium =
9.    BoilerAndTurbine.Media.MainClasses.SOS_CO2.SOS10Components;
10.   Medium fluid( T_start=
11.   T_start, p_start=p_start, X_start=X_start, rho_start=rho_start)
12.   BoilerAndTurbine.Interfaces.FlangeB flange(
13.   nXi=nX,
14.   nC=0,
15.   w(max=0)) annotation(. . .);
16.   Modelica.Blocks.Interfaces.RealInput w annotation(. . .);
17.   Modelica.Blocks.Interfaces.RealInput X[nX] annotation(. . .);
18.   Modelica.Blocks.Interfaces.RealInput Tset annotation(. . .);
19.   equation
20.   fluid.T = Tset;
21.   fluid.p = flange.p;
22.   fluid.Xi = X "";
23.   flange.w = -w;
24.   flange.Xi = fluid.X;
25.   flange.h = fluid.h;
26.   annotation(. . .);
27.   end SourceIdealMassFlow;
```

The complete OpenModelica code of the *IdealSinkPressure* class is reported below

```
1.    within BoilerAndTurbine.Components.Sources;
2.    model IdealSinkPressure
3.    parameter Integer nX = 10 "Number of species in the mixture";
4.    BoilerAndTurbine.Interfaces.FlangeB flange(
5.    nXi=nX,
6.    nC=0,
7.    w(min=0)) annotation(. . .);
8.    parameter BoilerAndTurbine.Types.Pressure p;
9.    equation
10.   flange.p = p;
11.   flange.Xi = zeros(nX);
12.   flange.h = 0;
13.   annotation(. . .);
14.   end IdealSinkPressure;
```

B.5 Top-Level Interconnections

This appendix section reports the top-level OpenModelica class used to assemble the complete heat exchanger-turbine model by interconnecting the previously described components. The class does not introduce new physical models or equations; its purpose is to define structural composition of the system and the signal connections among subsystems.

B.5.1 BaseTest Model

The overall system is instantiated in the class *BaseTest*, which serves as the simulation-level container of the model. The following components are instantiated:

- One counter-current heat exchanger *HX;*
- One compressible-flow turbine *Turbine;*
- Two inlet boundary sources *SourceIdealMassFlow;*
- Two outlet pressure sinks *IdealSinkPressure;*
- One global *System* object;
- Signal sources defining time-dependent inputs.

No additional state variables are introduced at this level.

B.5.2 Structural Interconnections

The interconnection structure follows the physical flow path of the working fluid:

- Hot stream *SourceIdealMassFlow* → *HX* → *IdealSinkPressure;*
- Cold stream *SourceIdealMassFlow1* → *HX* → *IdealSinkPressure1.*

The turbine inlet is directly connected to the cold side outlet of the heat exchanger. Signal connections are used to prescribe mass flow rates, inlet temperatures, and fluid composition; all thermodynamic variables are exchanged through OpenModelica *FlangeA* and *FlangeB* connectors.

B.5.3 External Inputs and Outputs

The global system inputs are

- Hot side mass flow rate;
- Cold side mass flow rate;
- Hot side inlet temperature;
- Cold side inlet temperature;
- Hot side composition;
- Cold side composition.

The global system outputs of interest are:

- Electrical power produced by the turbine, named *Pe;*
- Turbine inlet temperature, named *TIT;*

B.5.4 BaseTest Class: OpenModelica Code

The complete OpenModelica code of the *BaseTest* class is reported below

```
1.    within BoilerAndTurbine.Tests;
2.    model BaseTest
3.    extends Modelica.Icons.Example;
4.    ParametrizedComponents.HX hX annotation(. . .);
5.    BoilerAndTurbine.Components.Sources.SourceIdealMassFlow
6.    sourceIdealMassFlow(
7.    nX=5,
8.    T_start(displayUnit="K") = 800,
9.    X_start={0.08,0.9,0.01,0.006,0.004},
```

```
10.   rho_start=5,
11.   redeclare model Medium =
12.   BoilerAndTurbine.Media.MainClasses.SOS_CO2.SOS5Components,
13.   p_start=120000)
14.   Annotation(. . .);
15.   BoilerAndTurbine.Components.Sources.IdealSinkPressure
16.   idealSinkPressure(
17.   nX=5,
18.   P=120000) annotation(. . .);
19.   BoilerAndTurbine.Components.Sources.SourceIdealMassFlow
20.   sourceIdealMassFlow1(
21.   nX=5,
22.   redeclare model Medium =
23.   BoilerAndTurbine.Media.MainClasses.SOS_CO2.SOS5Components,
24.   T_start(displayUnit="K") = 370,
25.   X_start={0.002,0.97,0.01,0.01,0.008},
26.   rho_start=5) annotation(. . .);
27.   BoilerAndTurbine.Components.Sources.IdealSinkPressure
28.   idealSinkPressure1(
29.   nX=5,
30.   p=120000) annotation(. . .);
31.   annotation(. . .);
32.   Modelica.Blocks.Sources.Constant Thot(k = 800) annotation(. . .);
33.   Modelica.Blocks.Sources.Constant Xhot[5](k = {0.08, 0.9, 0.01,
34.   0.006, 34.  0.004}) annotation(. . .);
35.   Modelica.Blocks.Sources.Constant Tcold(k = 370) annotation(. . .);
36.   Modelica.Blocks.Sources.Constant Xcold[5](k = {0.002, 0.97, 0.01,
37.   0.01, 0.008}) annotation(. . .);
38.   Modelica.Blocks.Sources.RealExpression wHot(y=if time < 5 then 100
39.   else 100) annotation(. . .);
40.   Modelica.Blocks.Sources.RealExpression wCold(y=if time < 10 then 45
41.   else 45) annotation(. . .);
42.   inner BoilerAndTurbine.System system annotation(. . .);
43.   ParametrizedComponents.Turbine turbine annotation(. . .);
44.   equation
45.   connect(sourceIdealMassFlow.flange, hX.inhot annotation(. . .);
46.   connect(sourceIdealMassFlow1.flange, hX.incold) annotation(. . .);
47.   connect(Xhot.y, sourceIdealMassFlow.X) annotation(. . .);
48.   connect(Thot.y, sourceIdealMassFlow.Tset) annotation(. . .);
49.   connect(Tcold.y, sourceIdealMassFlow1.Tset) annotation(. . .);
50.   connect(Xcold.y, sourceIdealMassFlow1.X) annotation(. . .);
51.   connect(hX.outhot, idealSinkPressure.flange) annotation(. . .);
52.   connect(turbine.outlet, idealSinkPressure1.flange)
53.   annotation(. . .);
54.   connect(turbine.inlet, hX.outcold) annotation(. . .);
55.   connect(TIT_ref.y, add.u1) annotation(. . .);
56.   connect(turbine.TIT, add.u2) annotation(. . .);
57.   connect(add.y, PI.u) annotation(. . .);
58.   connect(wHot.y, sourceIdealMassFlow.w) annotation(. . .);
59.   annotation(. . .);
60.   end BaseTest;
```

Appendix C

This appendix introduces the signal generator employed in the thesis to excite the full-order models and generate data for MOR. The signal generator is implemented as a standalone OpenModelica model and is based on a low-dimensional autonomous linear dynamical system. The theoretical role of the signal generator, including its interpretation in terms of moment matching, interpolation points, Sylvester equations, and invariant manifolds, is discussed extensively in Chapter 3, and Chapter 5. The purpose of this appendix is to document the implementation of the generator.

C.1 Signal Generator Class: OpenModelica Code

The complete OpenModelica code of the *SignalGenerator* class is reported below with an example of a signal generator of order two.

```
1.    model signal_generator
2.    parameter Real S[2,2] = [0,0.1;-0.1,0]
3.    parameter Real L[1,2] = [1,0]
4.    parameter Real y0 = 2.7 "output offset"
5.    parameter Real amp_rel = 0.0035 "relative amplitude"
5.    parameter Real x1_start = 1 "intial value of state x1"
6.    parameter Real x2_start = 0 "initial value oof state x2"
7.    Real x[2](start = {x1_start, x2_start}, fixed = {true, true});
8.    Real y_rel;
9.    Real y;
10.   Modelica.Blocks.Interfaces.RealOutput y_out annotation(. . .);
11.   Modelica.Blocks.Interfaces.RealOutput x_out1 annotation(. . .);
12.   Modelica.Blocks.Interfaces.RealOutput x_out2 annotation(. . .);
13.   equation
14.   der(x) = S*x;
15.   y_rel = (L*x)[1];
16.   y = y_0*(1+amp_rel*y_rel);
17.   y_out = y;
18.   x_out1 = x[1];
19.   x_out2 = x[2];
20.   end signal_generator_1match;
```

Appendix D

D.1 SISO Reduction Algorithm: MATLAB Code

The complete MATLAB code that implements the SISO reduction algorithm for heat exchanger-turbine is reported below

```
1.    %% Section 0: user damping parameters (init)
2.    alpha_damp = 0.025; % scalar damping (init)
3.    beta_damp = 0.75; % resonance damping (init)
4.    %% Section 1: Full Order Models (FOMs)
```

```
sys = ss(A_full_1,B_full_1,C_full_1,D_full_1);
sys1 = ss(A_full_2,B_full_2,C_full_2,D_full_2);
sys2 = ss(A_full_3,B_full_3,C_full_3,D_full_3);
% Ensure D matrices are zero
sys.D = 0;
sys1.D = 0;
sys2.D = 0;
%% Section 2: Datasets preprocessing (steady-state)
t_out = exportedVariables1006match.time; % time vector (example)
tSS = 200; fs = 5;
idx = find(t_out > tSS,1,'first');
omega_full_1 = [exportedVariables1006match.wcoldx_out1, ...
exportedVariables1006match.wcoldx_out2, ...
exportedVariables1006match.wcoldx_out3, ...
exportedVariables1006match.wcoldx_out4, ...
exportedVariables1006match.wcoldx_out5, ...
exportedVariables1006match.wcoldx_out6, ...
exportedVariables1006match.wcoldx_out7, ...
exportedVariables1006match.wcoldx_out8, ...
exportedVariables1006match.wcoldx_out9, ...
exportedVariables1006match.wcoldx_out10, ...
exportedVariables1006match.wcoldx_out11, ...
exportedVariables1006match.wcoldx_out12]';
omega_full_2 = [exportedVariables306match.wcoldx_out1, ...
exportedVariables306match.wcoldx_out2, ...
exportedVariables306match.wcoldx_out3, ...
exportedVariables306match.wcoldx_out4, ...
exportedVariables306match.wcoldx_out5, ...
exportedVariables306match.wcoldx_out6, ...
exportedVariables306match.wcoldx_out7, ...
exportedVariables306match.wcoldx_out8, ...
exportedVariables306match.wcoldx_out9, ...
exportedVariables306match.wcoldx_out10, ...
exportedVariables306match.wcoldx_out11, ...
exportedVariables306match.wcoldx_out12]';
y_full_1 = exportedVariables1006match.turbineTIT;
y_full_2 = exportedVariables306match.turbineTIT;
u_full_1 = exportedVariables1006match.wcoldy_out;
u_full_2 = exportedVariables306match.wcoldy_out;
% subsample steady-state
omega_1 = omega_full_1(:, idx:fs:end);
omega_2 = omega_full_2(:, idx:fs:end);
y_1 = y_full_1(idx:fs:end)/scale;
y_2 = y_full_2(idx:fs:end)/scale;
u_1 = u_full_1(idx:fs:end);
u_2 = u_full_2(idx:fs:end);
% ensure column vectors for u datasets
u_1 = u_1(:);
u_2 = u_2(:);
% combine
omega_all = [omega_1, omega_2];
y_all = [y_1(:); y_2(:)];
u_all = [u_1(:); u_2(:)];
```

```
%% Section 3: Nonlinear dictionary and weight estimation
dictionaryOfFunc = {
@(z) z(1);
@(z) 1
};
omegaNL_all = omega_all.'; % each row is a sample
windowSize = size(omegaNL_all,1);
nlValFromZeta_all = zeros(windowSize, length(dictionaryOfFunc));
for i = 1:windowSize
nlValFromZeta_all(i,:)= modred.Internal
.computeNonlinearityFromState(real(omegaNL_all(i,:)),
 dictionaryOfFunc);
end
yNL_all = y_all(:);
if length(yNL_all) ~= windowSize
yNL_all = yNL_all(1:windowSize);
end
lambda = 1e-6; % ridge
weights=(nlValFromZeta_all'*nlValFromZeta_all+ 77.lambda*
eye(length(dictionaryOfFunc))) \ (nlValFromZeta_all' * yNL_all);
weights = weights(:);
%% Section 4: ROM components damping build (nominal)
% feature vector (phi)
phi_col=@(xi)reshape(cell2mat(cellfun(@(f)f(xi),
dictionaryOfFunc, 'UniformOutput', false)), [], 1);
% compute nominal S (interpolation-based)
S0 = computeSFromInterpolationPoints([0.1 0.1 0.01 0.01 1 1]);
nu = size(S0,1); % reduced state dimension (rows)
% damping D = alpha*I + beta*S0 (here use init values)
D_rayleigh = alpha_damp * eye(nu) + beta_damp * S0;
S_damped = S0 - D_rayleigh;
L = ones(1,nu);
s_fun = @(xi) S_damped * xi; % includes damping (nominal)
l_fun = @(xi) L * xi;
% delta default builder with state-input product
delta_default = @(xi,u) diag(0.5 + 0.5*tanh(2*xi(:)) + 0.2*(xi(:).^2)
+ 0.01*(xi(:).*sum(u(:))));
% reducedModel struct
reducedModel.S = S_damped;
reducedModel.S0 = S0;
reducedModel.D_rayleigh = D_rayleigh;
reducedModel.L = L;
reducedModel.weights = weights;
reducedModel.dictionaryOfFunc = dictionaryOfFunc;
reducedModel.s = s_fun;
reducedModel.l = l_fun;
reducedModel.phi = phi_col;
reducedModel.output = @(xi) (weights' * phi_col(xi));
reducedModel.delta = delta_default;
%% Section 5: Operating points (initial guess)
xi_eq1 = 0.09 * ones(nu,1);
xi_eq2 = 0.09 * ones(nu,1);
xi_eq3 = 0.09 * ones(nu,1);
```

```
u_eq1 = mean(u_1);
u_eq2 = mean(u_2);
u_eq3 = 75;
%% Section 6: Parametric delta(xi,u) to optimize (5 params)
% build_delta returns a function handle delta(xi,u)
build_delta = @(p) ( @(xi,u) diag( ...
p(1) + ...
p(2) .* tanh( p(3) .* real(xi(:)) ) + ...
p(4) .* (real(xi(:)).^2) + ...
p(5) .* ( real(xi(:)) .* sum(real(u(:))) ) ...
) );
p0_delta = [1.0, 0.3, 2.0, 0.05, 0.01];
%% Section 7: Precompute FOM FRFs
w_opt = 1i * logspace(-4, 0.5, 1000); % complex freq vector (reduced
points)
FRF_f1_pre = squeeze(freqresp(sys, w_opt));
FRF_f2_pre = squeeze(freqresp(sys1, w_opt));
FRF_f3_pre = squeeze(freqresp(sys2, w_opt));
% convert to magnitude (dB) and wrapped phase (deg) for objective
mag_FOM1 = 20*log10(abs(FRF_f1_pre)+eps);
ph_FOM1 = wrapTo180(rad2deg(unwrap(angle(FRF_f1_pre))));
mag_FOM2 = 20*log10(abs(FRF_f2_pre)+eps);
ph_FOM2 = wrapTo180(rad2deg(unwrap(angle(FRF_f2_pre))));
mag_FOM3 = 20*log10(abs(FRF_f3_pre)+eps);
ph_FOM3 = wrapTo180(rad2deg(unwrap(angle(FRF_f3_pre))));
%% Section 8: Delta( p(1:5) ) + xi + alpha + beta optimization
% full parameter vector: [alpha_damp; beta_damp; p_delta(5); xi1(nu);
 xi2(nu); xi3(nu)]
p0_all = [alpha_damp; beta_damp; p0_delta(:); xi_eq1(:); xi_eq2(:);
xi_eq3(:)];
% objective wrapper (fast: compares to precomputed mag/phase)
objfun_all=@(p_all)bode_fit_obj_3op_fast_with_u(p_all,build_delta,
reducedModel, ...
u_eq1, u_eq2, u_eq3, ...
mag_FOM1, ph_FOM1, mag_FOM2, ph_FOM2, mag_FOM3, ph_FOM3, w_opt);
opts=optimset('Display','iter','TolX',1e-6,'TolFun',1e6,
'MaxIter',35000,'MaxFunEvals',45000);
fprintf("\n=== START DELTA(5 params) + XI + ALPHA/BETA
 OPTIMIZATION (3 OPs) ===\n");
[p_opt_all, fval_opt] = fminsearch(objfun_all, p0_all, opts);
fprintf("Done: fval = %.6g\n", fval_opt);
% Extract optimized
% p_opt_all structure: [alpha; beta; p_delta(1:5); xi1(nu); xi2(nu);
xi3(nu)]
alpha_damp_opt = p_opt_all(1);
beta_damp_opt = p_opt_all(2);
p_delta_opt = p_opt_all(3:7);
% compute nu and extract xi segments (usa nu da S0 per coerenza)
nu_check = size(S0,1);
xi1_idx = 8 : 7+nu_check;
xi2_idx = 8+nu_check : 7+2*nu_check;
xi3_idx = 8+2*nu_check : 7+3*nu_check;
xi_eq1_opt = p_opt_all(xi1_idx);
```

```
xi_eq2_opt = p_opt_all(xi2_idx);
xi_eq3_opt = p_opt_all(xi3_idx);
delta_fun_opt = build_delta(p_delta_opt);
reducedModel.delta = delta_fun_opt;
%% Section 9: Rebuild Rayleigh damping and linearize
 ROM around optimized OPs
% rebuilt D_rayleigh and S_damped with optimized parameters
D_rayleigh = alpha_damp_opt * eye(nu) + beta_damp_opt * S0;
S_damped = S0 - D_rayleigh;
% update reducedModel.s
reducedModel.S = S_damped;
reducedModel.D_rayleigh = D_rayleigh;
reducedModel.s = @(xi) S_damped * xi;
% non linear ROM definition (use delta depending on xi and u)
f_rom = @(xi,u) ( S0*xi - D_rayleigh*xi ) - delta_fun_opt(xi,u)
*(L'*(L*xi)) + delta_fun_opt(xi,u)*(L'*u);
g_rom = @(xi) reducedModel.output(xi);
% Linearize at OP1
A1 = jac_cs(@(x) f_rom(x,u_eq1), xi_eq1_opt);
B1 = jac_cs(@(u) f_rom(xi_eq1_opt,u), u_eq1);
C1 = jac_cs(@(x) g_rom(x), xi_eq1_opt);
sys_r1 = ss(A1,B1,C1,0);
% Linearize at OP2
A2 = jac_cs(@(x) f_rom(x,u_eq2), xi_eq2_opt);
B2 = jac_cs(@(u) f_rom(xi_eq2_opt,u), u_eq2);
C2 = jac_cs(@(x) g_rom(x), xi_eq2_opt);
sys_r2 = ss(A2,B2,C2,0);
% Linearize at OP3
A3 = jac_cs(@(x) f_rom(x,u_eq3), xi_eq3_opt);
B3 = jac_cs(@(u) f_rom(xi_eq3_opt,u), u_eq3);
C3 = jac_cs(@(x) g_rom(x), xi_eq3_opt);
sys_r3 = ss(A3,B3,C3,0);
%% Section 10: Full-resolution Bode plots
w = logspace(-4, 0.5, 5000);
FOMs = {sys, sys1, sys2};
ROMs = {sys_r1, sys_r2, sys_r3};
OP_labels = {'OP1','OP2','OP3'};
for k = 1:3
FRF_f = squeeze(freqresp(FOMs{k}, 1i*w));
FRF_r = squeeze(freqresp(ROMs{k}, 1i*w));
mag_f = abs(FRF_f); ph_f = wrapTo180(rad2deg(unwrap(angle(FRF_f))));
mag_r = abs(FRF_r); ph_r = wrapTo180(rad2deg(unwrap(angle(FRF_r))));
figure('Name', ['Bode - ' OP_labels{k}], 'NumberTitle','off');
subplot(2,1,1);
semilogx(w,20*log10(mag_f),'r-',w,20*log10(mag_r),'b--', 'LineWidth',
 1.2);
grid on; ylabel('Magnitude [dB]');
title(['FOM vs ROM - Magnitude - ' OP_labels{k}]);
legend('FOM','ROM','Location','best');
subplot(2,1,2);
semilogx(w, ph_f, 'r-', w, ph_r, 'b--', 'LineWidth', 1.2);
grid on; ylabel('Phase [deg]'); xlabel('Frequency [rad/s]');
title(['FOM vs ROM - Phase - ' OP_labels{k}]);
```

```
217.  end
218.  %% Section 11: Print results
219.  fprintf('Optimized alpha_damp: %.6g\n', alpha_damp_opt);
220.  fprintf('Optimized beta_damp: %.6g\n', beta_damp_opt);
221.  fprintf('Optimized delta params (p1..p5):
222.   %s\n', mat2str(p_delta_opt,6));
223.  fprintf('xi_eq1_opt: %s\n', mat2str(xi_eq1_opt,6));
224.  fprintf('xi_eq2_opt: %s\n', mat2str(xi_eq2_opt,6));
225.  fprintf('xi_eq3_opt: %s\n', mat2str(xi_eq3_opt,6));
226.  %% Local functions
227.  function J = jac_cs(fun, x)
228.  % complex-step Jacobian (returns real Jacobian)
229.  % fun: function handle returning vector given x (or u)
230.  h = 1e-20;
231.  n = numel(x);
231.  f0 = fun(x); f0 = f0(:);
232.  m = numel(f0);
233.  J = zeros(m,n);
234.  for k = 1:n
235.  e = zeros(n,1); e(k) = 1;
236.  fp = fun(x + 1i*h*e);
237.  J(:,k) = imag(fp(:))/h;
238.  end
239.  end
240.  function obj = bode_fit_obj_3op_fast_with_u(p_all, build_delta, RM,
241.  u1, u2, u3, ...
242.  mag_FOM1, ph_FOM1, mag_FOM2, ph_FOM2, mag_FOM3, ph_FOM3, w)
243.  % Fast objective: compare FRF ROM con FRF FOM
244.  precompute(mag + phase)
245.  % p_all = [alpha; beta; p_delta(5); xi1(nu); xi2(nu); xi3(nu)]
246.  % extract alpha and beta
247.  alpha_damp = p_all(1);
248.  beta_damp = p_all(2);
249.  % extract p_delta (5 params)
250.  p_delta = p_all(3:7);
251.  % number of reduced states nu determined by RM.S0
252.  S0_local = RM.S0;
253.  nu = size(S0_local,1);
254.  % check length
255.  if numel(p_all) ~= 7 + 3*nu
256.  obj = 1e12;
257.  return;
258.  end
259.  % extract xi
260.  xi1 = p_all(8 : 7+nu);
261.  xi2 = p_all(8+nu : 7+2*nu);
262.  xi3 = p_all(8+2*nu : 7+3*nu);
263.  % rebult damping and delta
264.  delta_h = build_delta(p_delta);
265.  % rebuilt Rayleigh and S_damped locals
266.  D_rayleigh_local = alpha_damp * eye(nu) + beta_damp * S0_local;
267.  S_damped_local = S0_local - D_rayleigh_local;
268.  s_fun_local = @(xi) S_damped_local * xi;
```

```
% defined f_red including s_fun_local and delta_h (delta takes xi,u)
f_red_h = @(xi,u) s_fun_local(xi) ...
- delta_h(xi,u) * (RM.L'*(RM.L*xi)) ...
+ delta_h(xi,u) * (RM.L'*u);
g_red_h = @(xi) RM.output(xi);
% Linearize at three OPs (complex-step jacobian)
A1 = jac_cs(@(x) f_red_h(x,u1), xi1);
B1 = jac_cs(@(u) f_red_h(xi1,u), u1);
C1 = jac_cs(@(x) g_red_h(x), xi1);
sys1 = ss(A1,B1,C1,0);
A2 = jac_cs(@(x) f_red_h(x,u2), xi2);
B2 = jac_cs(@(u) f_red_h(xi2,u), u2);
C2 = jac_cs(@(x) g_red_h(x), xi2);
sys2 = ss(A2,B2,C2,0);
A3 = jac_cs(@(x) f_red_h(x,u3), xi3);
B3 = jac_cs(@(u) f_red_h(xi3,u), u3);
C3 = jac_cs(@(x) g_red_h(x), xi3);
sys3 = ss(A3,B3,C3,0);
% compute FRF for each linear systems on w
FRFr1 = squeeze(freqresp(sys1, w));
FRFr2 = squeeze(freqresp(sys2, w));
FRFr3 = squeeze(freqresp(sys3, w));
% magnitude differences (dB) and phase diffs in degrees
mag_err1 = 20*log10(abs(FRFr1)+eps) - mag_FOM1;
mag_err2 = 20*log10(abs(FRFr2)+eps) - mag_FOM2;
mag_err3 = 20*log10(abs(FRFr3)+eps) - mag_FOM3;
ph_err1 = wrapTo180(rad2deg(unwrap(angle(FRFr1))) - ph_FOM1);
ph_err2 = wrapTo180(rad2deg(unwrap(angle(FRFr2))) - ph_FOM2);
ph_err3 = wrapTo180(rad2deg(unwrap(angle(FRFr3))) - ph_FOM3);
% cost = RMS of mag errors across three OPs + weighted phase + simple
reg on p_delta
Nw = length(w);
obj = (norm(mag_err1(:)) + norm(mag_err2(:)) + norm(mag_err3(:))) /
sqrt(Nw) + ...
0.01 * (norm(ph_err1(:)) + norm(ph_err2(:)) + norm(ph_err3(:))) / 304.
    sqrt(Nw) + ...
1e-3 * sum(abs(p_delta(:)));
% small regularisation on alpha/beta could be added if desired
end
% to realize reduced-order models without state-input coupling use
% This delta function
% delta_default=@(xi)diag(0.5 + 0.5*tanh(2*xi(:)) + 0.2*(xi(:).^2));
```

D.2 MIMO Reduction Algorithm: MATLAB Code

The complete MATLAB code that implements the MIMO reduction algorithm for heat exchanger-turbine is reported below

```
%% FULL ORDER MODELS (FOM)
sys = ss(A_full_1,B_full_1,C_full_1,D_full_1);
sys1 = ss(A_full_2,B_full_2,C_full_2,D_full_2);
sys2 = ss(A_full_3,B_full_3,C_full_3,D_full_3);
sys.D = 0; sys1.D = 0; sys2.D = 0;
```

```
%% RANDOM INITS
n = size(sys.A,1);
x0_1 = randn(n,1);
x0_2 = randn(n,1);
x0_3 = randn(n,1);
damp=0.005;
%% SIGNAL GENERATOR / S (interpolation)
freq = [0.1 0.001 1 0.1];
S = computeSFromInterpolationPoints(freq);
S = S - damp * eye(size(S)); % small shift
nu = size(S,1);
L = ones(1, nu); L(2:2:end) = 0;
L = blkdiag(L, L); % final reduced dimension = 2*nu
S = blkdiag(S, S);
nu = size(S,1); % nu final (2*original)
omega0 = randn(nu,1);
%% MOMENT MATCHING FROM MODEL
Pi = computeMoment(S, L, sys.A, sys.B);
Pi1 = computeMoment(S, L, sys1.A, sys1.B);
CPi_Model = sys.C * Pi;
CPi_Model1 = sys1.C * Pi1;
CPi_All = (CPi_Model + CPi_Model1) / 2;
%% ROM base maps
s_fun = @(xi) S * xi;
l_fun = @(xi) L * xi;
%% baseline delta
delta_baseline = @(xi) diag( 0.16 + 0.5 * tanh(2 * xi(:))
 + 0.2 *  (xi(:).^2) );
%% parametrized delta builder
% p = [p1, p2, p3, p4, p5]
% delta(xi,u) = diag( p1 + p2 * tanh(p3*real(xi)) + p4*(real(xi).^2)
+ p5*( 37.   %real(xi) .* sum(real(u)) ) )
build_delta = @(p) ( @(xi,u) diag( ...
p(1) + ...
p(2) .* tanh( p(3) .* real(xi(:)) ) + ...
p(4) .* (real(xi(:)).^2) + ...
p(5) .* ( real(xi(:)) .* sum(real(u(:))) ) ...
) );
% create ROM dynamics builder using delta_handle(xi,u)
make_fred = @(delta_handle) ( @(xi,u) ...
s_fun(xi) ...
- delta_handle(xi,u) * (L' * (L * xi)) ...
+ delta_handle(xi,u) * (L' * u) ...
);
% linear output map from moments
g_red = @(xi) CPi_All * xi;
%% OPERATING POINTS (initial guesses)
u_eq1 = [0.35; 0.35];
u_eq2 = [-0.14; -0.14];
u_eq3 = [0.105; 0.105];
xi_eq1 = 0.1 * ones(nu,1);
xi_eq2 = 0.1 * ones(nu,1);
xi_eq3 = 0.1 * ones(nu,1);
```

```
%% FREQUENCY GRIDS
w_opt = logspace(-4, 0.5, 600); % optimization grid
w_plot = logspace(-4, 0.5, 800); % plotting grid
%% PRECOMPUTE SINGULAR VALUES (FOM)
fprintf('Precomputing FOM singular values on optimization
grid...\n');
S_fom1 = squeeze(sigma(sys, w_opt));
S_fom2 = squeeze(sigma(sys1, w_opt));
S_fom3 = squeeze(sigma(sys2, w_opt));
nSV = size(S_fom1,1);
%% OPTIMIZATION: delta parameters + xi_eq (for 3 OPs)
% decision vector q = [p(1:5); xi1(1:nu); xi2(1:nu); xi3(1:nu)]
p0_delta = [0.16, 0.5, 2.0, 0.2, 0.01]; % initial guess for 5 params
p0_all = [p0_delta(:);
xi_eq1(:);
xi_eq2(:);
xi_eq3(:)];
% objective wrapper
objfun_all = @(q) sigma_distance_obj_with_xi(q, build_delta, S, L,
CPi_All, ...
u_eq1, u_eq2, u_eq3, ...
w_opt, S_fom1, S_fom2, S_fom3);
opts = optimset('Display','iter','TolX',1e-6,
'TolFun',1e-6,'MaxIter',30000,'MaxFunEvals',40000);
fprintf("\n=== START DELTA(p,xi_eq) OPTIMIZATION (3 OPs) ===\n");
[q_opt, fval_opt] = fminsearch(objfun_all, p0_all, opts);
fprintf("Done: fval = %.6g\n", fval_opt);
% Extract optimized pieces
p_opt = q_opt(1:5);
idx_xi1 = 6 : 5 + nu;
idx_xi2 = 5 + nu + 1 : 5 + 2*nu;
idx_xi3 = 5 + 2*nu + 1 : 5 + 3*nu;
xi_eq1_opt = q_opt(idx_xi1);
xi_eq2_opt = q_opt(idx_xi2);
xi_eq3_opt = q_opt(idx_xi3);
delta_fun_opt = build_delta(p_opt);
f_red = make_fred(delta_fun_opt);
ROM.S = S; ROM.L = L; ROM.delta = delta_fun_opt; ROM.s = s_fun;
 ROM.l = l_fun; ROM.output = g_red;
%% LINEARIZE ROM at optimized equilibria (complex-step jacobian)
A_red1 = jac_cs(@(x) f_red(x, u_eq1), xi_eq1_opt);
B_red1 = jac_cs(@(u) f_red(xi_eq1_opt, u), u_eq1);
C_red1 = jac_cs(@(x) g_red(x), xi_eq1_opt);
sys_red_lin1 = ss(A_red1, B_red1, C_red1,
zeros(size(C_red1,1), size(B_red1,2)));
A_red2 = jac_cs(@(x) f_red(x, u_eq2), xi_eq2_opt);
B_red2 = jac_cs(@(u) f_red(xi_eq2_opt, u), u_eq2);
C_red2 = jac_cs(@(x) g_red(x), xi_eq2_opt);
sys_red_lin2 = ss(A_red2, B_red2, C_red2,
zeros(size(C_red2,1), size(B_red2,2)));
A_red3 = jac_cs(@(x) f_red(x, u_eq3), xi_eq3_opt);
B_red3 = jac_cs(@(u) f_red(xi_eq3_opt, u), u_eq3);
C_red3 = jac_cs(@(x) g_red(x), xi_eq3_opt);
```

```
sys_red_lin3 = ss(A_red3, B_red3, C_red3,
zeros(size(C_red3,1), size(B_red3,2)));
%% SINGULAR VALUE PLOTS
w = w_plot;
S_fom1_plot = squeeze(sigma(sys, w));
S_fom2_plot = squeeze(sigma(sys1, w));
S_fom3_plot = squeeze(sigma(sys2, w));
S_rom1_plot = squeeze(sigma(sys_red_lin1, w));
S_rom2_plot = squeeze(sigma(sys_red_lin2, w));
S_rom3_plot = squeeze(sigma(sys_red_lin3, w));
% OP1
figure('Name','Singular Values - OP1'); hold on;
for ksv=1:size(S_fom1_plot,1)
semilogx(w, 20*log10(S_fom1_plot(ksv,:)),'r-');
semilogx(w, 20*log10(S_rom1_plot(ksv,:)),'b--');
end
grid on; title('Singular values - FOM vs ROM (OP1)');
legend('FOM','ROM'); xlabel('Frequency [rad/s]');
ylabel('Magnitude [dB]'); set(gca,'XScale','log');
% OP2
figure('Name','Singular Values - OP2'); hold on;
for ksv=1:size(S_fom2_plot,1)
semilogx(w, 20*log10(S_fom2_plot(ksv,:)),'g-');
semilogx(w, 20*log10(S_rom2_plot(ksv,:)),'b--');
end
grid on; title('Singular values - FOM vs ROM (OP2)');
legend('FOM','ROM'); xlabel('Frequency [rad/s]');
 ylabel('Magnitude [dB]'); set(gca,'XScale','log');
% OP3
figure('Name','Singular Values - OP3'); hold on;
for ksv=1:size(S_fom3_plot,1)
semilogx(w, 20*log10(S_fom3_plot(ksv,:)),'m-');
semilogx(w, 20*log10(S_rom3_plot(ksv,:)),'b--');
end
grid on; title('Singular values - FOM vs ROM (OP3)');
legend('FOM','ROM'); xlabel('Frequency [rad/s]');
ylabel('Magnitude [dB]'); set(gca,'XScale','log');
%% REPORT
fprintf('Optimized p (5 params): %s\n', mat2str(p_opt,6));
fprintf('xi_eq1_opt: %s\n', mat2str(xi_eq1_opt,6));
fprintf('xi_eq2_opt: %s\n', mat2str(xi_eq2_opt,6));
fprintf('xi_eq3_opt: %s\n', mat2str(xi_eq3_opt,6));
fprintf('Final objective: %.6g\n', fval_opt);
%% LOCAL FUNCTIONS
function J = jac_cs(fun, x)
% complex-step Jacobian (returns real Jacobian)
h = 1e-20;
n = numel(x);
f0 = fun(x); f0 = f0(:);
m = numel(f0);
J = zeros(m,n);
for k = 1:n
e = zeros(n,1); e(k) = 1;
```

```
164.  fp = fun(x + 1i*h*e);
165.  J(:,k) = imag(fp(:))/h;
166.  end
167.  end
168.  function obj = sigma_distance_obj_with_xi(q, build_delta_handle,
169.   S, L, CPi_All, ...
170.  u1, u2, u3, w, S_fom1, S_fom2, S_fom3)
171.  % q = [p(1:5); xi1(nu); xi2(nu); xi3(nu)]
172.  p = q(1:5);
173.  % compute nu from S
174.  nu = size(S,1);
175.  % indices
176.  xi1_idx = 6 : 5+nu;
177.  xi2_idx = 5+nu+1 : 5+2*nu;
178.  xi3_idx = 5+2*nu+1 : 5+3*nu;
179.  xi1 = q(xi1_idx);
180.  xi2 = q(xi2_idx);
181.  xi3 = q(xi3_idx);
182.  xis = {xi1, xi2, xi3};
183.  ues = {u1, u2, u3};
184.  % build delta handle that accepts (xi,u)
185.  delta_h = build_delta_handle(p);
186.  % reduced dynamics handle f_red_h(xi,u) using delta_h(xi,u)
187.  f_red_h = @(xi,u) S*xi - delta_h(xi,u) * (L' * (L * xi)) +
delta_h(xi,u) * 188.    (L' * u);
189.  g_red_h = @(xi) CPi_All * xi;
190.  Nop = numel(xis);
191.  errs = zeros(Nop,1);
192.  for k = 1:Nop
193.  xi0 = xis{k};
194.  ue = ues{k};
195.  % Jacobians with delta depending on (xi,u)
196.  A_k = jac_cs(@(x) f_red_h(x, ue), xi0);
197.  B_k = jac_cs(@(u) f_red_h(xi0, u), ue);
198.  C_k = jac_cs(@(x) g_red_h(x), xi0);
199.  % build linear system
200.  sysk = ss(A_k, B_k, C_k, zeros(size(C_k,1), size(B_k,2)));
201.  % singular values of ROM linear system on grid w
202.  S_rom = squeeze(sigma(sysk, w));
203.  % pick reference FOM singular values
204.  if k==1
205.  S_ref = S_fom1;
206.  elseif k==2
207.  S_ref = S_fom2;
208.  else
209.  S_ref = S_fom3;
210.  end
211.  % match dimensions (nSV x nw)
212.  nmin = min(size(S_ref,1), size(S_rom,1));
213.  S_ref_tr = S_ref(1:nmin,:);
214.  S_rom_tr = S_rom(1:nmin,:);
215.  % compute dB difference and RMS norm
215.  diff = 20*log10(S_rom_tr + eps) - 20*log10(S_ref_tr + eps);
```

```
216.  errs(k) = norm(diff(:)) / sqrt(numel(w));
217.  end
218.  % objective average + small regularization on p + small penalty to
keep xi 219.      %near initial guess
220.  obj = mean(errs) + 1e-6 * norm(p(:),2);
221.  end
```

The complete MATLAB code that implements the SISO reduction algorithm for the SOFC is reported below

```
1.    %% Section 0: user damping parameters (init)
2.    alpha_damp = 0.025; % scalar damping (init)
3.    beta_damp = 0.75; % resonance damping (init)
4.    %% Section 1: Full Order Models (FOMs)
5.    sys = ss(A_full_1,B_full_1,C_full_1,D_full_1);
6.    sys1 = ss(A_full_2,B_full_2,C_full_2,D_full_2);
7.    sys2 = ss(A_full_3,B_full_3,C_full_3,D_full_3);
8.    % Ensure D matrices are zero
9.    sys.D = 0;
10.   sys1.D = 0;
11.   sys2.D = 0;
12.   %% Section 2: Datasets preprocessing (steady-state)
13.   t_out = exportedVariables1008match.time; % time vector (example)
14.   tSS = 200; fs = 5;
15.   idx = find(t_out > tSS,1,'first');
16.   omega_full_1 = [exportedVariables1008match.current_Ix_out1, ...
17.   exportedVariables1008match.current_Ix_out2, ...
18.   exportedVariables1008match.current_Ix_out3, ...
19.   exportedVariables1008match.current_Ix_out4, ...
20.   exportedVariables1008match.current_Ix_out5, ...
21.   exportedVariables1008match.current_Ix_out6, ...
22.   exportedVariables1008match.current_Ix_out7, ...
23.   exportedVariables1008match.current_Ix_out8, ...
24.   exportedVariables1008match.current_Ix_out9, ...
25.   exportedVariables1008match.current_Ix_out10, ...
26.   exportedVariables1008match.current_Ix_out11, ...
27.   exportedVariables1008match.current_Ix_out12, ...
28.   exportedVariables1008match.current_Ix_out13, ...
29.   exportedVariables1008match.current_Ix_out14, ...
30.   exportedVariables1008match.current_Ix_out15, ...
31.   exportedVariables1008match.current_Ix_out16]';
32.   omega_full_2 = [exportedVariables308match.current_Ix_out1, ...
33.   exportedVariables308match.current_Ix_out2, ...
34.   exportedVariables308match.current_Ix_out3, ...
35.   exportedVariables308match.current_Ix_out4, ...
36.   exportedVariables308match.current_Ix_out5, ...
37.   exportedVariables308match.current_Ix_out6, ...
38.   exportedVariables308match.current_Ix_out7, ...
39.   exportedVariables308match.current_Ix_out8, ...
40.   exportedVariables308match.current_Ix_out9, ...
41.   exportedVariables308match.current_Ix_out10, ...
42.   exportedVariables308match.current_Ix_out11, ...
43.   exportedVariables308match.current_Ix_out12, ...
```

```
44.   exportedVariables308match.current_Ix_out13, ...
45.   exportedVariables308match.current_Ix_out14, ...
46.   exportedVariables308match.current_Ix_out15, ...
47.   exportedVariables308match.current_Ix_out16]';
48.   y_full_1 = exportedVariables1008match.stackvStack;
49.   y_full_2 = exportedVariables308match.stackvStack;
50.   u_full_1 = exportedVariables1008match.current_Iy_out;
51.   u_full_2 = exportedVariables308match.current_Iy_out;
52.   % subsample steady-state
53.   omega_1 = omega_full_1(:, idx:fs:end);
54.   omega_2 = omega_full_2(:, idx:fs:end);
55.   y_1 = y_full_1(idx:fs:end)/scale;
56.   y_2 = y_full_2(idx:fs:end)/scale;
57.   u_1 = u_full_1(idx:fs:end);
59.   u_2 = u_full_2(idx:fs:end);
60.   % ensure column vectors for u datasets
61.   u_1 = u_1(:);
62.   u_2 = u_2(:);
63.   % combine
64.   omega_all = [omega_1, omega_2];
65.   y_all = [y_1(:); y_2(:)];
66.   u_all = [u_1(:); u_2(:)];
67.   %% Section 3: Nonlinear dictionary and weight estimation
68.   dictionaryOfFunc = {
69.   @(z) z(1);
70.   @(z) z(1)^3;
71.   };
72.   omegaNL_all = omega_all.'; % each row is a sample
73.   windowSize = size(omegaNL_all,1);
74.   nlValFromZeta_all = zeros(windowSize, length(dictionaryOfFunc));
75.   for i = 1:windowSize
76.   nlValFromZeta_all(i,:) =
77.   modred.internal.computeNonlinearityFromState(real(omegaNL_all(i,:)),
78.   dictionaryOfFunc);
79.   end
80.   yNL_all = y_all(:);
81.   if length(yNL_all) ~= windowSize
82.   yNL_all = yNL_all(1:windowSize);
83.   end
84.   lambda = 1e-6; % ridge
85.   weights = (nlValFromZeta_all' * nlValFromZeta_all +
86.   lambda*eye(length(dictionaryOfFunc)))
87.    \ (nlValFromZeta_all' * yNL_all);
88.   weights = weights(:);
89.   %% Section 4: ROM components damping build (nominal)
90.   % feature vector (phi)
91.   phi_col = @(xi) reshape(cell2mat(cellfun(@(f) f(xi),
92.   dictionaryOfFunc,'UniformOutput', false)), [], 1);
93.   % compute nominal S (interpolation-based)
94.   S0 = computeSFromInterpolationPoints([0.1 0.1 0.1 0.1 1 1 1 1]); %
95.   original S (user function)
96.   nu = size(S0,1); % reduced state dimension (rows)
```

```
97.   % damping D = alpha*I + beta*S0 (here use init values;will be
      opimized)
98.   D_rayleigh = alpha_damp * eye(nu) + beta_damp * S0;
99.   S_damped = S0 - D_rayleigh;
100.  L = ones(1,nu);
101.  s_fun = @(xi) S_damped * xi; % includes damping (nominal)
102.  l_fun = @(xi) L * xi;
103.  % delta default builder
104.  delta_default = @(xi,u) diag(0.5 + 0.5*tanh(2*xi(:))
105.   + 0.2*(xi(:).^2) + 0.01*(xi(:).*sum(u(:))));
106.  % reducedModel struct
107.  reducedModel.S = S_damped;
108.  reducedModel.S0 = S0;
109.  reducedModel.D_rayleigh = D_rayleigh;
110.  reducedModel.L = L;
111.  reducedModel.weights = weights;
112.  reducedModel.dictionaryOfFunc = dictionaryOfFunc;
113.  reducedModel.s = s_fun;
114.  reducedModel.l = l_fun;
115.  reducedModel.phi = phi_col;
116.  reducedModel.output = @(xi) (weights' * phi_col(xi));
117.  reducedModel.delta = delta_default;
118.  %% Section 5: Operating points (initial guess)
119.  xi_eq1 = 0.09 * ones(nu,1);
120.  xi_eq2 = 0.09 * ones(nu,1);
121.  xi_eq3 = 0.09 * ones(nu,1);
122.  u_eq1 = mean(u_1);
123.  u_eq2 = mean(u_2);
124.  u_eq3 = 6;
125.  %u_eq3=75;
126.  %% Section 6: Parametric delta(xi,u) to optimize (5 params)
127.  % build_delta returns a function handle delta(xi,u)
128.  build_delta = @(p) ( @(xi,u) diag( ...
129.  p(1) + ...
130.  p(2) .* tanh( p(3) .* real(xi(:)) ) + ...
131.  p(4) .* (real(xi(:)).^2) + ...
132.  p(5) .* ( real(xi(:)) .* sum(real(u(:))) ) ...
133.  ) );
134.  p0_delta = [1.0, 0.3, 2.0, 0.05, 0.01];
135.  %% Section 7: Precompute FOM FRFs
136.  w_opt = 1i * logspace(-5, 3, 1000); % complex freq vector (reduced
      points)
137.  FRF_f1_pre = squeeze(freqresp(sys, w_opt));
138.  FRF_f2_pre = squeeze(freqresp(sys1, w_opt));
139.  FRF_f3_pre = squeeze(freqresp(sys2, w_opt));
140.  % convert to magnitude (dB) and wrapped phase (deg) for objective
141.  mag_FOM1 = 20*log10(abs(FRF_f1_pre)+eps);
142.  ph_FOM1 = wrapTo180(rad2deg(unwrap(angle(FRF_f1_pre))));
143.  mag_FOM2 = 20*log10(abs(FRF_f2_pre)+eps);
144.  ph_FOM2 = wrapTo180(rad2deg(unwrap(angle(FRF_f2_pre))));
145.  mag_FOM3 = 20*log10(abs(FRF_f3_pre)+eps);
146.  ph_FOM3 = wrapTo180(rad2deg(unwrap(angle(FRF_f3_pre))));
147.  %% Section 8: Delta( p(1:5) ) + xi + alpha + beta optimization
```

```
% full parameter vector: [alpha_damp; beta_damp; p_delta(5);
xi1(nu); xi2(nu); xi3(nu)]
p0_all = [alpha_damp; beta_damp; p0_delta(:);
xi_eq1(:); xi_eq2(:); xi_eq3(:)];
% objective wrapper (fast: compares to precomputed mag/phase)
objfun_all = @(p_all) bode_fit_obj_3op_fast_with_u(p_all,
build_delta, reducedModel, ...
u_eq1, u_eq2, u_eq3, ...
mag_FOM1, ph_FOM1, mag_FOM2, ph_FOM2, mag_FOM3, ph_FOM3, w_opt);
opts = optimset('Display','iter','TolX',1e-6,
'TolFun',1e-6,'MaxIter',65000,'MaxFunEvals',75000);
fprintf("\n=== START DELTA(5 params) + XI + ALPHA/BETA OPTIMIZATION
(3 OPs) ===\n");
[p_opt_all, fval_opt] = fminsearch(objfun_all, p0_all, opts);
fprintf("Done: fval = %.6g\n", fval_opt);
% Extract optimized
% p_opt_all structure: [alpha; beta; p_delta(1:5); xi1(nu); xi2(nu);
xi3(nu)]
alpha_damp_opt = p_opt_all(1);
beta_damp_opt = p_opt_all(2);
p_delta_opt = p_opt_all(3:7);
% compute nu and extract xi segments (usa nu da S0 per coerenza)
nu_check = size(S0,1);
xi1_idx = 8 : 7+nu_check;
xi2_idx = 8+nu_check : 7+2*nu_check;
xi3_idx = 8+2*nu_check : 7+3*nu_check;
xi_eq1_opt = p_opt_all(xi1_idx);
xi_eq2_opt = p_opt_all(xi2_idx);
xi_eq3_opt = p_opt_all(xi3_idx);
delta_fun_opt = build_delta(p_delta_opt);
reducedModel.delta = delta_fun_opt;
%% Section 9: Rebuild Rayleigh damping and linearize ROM around
optimized OPs
% rebuilt D_rayleigh and S_damped with optimized parameters
D_rayleigh = alpha_damp_opt * eye(nu) + beta_damp_opt * S0;
S_damped = S0 - D_rayleigh;
% update reducedModel.s
reducedModel.S = S_damped;
reducedModel.D_rayleigh = D_rayleigh;
reducedModel.s = @(xi) S_damped * xi;
% non linear ROM definition (use delta depending on xi and u)
f_rom = @(xi,u) ( S0*xi - D_rayleigh*xi ) - delta_fun_opt(xi,u)*(L'*
(L*xi))+ delta_fun_opt(xi,u)*(L'*u);
g_rom = @(xi) reducedModel.output(xi);
% Linearize at OP1
A1 = jac_cs(@(x) f_rom(x,u_eq1), xi_eq1_opt);
194.  B1 = jac_cs(@(u) f_rom(xi_eq1_opt,u), u_eq1);
C1 = jac_cs(@(x) g_rom(x), xi_eq1_opt);
sys_r1 = ss(A1,B1,C1,0);
% Linearize at OP2
A2 = jac_cs(@(x) f_rom(x,u_eq2), xi_eq2_opt);
B2 = jac_cs(@(u) f_rom(xi_eq2_opt,u), u_eq2);
C2 = jac_cs(@(x) g_rom(x), xi_eq2_opt);
```

```
sys_r2 = ss(A2,B2,C2,0);
% Linearize at OP3
A3 = jac_cs(@(x) f_rom(x,u_eq3), xi_eq3_opt);
B3 = jac_cs(@(u) f_rom(xi_eq3_opt,u), u_eq3);
C3 = jac_cs(@(x) g_rom(x), xi_eq3_opt);
sys_r3 = ss(A3,B3,C3,0);
%% Section 10: Full-resolution Bode plots
w = logspace(-5, 3, 5000);
FOMs = {sys, sys1, sys2};
ROMs = {sys_r1, sys_r2, sys_r3};
OP_labels = {'OP1','OP2','OP3'};
for k = 1:3
FRF_f = squeeze(freqresp(FOMs{k}, 1i*w));
FRF_r = squeeze(freqresp(ROMs{k}, 1i*w));
mag_f = abs(FRF_f); ph_f = wrapTo180(rad2deg(unwrap(angle(FRF_f))));
mag_r = abs(FRF_r); ph_r = wrapTo180(rad2deg(unwrap(angle(FRF_r))));
figure('Name', ['Bode - ' OP_labels{k}], 'NumberTitle','off');
subplot(2,1,1);
semilogx(w, 20*log10(mag_f), 'r-', w, 20*log10(mag_r), 'b--',
'LineWidth', 221. 1.2); grid on; ylabel('Magnitude [dB]');
title(['FOM vs ROM - Magnitude - ' OP_labels{k}]);
legend('FOM','ROM','Location','best');
subplot(2,1,2);
semilogx(w, ph_f, 'r-', w, ph_r, 'b--', 'LineWidth', 1.2);
grid on; ylabel('Phase [deg]'); xlabel('Frequency [rad/s]');
title(['FOM vs ROM - Phase - ' OP_labels{k}]);
end
%% Section 11: Print results
fprintf('Optimized alpha_damp: %.6g\n', alpha_damp_opt);
fprintf('Optimized beta_damp: %.6g\n', beta_damp_opt);
fprintf('Optimized delta params (p1..p5): %s\n',
mat2str(p_delta_opt,6));
fprintf('xi_eq1_opt: %s\n', mat2str(xi_eq1_opt,6));
fprintf('xi_eq2_opt: %s\n', mat2str(xi_eq2_opt,6));
fprintf('xi_eq3_opt: %s\n', mat2str(xi_eq3_opt,6));
%% Local functions
function J = jac_cs(fun, x)
% complex-step Jacobian (returns real Jacobian)
% fun: function handle returning vector given x (or u)
h = 1e-20;
n = numel(x);
f0 = fun(x); f0 = f0(:);
m = numel(f0);
J = zeros(m,n);
for k = 1:n
e = zeros(n,1); e(k) = 1;
fp = fun(x + 1i*h*e);
J(:,k) = imag(fp(:))/h;
end
end
function obj = bode_fit_obj_3op_fast_with_u(p_all, build_delta, RM,
u1, u2, u3, ...
mag_FOM1, ph_FOM1, mag_FOM2, ph_FOM2, mag_FOM3, ph_FOM3, w)
```

```
% Fast objective: compare FRF ROM con FRF FOM precompute (mag + phase)
% p_all = [alpha; beta; p_delta(5); xi1(nu); xi2(nu); xi3(nu)]
% extract alpha and beta
alpha_damp = p_all(1);
beta_damp = p_all(2);
% extract p_delta (5 params)
p_delta = p_all(3:7);
% number of reduced states nu determined by RM.S0
S0_local = RM.S0;
nu = size(S0_local,1);
% check length
if numel(p_all) ~= 7 + 3*nu
obj = 1e12;
return;
end
% extract xi
xi1 = p_all(8 : 7+nu);
xi2 = p_all(8+nu : 7+2*nu);
xi3 = p_all(8+2*nu : 7+3*nu);
% rebult damping and delta
delta_h = build_delta(p_delta);
% rebuilt Rayleigh and S_damped locals
D_rayleigh_local = alpha_damp * eye(nu) + beta_damp * S0_local;
S_damped_local = S0_local - D_rayleigh_local;
s_fun_local = @(xi) S_damped_local * xi;
% defined f_red including s_fun_local and delta_h (delta takes xi,u)
f_red_h = @(xi,u) s_fun_local(xi) ...
- delta_h(xi,u) * (RM.L'*(RM.L*xi)) ...
+ delta_h(xi,u) * (RM.L'*u);
g_red_h = @(xi) RM.output(xi);
% Linearize at three OPs (complex-step jacobian)
A1 = jac_cs(@(x) f_red_h(x,u1), xi1);
B1 = jac_cs(@(u) f_red_h(xi1,u), u1);
C1 = jac_cs(@(x) g_red_h(x), xi1);
sys1 = ss(A1,B1,C1,0);
A2 = jac_cs(@(x) f_red_h(x,u2), xi2);
B2 = jac_cs(@(u) f_red_h(xi2,u), u2);
C2 = jac_cs(@(x) g_red_h(x), xi2);
sys2 = ss(A2,B2,C2,0);
A3 = jac_cs(@(x) f_red_h(x,u3), xi3);
B3 = jac_cs(@(u) f_red_h(xi3,u), u3);
C3 = jac_cs(@(x) g_red_h(x), xi3);
sys3 = ss(A3,B3,C3,0);
% compute FRF for each linear systems on w
FRFr1 = squeeze(freqresp(sys1, w));
FRFr2 = squeeze(freqresp(sys2, w));
FRFr3 = squeeze(freqresp(sys3, w));
% magnitude differences (dB) and phase diffs in degrees
mag_err1 = 20*log10(abs(FRFr1)+eps) - mag_FOM1;
mag_err2 = 20*log10(abs(FRFr2)+eps) - mag_FOM2;
mag_err3 = 20*log10(abs(FRFr3)+eps) - mag_FOM3;
ph_err1 = wrapTo180(rad2deg(unwrap(angle(FRFr1))) - ph_FOM1);
```

```
307.  ph_err2 = wrapTo180(rad2deg(unwrap(angle(FRFr2))) - ph_FOM2);
308.  ph_err3 = wrapTo180(rad2deg(unwrap(angle(FRFr3))) - ph_FOM3);
309.  % cost = RMS of mag errors across three OPs + weighted phase +
310. simple reg on p_delta
311.  Nw = length(w);
312.  obj = (norm(mag_err1(:)) + norm(mag_err2(:)) + norm(mag_err3(:))) /
313.  sqrt(Nw) + ...
314.  0.01 * (norm(ph_err1(:)) + norm(ph_err2(:)) + norm(ph_err3(:)))
315.   / sqrt(Nw) + ...
316.  1e-3 * sum(abs(p_delta(:)));
317.  % small regularisation on alpha/beta could be added if desired
318.  End
```

The complete MATLAB code that implements the MIMO reduction algorithm for the SOFC is reported below

```
1. %% FULL ORDER MODELS (FOM)
2. sysFOM = {
3. ss(A_full_1,B_full_1,C_full_1,D_full_1)
4. ss(A_full_2,B_full_2,C_full_2,D_full_2)
5. ss(A_full_3,B_full_3,C_full_3,D_full_3)
6. ss(A_full_4,B_full_4,C_full_4,D_full_4)
7. ss(A_full_5,B_full_5,C_full_5,D_full_5)
8. ss(A_full_6,B_full_6,C_full_6,D_full_6)
9. ss(A_full_7,B_full_7,C_full_7,D_full_7)
10. ss(A_full_8,B_full_8,C_full_8,D_full_8)
11. ss(A_full_9,B_full_9,C_full_9,D_full_9)
12. };
13. Nop = numel(sysFOM);
14. %% SIGNAL GENERATOR / INTERPOLATION MODEL
15. freq = [0.1 0.1 0.1];
16. damp = 0.005;
17. S0 = computeSFromInterpolationPoints(freq);
18. S0 = S0 - damp * eye(size(S0));
19. nu0 = size(S0,1);
20. L0 = ones(1,nu0);
21. L0(2:2:end) = 0;
22. S = blkdiag(S0,S0);
23. L = blkdiag(L0,L0);
24. nu = size(S,1);
25. %% MOMENT MATCHING (AVERAGED OUTPUT MAP)
26. CPi_sum = 0;
27. for k=1:Nop
28. Pi_k = computeMoment(S,L,sysFOM{k}.A,sysFOM{k}.B);
29. CPi_sum = CPi_sum + sysFOM{k}.C * Pi_k;
30. end
31. CPi_All = CPi_sum / Nop;
32. ny = size(CPi_All,1);
33. nu_in = size(sysFOM{1}.B,2);
34. %% REDUCED MODEL MAPS
35. build_delta = @(p)( @(xi,u) diag( ...
36. p(1) + ...
37. p(2).*tanh(p(3).*real(xi(:))) + ...
```

```
p(4).*(real(xi(:)).^2) + ...
p(5).*( real(xi(:)) .* sum(real(u(:))) ) ));
make_fred = @(delta_h) ( @(xi,u) ...
S*xi ...
- delta_h(xi,u)*(L'*(L*xi)) ...
+ delta_h(xi,u)*(L'*u) );
%% OPERATING POINTS
u_eq = {
[ 0.35; 0.35]
[-0.14; -0.14]
[ 0.105; 0.105]
[ 0.20; 0.20]
[-0.25; -0.25]
[ 0.15; 0.15]
[-0.05; -0.05]
[ 0.30; 0.30]
[ 0.10; 0.10]
};
xi_eq0 = cell(Nop,1);
for k=1:Nop
xi_eq0{k} = 0.1*ones(nu,1);
end
%% FREQUENCY GRIDS
w_opt = logspace(-4,1.5,600);
w_plot = logspace(-4,1.5,800);
%% PRECOMPUTE FOM SINGULAR VALUES
fprintf('Precomputing FOM singular values...\n');
S_fom = cell(Nop,1);
for k=1:Nop
S_fom{k} = squeeze(sigma(sysFOM{k}, w_opt));
end
%% OPTIMIZATION VARIABLES (INCLUDING D0)
p0_delta = [0.16 0.5 2.0 0.2 0.01];
D0_init = zeros(ny,nu_in);
vecD0_init = D0_init(:);
q0 = [p0_delta(:); vecD0_init];
for k=1:Nop
q0 = [q0; xi_eq0{k}];
end
objfun = @(q) sigma_distance_obj_multi( ...
q, build_delta, S, L, CPi_All, ...
u_eq, w_opt, S_fom );
opts = optimset('Display','iter', ...
'TolX',1e-6,'TolFun',1e-6, ...
'MaxIter',70000,'MaxFunEvals',140000);
fprintf('\n=== START OPTIMIZATION (%d OPs) ===\n',Nop);
[q_opt,fval_opt] = fminsearch(objfun,q0,opts);
fprintf('Done. Final objective = %.6g\n',fval_opt);
%% EXTRACT OPTIMAL PARAMETERS
p_opt = q_opt(1:5);
idxD = 6 : 5 + ny*nu_in;
D0_opt = reshape(q_opt(idxD), ny, nu_in);
offset = 5 + ny*nu_in;
```

```
xi_eq_opt = cell(Nop,1);
for k=1:Nop
idx = offset + (k-1)*nu + (1:nu);
xi_eq_opt{k} = q_opt(idx);
end
delta_opt = build_delta(p_opt);
f_red = make_fred(delta_opt);
g_red = @(xi,u) CPi_All*xi + D0_opt*u;
%% LINEARIZE ROMs
sysROM= cell(Nop,1);
for k=1:Nop
xi0 = xi_eq_opt{k};
ue = u_eq{k};
A = jac_cs(@(x) f_red(x,ue), xi0);
B = jac_cs(@(u) f_red(xi0,u), ue);
C = jac_cs(@(x) g_red(x,ue), xi0);
D = jac_cs(@(u) g_red(xi0,u), ue);
sysROM{k} = ss(A,B,C,D);
end
%% SINGULAR VALUE PLOTS
for k=1:Nop
figure('Name',sprintf('Singular Values - OP%d',k)); hold on;
S_fom_plot = squeeze(sigma(sysFOM{k}, w_plot));
S_rom_plot = squeeze(sigma(sysROM{k}, w_plot));
for i=1:size(S_fom_plot,1)
semilogx(w_plot,20*log10(S_fom_plot(i,:)),'r-');
semilogx(w_plot,20*log10(S_rom_plot(i,:)),'b--');
end
grid on;
set(gca,'XScale','log');
title(sprintf('FOM vs ROM - OP%d',k));
legend('FOM','ROM');
end
%% REPORT
fprintf('\nOptimized p:\n');
disp(p_opt)
fprintf('Optimized D0:\n');
disp(D0_opt)
fprintf('Final objective = %.6g\n',fval_opt);
%% LOCAL FUNCTIONS
function J = jac_cs(fun,x)
h = 1e-20;
n = numel(x);
f0 = fun(x); f0 = f0(:);
m = numel(f0);
J = zeros(m,n);
for k=1:n
e = zeros(n,1); e(k)=1;
fp = fun(x + 1i*h*e);
J(:,k) = imag(fp(:))/h;
end
end
```

```
function obj = sigma_distance_obj_multi(q, build_delta, S, L, CPi_All,
...
u_eq, w, S_fom)
p = q(1:5);
ny = size(CPi_All,1);
nu_in = size(u_eq{1},1);
nu = size(S,1);
idxD = 6 : 5 + ny*nu_in;
D0 = reshape(q(idxD), ny, nu_in);
offset = 5 + ny*nu_in;
Nop = numel(u_eq);
delta_h = build_delta(p);
f_red = @(xi,u) S*xi ...
- delta_h(xi,u)*(L'*(L*xi)) ...
+ delta_h(xi,u)*(L'*u);
g_red = @(xi,u) CPi_All*xi + D0*u;
errs = zeros(Nop,1);
for k=1:Nop
idx = offset + (k-1)*nu + (1:nu);
xi0 = q(idx);
ue = u_eq{k};
A = jac_cs(@(x) f_red(x,ue), xi0);
B = jac_cs(@(u) f_red(xi0,u), ue);
C = jac_cs(@(x) g_red(x,ue), xi0);
D = jac_cs(@(u) g_red(xi0,u), ue);
sysk = ss(A,B,C,D);
S_rom = squeeze(sigma(sysk,w));
S_ref = S_fom{k};
nmin = min(size(S_ref,1),size(S_rom,1));
diff = 20*log10(S_rom(1:nmin,:)+eps) ...
- 20*log10(S_ref(1:nmin,:)+eps);
errs(k) = norm(diff(:))/sqrt(numel(w));
end
obj = mean(errs) ...
+ 1e-6*norm(p) ...
+ 1e-6*norm(D0,'fro');
end
```

Appendix E

	OP1	OP2	OP3
Order $v = 4$	-0.1170 + 0.0749i	-0.3106	-0.3247 + 0.0000i
	-0.1170 - 0.0749i	-0.0215	-0.0220 + 0.0366i
	-0.0512 + 0.0000i	-0.0530	-0.0220 - 0.0366i
	-0.0253 + 0.0000i	-0.0593	-0.0500 + 0.0000i

Order $v = 8$	-0.6292 + 0.0000i -0.0194 + 0.0987i -0.0194 - 0.0987i -0.0163 + 0.0101i -0.0163 - 0.0101i -0.0202 + 0.0016i -0.0202 - 0.0016i -0.0161 + 0.0000i	-0.5675 + 0.0000i -0.0160 + 0.1036i -0.0160 - 0.1036i -0.0163 + 0.0101i -0.0163 - 0.0101i -0.0147 + 0.0000i -0.0176 + 0.0000i -0.0166 + 0.0000i	-0.5790 + 0.0000i -0.0195 + 0.1002i -0.0195 - 0.1002i -0.0265 + 0.0000i -0.0163 + 0.0101i -0.0163 - 0.0101i -0.0161 + 0.0009i -0.0161 - 0.0009i
Order $v = 12$	-0.5080 + 0.0000i -0.0556 + 0.0565i -0.0556 - 0.0565i -0.0576 + 0.0097i -0.0576 - 0.0097i -0.0532 + 0.0000i -0.0606 + 0.0020i -0.0606 - 0.0020i -0.0576 + 0.0971i -0.0576 - 0.0971i -0.0576 + 0.0010i -0.0576 - 0.0010i	-0.2616 + 0.0000i -0.1053 + 0.0462i -0.1053 - 0.0462i -0.0177 + 0.0000i -0.0576 + 0.0097i -0.0576 - 0.0097i -0.0623 + 0.0000i -0.0615 + 0.0000i -0.0576 + 0.0971i -0.0576 - 0.0971i -0.0576 + 0.0010i -0.0576 - 0.0010i	-0.1745 + 0.0000i -0.1479 + 0.0596i -0.1479 - 0.0596i -0.0235 + 0.0000i -0.0576 + 0.0097i -0.0576 - 0.0097i -0.0594 + 0.0000i -0.0574 + 0.0000i -0.0576 + 0.0971i -0.0576 - 0.0971i -0.0576 + 0.0010i -0.0576 - 0.0010i

Table E.1: Eigenvalues of linearized ROM around OPs

	OP1	OP2	OP3
Order $v = 12$	-0.5258 + 0.0000i -0.2305 + 0.0000i	-0.3897 + 0.0000i -0.2630 + 0.0000i	-0.5132 + 0.0000i -0.0239 + 0.0000i

	-0.0274 + 0.0000i	-0.0196 + 0.0000i	-0.1766 + 0.0000i
	-0.1133 + 0.0062i	-0.1134 + 0.0065i	-0.1133 + 0.0062i
	-0.1133 - 0.0062i	-0.1134 - 0.0065i	-0.1133 - 0.0062i
	-0.1124 + 0.0048i	-0.1133 + 0.0062i	-0.1121 + 0.0042i
	-0.1124 - 0.0048i	-0.1133 - 0.0062i	-0.1121 - 0.0042i
	-0.1130 + 0.0000i	-0.1134 + 0.0000i	-0.1144 + 0.0000i
	-0.1133 + 0.0623i	-0.1133 + 0.0623i	-0.1133 + 0.0623i
	-0.1133 - 0.0623i	-0.1133 - 0.0623i	-0.1133 - 0.0623i
	-0.1133 + 0.0006i	-0.1133 + 0.0006i	-0.1133 + 0.0006i
	-0.1133 - 0.0006i	-0.1133 - 0.0006i	-0.1133 - 0.0006i

Table E.2: Eigenvalues of linearized ROM around OPs

	OP1	OP2	OP3
Order $v = 12$	-0.0272 + 0.9966i	-0.0272 + 0.9966i	-0.0272 + 0.9966i
	-0.0272 - 0.9966i	-0.0272 - 0.9966i	-0.0272 - 0.9966i
	-0.0179 + 0.0840i	-0.0179 + 0.0840i	-0.0179 + 0.0840i
	-0.0179 - 0.0840i	-0.0179 - 0.0840i	-0.0179 - 0.0840i
	-0.0748 + 0.0000i	-0.0748 + 0.0000i	-0.0748 + 0.0000i
	-0.0001 + 0.0000i	-0.0001 + 0.0000i	-0.0001 + 0.0000i
	-0.0272 + 0.9966i	-0.0272 + 0.9966i	-0.0272 + 0.9966i
	-0.0272 - 0.9966i	-0.0272 - 0.9966i	-0.0272 - 0.9966i
	-0.0179 + 0.0840i	-0.0179 + 0.0840i	-0.0179 + 0.0840i
	-0.0179 - 0.0840i	-0.0179 - 0.0840i	-0.0179 - 0.0840i
	-0.0748 + 0.0000i	-0.0748 + 0.0000i	-0.0748 + 0.0000i
	-0.0001 + 0.0000i	-0.0001 + 0.0000i	-0.0001 + 0.0000i

Table E.3: Eigenvalues of linearized ROM around OPs

	OP1	OP2	OP3
Order $v = 12$	-0.1890 + 0.8971i	-0.1663 + 0.9005i	-0.1768 + 0.9163i
	-0.1890 - 0.8971i	-0.1663 - 0.9005i	-0.1768 - 0.9163i
	-0.4092 + 0.0000i	-0.4792 + 0.0000i	-0.3489 + 0.0000i
	-0.0247 + 0.0000i	-0.0280 + 0.0000i	-0.0269 + 0.0000i
	-0.0141 + 0.0000i	-0.0060 + 0.0002i	-0.0156 + 0.0000i
	-0.0052 + 0.0000i	-0.0060 - 0.0002i	-0.0051 + 0.0000i
	0.0347 + 1.0038i	-0.1224 + 0.9673i	-0.0401 + 0.9942i
	0.0347 - 1.0038i	-0.1224 - 0.9673i	-0.0401 - 0.9942i
	-0.0397 + 0.0742i	-0.1536 + 0.0000i	-0.0490 + 0.0559i
	-0.0397 - 0.0742i	-0.0470 + 0.0000i	-0.0490 - 0.0559i
	-0.0513 + 0.0000i	-0.0272 + 0.0000i	-0.0621 + 0.0000i
	-0.0050 + 0.0000i	-0.0051 + 0.0000i	-0.0050 + 0.0000i

Table E.4: Eigenvalues of linearized ROM around OPs

	OP1	OP2	OP3
Order $v = 16$	-0.0758 + 0.9706i	-0.0440 + 0.9825i	-0.1390 + 0.9379i
	-0.0758 - 0.9706i	-0.0440 - 0.9825i	-0.1390 - 0.9379i
	-0.3444 + 0.0000i	-0.3986 + 0.0000i	-0.3514 + 0.0000i
	-0.0050 + 0.1000i	-0.0050 + 0.1000i	-0.0050 + 0.1000i
	-0.0050 - 0.1000i	-0.0050 - 0.1000i	-0.0050 - 0.1000i
	-0.0216 + 0.0000i	-0.0274 + 0.0000i	-0.0273 + 0.0000i
	-0.0186 + 0.0000i	-0.0084 + 0.0000i	-0.0141 + 0.0000i
	-0.0051 + 0.0000i	-0.0054 + 0.0000i	-0.0052 + 0.0000i
	-0.0202 + 0.9981i	-0.0208 + 0.9965i	-0.0254 + 0.9968i
	-0.0202 - 0.9981i	-0.0208 - 0.9965i	-0.0254 - 0.9968i
	-0.0050 + 0.1000i	-0.1580 + 0.0000i	-0.0050 + 0.1000i

	-0.0050 - 0.1000i	-0.0050 + 0.1000i	-0.0050 - 0.1000i
	-0.0346 + 0.0769i	-0.0050 - 0.1000i	-0.0472 + 0.0558i
	-0.0346 - 0.0769i	-0.0346 + 0.0114i	-0.0472 - 0.0558i
	-0.0598 + 0.0000i	-0.0346 - 0.0114i	-0.0661 + 0.0000i
	-0.0050 + 0.0000i	-0.0051 + 0.0000i	-0.0050 + 0.0000i

Table E.5: Eigenvalues of linearized ROM around OPs

	OP1	OP2	OP3
Order $v = 12$	-37.3212 + 0.0000i	-23.2656 + 0.0000i	-24.5452 + 0.0000i
	-0.0010 + 0.0016i	-0.0033 + 0.0033i	-0.0018 + 0.0025i
	-0.0010 - 0.0016i	-0.0033 - 0.0033i	-0.0018 - 0.0025i
	-0.0014 + 0.0000i	-0.0018 + 0.0025i	-0.0015 + 0.0000i
	-0.0018 + 0.0003i	-0.0018 - 0.0025i	-0.0018 + 0.0003i
	-0.0018 - 0.0003i	-0.0018 + 0.0003i	-0.0018 - 0.0003i
	-0.0018 + 0.0001i	-0.0018 - 0.0003i	-0.0018 + 0.0001i
	-0.0018 - 0.0001i	-0.0018 + 0.0003i	-0.0018 - 0.0001i
	-0.0018 + 0.0025i	-0.0018 - 0.0003i	-0.0018 + 0.0025i
	-0.0018 - 0.0025i	-0.0018 + 0.0000i	-0.0018 - 0.0025i
	-0.0018 + 0.0000i	-0.0018 + 0.0000i	-0.0018 + 0.0000i
	-0.0018 - 0.0000i	-0.0018 - 0.0000i	-0.0018 - 0.0000i

Table E.6: Eigenvalues of linearized ROM around OPs

	OP1	OP2	OP3
Order $v = 16$	-34.6215 + 0.0000i	-21.4943 + 0.0000i	-25.9976 + 0.0000i
	-0.0017 + 0.0015i	-0.0017 + 0.0017i	-0.0021 + 0.0014i
	-0.0017 - 0.0015i	-0.0017 - 0.0017i	-0.0021 - 0.0014i

	-0.0019 + 0.0000i -0.0018 + 0.0002i -0.0018 - 0.0002i -0.0018 + 0.0002i -0.0018 - 0.0002i -0.0018 + 0.0002i -0.0018 - 0.0002i -0.0018 + 0.0024i -0.0018 - 0.0024i -0.0018 + 0.0024i -0.0018 - 0.0024i -0.0018 + 0.0024i -0.0018 - 0.0024i	-0.0017 + 0.0000i -0.0018 + 0.0002i -0.0018 - 0.0002i -0.0018 + 0.0002i -0.0018 - 0.0002i -0.0018 + 0.0002i -0.0018 - 0.0002i -0.0018 + 0.0024i -0.0018 - 0.0024i -0.0018 + 0.0024i -0.0018 - 0.0024i -0.0018 + 0.0024i -0.0018 - 0.0024i	-0.0018 + 0.0000i -0.0018 + 0.0002i -0.0018 - 0.0002i -0.0018 + 0.0002i -0.0018 - 0.0002i -0.0018 + 0.0002i -0.0018 - 0.0002i -0.0018 + 0.0024i -0.0018 - 0.0024i -0.0018 + 0.0024i -0.0018 - 0.0024i -0.0018 + 0.0024i -0.0018 - 0.0024i

Table E.7: Eigenvalues of linearized ROM around OPs

	OP1.1	OP1,2	OP1.3	OP2.1	OP2.2	OP2.3	OP3.1	OP3.2	OP3.3
Order $v = 12$	-0.1089 + 0.0000i -0.0000 + 0.0000i -0.0000 + 0.0001i -0.0000 - 0.0001i -0.0000	-2.3878 + 0.0000i -0.0000 + 0.0000i -0.0000 + 0.0001i -0.0000 - 0.0001i -0.0000	-4.0527 + 0.0000i -0.0000 + 0.0000i -0.0000 + 0.0001i -0.0000 - 0.0001i -0.0000	-1.8918 + 0.0000i -0.0000 + 0.0000i -0.0000 + 0.0001i -0.0000 - 0.0001i -0.0000	-1.7948 + 0.0000i -0.0000 + 0.0000i -0.0000 + 0.0000i -0.0000 - 0.0000i -0.0000	-2.0443 + 0.0000i -0.0000 + 0.0000i -0.0000 + 0.0001i -0.0000 - 0.0001i -0.0000	-3.4205 + 0.0000i -0.0000 + 0.0000i -0.0000 + 0.0001i -0.0000 - 0.0001i -0.0000	-1.1147 + 0.0000i -0.0000 + 0.0000i -0.0000 + 0.0000i -0.0000 - 0.0000i -0.0000	-6.5274 + 0.0000i -0.0000 + 0.0000i -0.0000 + 0.0001i -0.0000 - 0.0001i -0.0000

	+ 0.0001 i - 0.0000 - 0.0001 i - 8.4616 + 0.0000 i - 0.0000 + 0.0000 i - 0.0000 + 0.0001 i - 0.0000 - 0.0001 i - 0.0000 + 0.0001 i - 0.0000 - 0.0001 i	+ 0.0001 i - 0.0000 - 0.0001 i - 3.4585 + 0.0000 i - 0.0000 + 0.0000 i - 0.0000 + 0.0001 i - 0.0000 - 0.0001 i - 0.0000 + 0.0001 i - 0.0000 - 0.0001 i	+ 0.0001 i - 0.0000 - 0.0001 i - 2.8726 + 0.0000 i - 0.0000 + 0.0000 i - 0.0000 + 0.0001 i - 0.0000 - 0.0001 i - 0.0000 + 0.0001 i - 0.0000 - 0.0001 i	+ 0.0001 i - 0.0000 - 0.0001 i - 3.6807 + 0.0000 i - 0.0000 + 0.0000 i - 0.0000 + 0.0001 i - 0.0000 - 0.0001 i - 0.0000 + 0.0001 i - 0.0000 - 0.0001 i	+ 0.0000 i - 0.0000 - 0.0000 i - 1.1660 + 0.0000 i - 0.0000 + 0.0000 i - 0.0000 + 0.0000 i - 0.0000 - 0.0000 i - 0.0000 + 0.0000 i - 0.0000 - 0.0000 i	+ 0.0001 i - 0.0000 - 0.0001 i - 3.1139 + 0.0000 i - 0.0000 + 0.0000 i - 0.0000 + 0.0001 i - 0.0000 - 0.0001 i - 0.0000 + 0.0001 i - 0.0000 - 0.0001 i	+ 0.0001 i - 0.0000 - 0.0001 i - 0.8440 + 0.0000 i - 0.0000 + 0.0000 i - 0.0000 + 0.0001 i - 0.0000 - 0.0001 i - 0.0000 + 0.0001 i - 0.0000 - 0.0001 i	+ 0.0000 i - 0.0000 - 0.0000 i - 0.1911 + 0.0000 i - 0.0000 + 0.0000 i - 0.0000 + 0.0000 i - 0.0000 - 0.0000 i - 0.0000 + 0.0000 i - 0.0000 - 0.0000 i	+ 0.0001 i - 0.0000 - 0.0001 i - 6.5452 + 0.0000 i - 0.0000 + 0.0000 i - 0.0000 + 0.0001 i - 0.0000 - 0.0001 i - 0.0000 + 0.0001 i - 0.0000 - 0.0001 i

For these eigenvalues, there is a scale factor of $10^4$for OP 3.2, and 10^3 for all the others OPs.

Table E.8: Eigenvalues of linearized ROM around OPs

List of Figures

List of Tables

Acknowledgements

I want to express my deepest gratitude to my mother Simona, to my uncles, Paolo and Piergiorgio, to my friend Stefano, and to my family for their constant support, encouragement, and understanding throughout my academic journey. Their presence and trust have been a fundamental source of motivation, without which this work would not have been possible. I would like to sincerely thank my supervisor, Professor Francesco Casella, and my co-supervisor, Matteo Luigi De Pascali, for their guidance, availability, and valuable scientific advice during the development of this thesis. Their expertise and constructive feedback were essential in shaping both the methodological and technical aspects of this work. I also gratefully acknowledge the Politecnico di Milano, and in particular the Department of Electronics Information and Bioengineering (DEIB), for providing the high-fidelity models used as test cases for the model order reduction activities presented in this thesis. My sincere thanks go to Imperial College London, and specifically to the Department of Electrical and Electronic Engineering (EEE), for hosting me during my research period. I am especially grateful to Professor Thomas Parisini and Alessio Moreschini, my supervisors at Imperial College London, for the continuous support, insightful discussions, and scientific guidance provided during my stay. I would also like to thank my colleagues at Imperial College London, in particular Matteo Aicardi, Riccardo Grimaldi, and Weirui Chen, for the stimulating research environment and for the many fruitful technical exchanges. A special and heartfelt thanks to Debraj Bhattacharjee, a colleague at Imperial College, who generously shared part of the MATLAB tool for moment matching model order reduction developed as part of their research. Without this tool, a substantial part of the work presented in this thesis would not have been possible. Finally, I would also like to acknowledge a brief yet meaningful encounter during my time in London, with a person I met by chance on a bus. Although our conversation lasted only a few minutes, it left a positive impression and a sense of regret for not having been able to continue the discussion due to my

own hesitation. This work is also a reminder of how even short exchanges can leave a lasting mark.