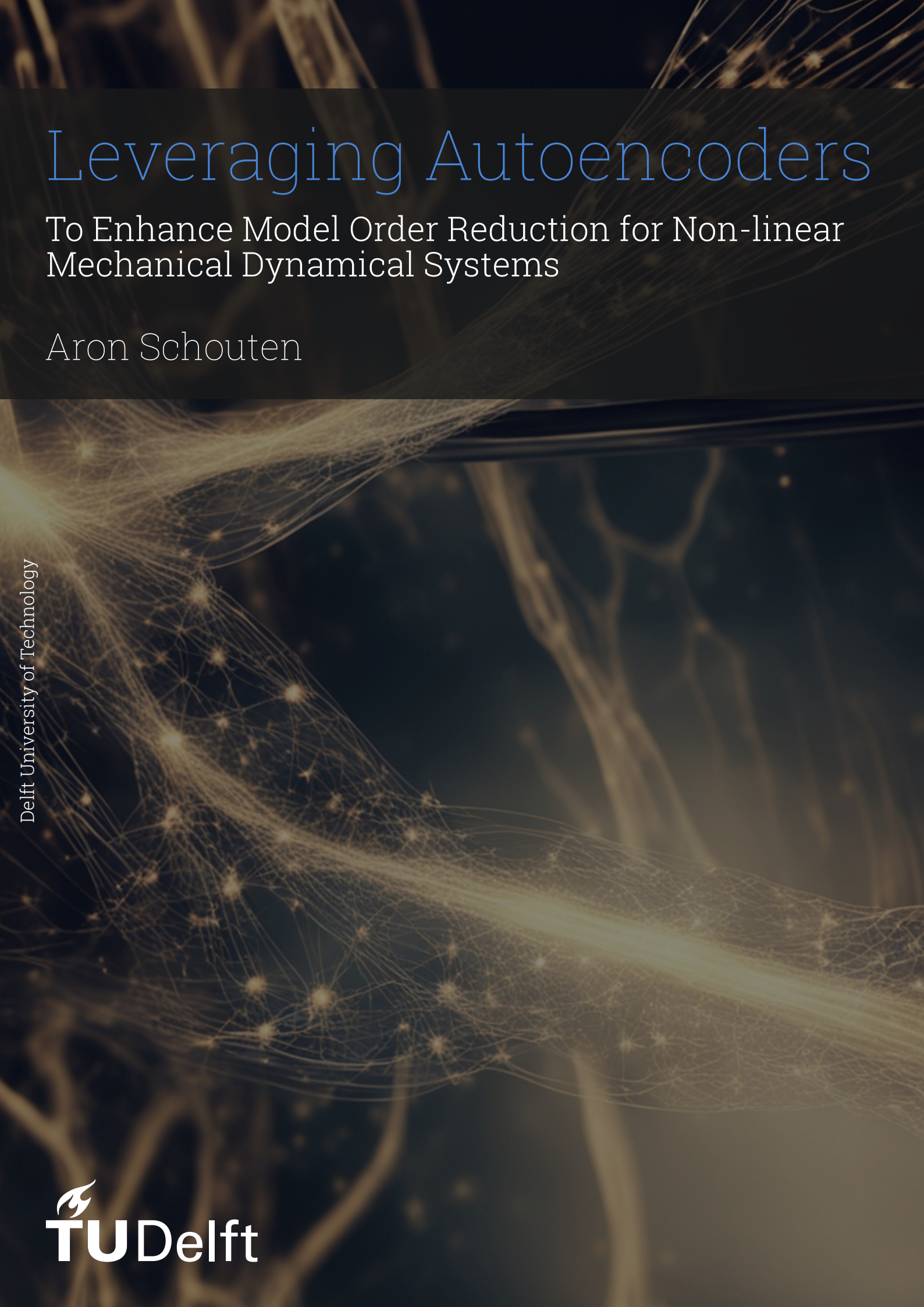
Leveraging Autoencoders

To Enhance Model Order Reduction for Non-linear Mechanical Dynamical Systems

Aron Schouten



Delft University of Technology

TU Delft

Leveraging Autoencoders

To Enhance Model Order Reduction for Non-linear Mechanical Dynamical Systems

by

Aron Schouten

to obtain the degree of Master of Science
in Applied Mathematics (Computational Science & Engineering)
at the Delft University of Technology, faculty EEMCS,
to be defended on Friday, October 18, 2024 at 3:30 PM.

Student number:	4999169	
Project duration:	December 18, 2023 – October 18, 2024	
Thesis committee:	Prof.dr.ir. M.B. van Gijzen (chair),	TU Delft, Numerical Analysis
	Dr.ir. S. Jain (supervisor),	TU Delft, Numerical Analysis
	Dr.ir. W.T. van Horssen	TU Delft, Mathematical Physics

An electronic version of this thesis is available at http://repository.tudelft.nl/.

TU Delft

Preface

You are about to read the thesis 'Leveraging Autoencoders to Enhance Model Order Reduction for Non-linear Mechanical Dynamical Systems'. This thesis was written as part of my graduation from the Master's program in Applied Mathematics at the Delft University of Technology. From December 2023 to October 2024, I conducted research and composed this thesis. The study explores advancements in model order reduction (MOR) techniques, with a specific focus on the utilization of autoencoders for non-linear dynamical systems. The research addresses the demand for more efficient computational methodologies in engineering and scientific simulations, especially in scenarios where high-dimensional dynamical models present computational challenges. By proposing a novel technique, and through extensive testing and comparison against established techniques, this research demonstrates the potential of integrating autoencoders with established MOR methods to enhance the accuracy of MOR.

I would like to thank the members of my thesis committee: Martin van Gijzen, Shobhit Jain, and Wim van Horssen, with a special thanks to Shobhit, my supervisor. His expertise and insights deepened my understanding of the topic and guided my research into new areas. The feedback he provided during our discussions was crucial in shaping the direction of this thesis. I am grateful for the time he committed to reviewing my progress and for his thoughtful suggestions, which greatly improved the quality of my work.

Enjoy reading.

Aron Schouten
The Hague, October 2024

Abstract

The computational demands of finite element simulations, particularly in predicting the time-dependent response of high-dimensional non-linear dynamical systems, pose significant challenges. To overcome these challenges, researchers have developed model order reduction (MOR) methods, which aim to reduce computational complexity by utilizing lower-dimensional models. This thesis proposes a MOR technique that simultaneously learns both the projection to, and the reduced dynamics on, a lower-dimensional manifold using autoencoders, a type of neural network. During training, the known linear part of the reduced dynamics is used to aid the optimization process, leading to an effective method of simultaneous projection and linear informed training (SPLIT). SPLIT demonstrates outstanding performance on the test case of a 2D cantilever beam, and is capable of making non-linear forced response predictions, even though being trained on unforced decaying trajectories. Even in scenarios involving highly non-linear behaviour, such as when the beam folds over itself, SPLIT continues to make accurate predictions, while other MOR techniques fail. This work highlights the potential of autoencoders to advance the field of MOR and improve the efficiency and reliability of simulations for complex dynamical systems.

Contents

Nomenclature

Abbreviations

Abbreviation	Definition
CNN	Convolutional Neural Network
DMD	Dynamic Mode Decomposition
ELU	Exponential Linear Unit
FE	Finite Element
FOM	Full Order Model
MOR	Model Order Reduction
MSE	Mean Squared Error
NLDR	Non-linear Dimensionality Reduction
NMTE	Normalized Mean-Trajectory-Error
ODE	Ordinary Differential Equation
OLS	Ordinary Least Squares
PCA	Principal Component Analysis
PDE	Partial Differential Equation
POD	Proper Orthogonal Decomposition
ROM	Reduced Order Model
SINDy	Sparse Identification of Non-linear Dynamics
SPLIT	Simultaneous Projection and Linear Informed Training
SSM	Spectral SubManifold

Symbols

Symbol	Definition
$\boldsymbol{A}$	State-transition matrix of first-order FOM
$\boldsymbol{b}$	Bias vector of layer from encoder or decoder
$\boldsymbol{C}$	Damping matrix of FOM
d	Dimension of reduced space
$\boldsymbol{f}^{ext}$	External forcing function of FOM
$\boldsymbol{f}^{int}$	Non-linear internal forcing function of FOM
$\boldsymbol{f}_0$	Internal forcing function of first-order FOM
$\boldsymbol{f}_1$	External forcing function of first-order FOM
h	Time-step of the time-integration
$\boldsymbol{I}$	Identity matrix
$\boldsymbol{K}$	Stiffness matrix of FOM

Symbol	Definition
l	Number of candidates in library function
$\mathcal{L}$	Loss term
m	Number of time-steps taken
$\boldsymbol{M}$	Mass matrix of FOM
n	Dimension of full space
p	Degree of ROM
$\boldsymbol{q}$	Displacement vector of FOM
$\boldsymbol{Q}$	Matrix with data of time-series of $\boldsymbol{q}$
$\boldsymbol{r}$	Function describing reduced dynamics
$\boldsymbol{r}_1$	Time-dependent part of reduced dynamics
$\boldsymbol{r}_{nl}$	Non-linear part of reduced dynamics
$\boldsymbol{R}$	Coefficients of polynomial regression
$\boldsymbol{R}_0$	Linear part of reduced dynamics
T	Period of eigenfrequency or external forcing frequency
$\boldsymbol{v}$	Decoder function
$\boldsymbol{v}_1$	Time-dependent part of decoder
$\boldsymbol{v}_{nl}$	Non-linear part of decoder
$\boldsymbol{V}_0$	Linear part of decoder
$\boldsymbol{w}$	Encoder function
$\boldsymbol{W}$	Weight matrix of layer from encoder or decoder
$\boldsymbol{W}_0$	Linear part of encoder
$\boldsymbol{x}$	Displacement and velocity vector of first-order FOM
$\boldsymbol{y}$	Output of a layer from encoder or decoder
$\boldsymbol{z}$	Displacement (vector) of ROM
$\boldsymbol{Z}$	Matrix with data of time-series of $\boldsymbol{z}$
$\boldsymbol{\alpha}$	Function of mapped external forcing function
β	Parameter of Newmark-beta integration scheme
γ	Parameter of Newmark-beta integration scheme
ε	External forcing amplitude parameter
ζ	Damping ratio
$\boldsymbol{\Theta}$	Library of candidate functions for SINDy or SPLIT
λ	Weight parameter
$\boldsymbol{\Xi}$	Coefficients of SINDy or SPLIT
σ	Activation function of layer from encoder or decoder
τ	Point in time where last period starts
$\boldsymbol{\varphi}$	Eigenmode
$\boldsymbol{\Phi}$	Library of monomials for polynomial regression
ω	Undamped natural frequency
$\boldsymbol{\Omega}$	Forcing frequency
∇	Directional derivative operator
Σ	Summation operator
$\|\| \cdot \|\|_p$	p - norm

1 Introduction

This introductory chapter presents an overview of the main research themes. The chapter first highlights the context of the research and reviews the current relevant literature. The chapter then discusses the contributions of the study, presents the research questions, and concludes with an outline of the remaining chapters of the report.

1.1. High-Dimensional Mechanics Problems

This report focuses on the dynamical systems arising from high-dimensional mechanics problems, which are commonly encountered in various engineering applications. These problems are typically governed by partial differential equations (PDEs) which can be spatially discretized using the finite element (FE) method. The discretization of the PDEs results in a system of second-order ordinary differential equations (ODEs) for the displacement vector $\boldsymbol{q}(t) \in \mathbb{R}^n$, expressed as

$$\boldsymbol{M}\ddot{\boldsymbol{q}} + \boldsymbol{C}\dot{\boldsymbol{q}} + \boldsymbol{K}\boldsymbol{q} + \boldsymbol{f}^{int}\left(\boldsymbol{q}, \dot{\boldsymbol{q}}\right) = \varepsilon \boldsymbol{f}^{ext}\left(\Omega t\right). \tag{1.1}$$

Here $\boldsymbol{M}$, $\boldsymbol{C}$, $\boldsymbol{K} \in \mathbb{R}^{n \times n}$ are the mass, damping and stiffness matrices, $\boldsymbol{f}^{int}\left(\boldsymbol{q}, \dot{\boldsymbol{q}}\right) \in \mathbb{R}^n$ is the purely non-linear internal force and $\boldsymbol{f}^{ext}\left(\Omega t\right) \in \mathbb{R}^n$ is the external force, which is periodic in time, with frequency Ω. The parameter ε is used to control the amplitude of the external forcing.

These FE models are crucial tools in both scientific and engineering fields, providing valuable insights into complex systems. However, the computational demands of FE simulations, particularly in predicting the time-dependent response of non-linear, high-dimensional dynamical systems, pose significant challenges. Even with dedicated commercial software, these simulations often encounter high computational costs [32].

To address these computational challenges, researchers have developed model order reduction (MOR) methods. These methods aim to reduce the computational complexity of mathematical models in numerical simulations [43]. In these MOR methods, the behaviour of full order models (FOMs) is approximated using lower-dimensional reduced order models (ROMs), to significantly lower computational cost.

ROMs are essential in scenarios where performing simulations using the FOM is impractical due to limitations in computational resources or when doing demanding simulations, for example real-time simulations [3].

For small displacements, the FE model from equation (1.1) exhibits linear behaviour. However, as displacements grow larger, non-linear effects become more pronounced, leading to more complex dynamics. In extreme cases, large deformations result in highly non-linear responses, which pose significant challenges when attempting to reduce the model to a simplified representation, thereby complicating the process of MOR. This thesis proposes a MOR technique that continues to provide accurate predictions even under highly non-linear behaviour.

1.2. Model Order Reduction

MOR techniques can be classified into two categories: intrusive and non-intrusive methods [54]. Intrusive methods directly manipulate the governing equations of the system to derive reduced models. While these methods can yield effective ROMs, they necessitate prior knowledge of the underlying equations of the model. In situations where such equations are not available, for example when utilizing (commercial) software packages, these methods may present challenges [4]. On the other hand, non-intrusive methods, also known as data-driven methods, utilize the output data of a model to construct a ROM [4]. This makes them applicable in scenarios where the underlying equations are unknown and inaccessible. This advantage, along with others, such as the increasing availability of large datasets from simulations and experiments [41], has led to recent interest in developing fully data-driven approaches that do not require access to full order model operators to establish ROMs, or data-assisted approaches, where both data and the governing equations of the system are used to derive a ROM.

The development of a MOR method typically consists of two components: determining a mapping to a reduced space and identifying the dynamics within that reduced space. The first step, known as dimensionality reduction, relies on the observation that in high-dimensional systems, not all variables are required to capture the core dynamics of the system. As a result, the high-dimensional system state, $\boldsymbol{q}(t) \in \mathbb{R}^n$, can be mapped to a reduced set of coordinates, $\boldsymbol{z}(t) \in \mathbb{R}^d$, where $d < n$, that retains only the essential dynamics of the system. This can be written as

$$\boldsymbol{z}(t) = \boldsymbol{w}(\boldsymbol{q}(t)),$$

where $\boldsymbol{w} : \mathbb{R}^n \to \mathbb{R}^d$ represents the mapping to the reduced space. Once the reduced coordinates are obtained, the second step involves determining the dynamics in this reduced space. The goal is to approximate the system its behaviour using a reduced order model

$$\dot{\boldsymbol{z}}(t) = \boldsymbol{r}(\boldsymbol{z}(t)).$$

Here, $\boldsymbol{r} : \mathbb{R}^d \to \mathbb{R}^d$ represents the reduced dynamics, which can either be derived from the underlying physical model (in the case of intrusive methods) or learned from data (in non-intrusive methods). These reduced dynamics can be utilized to make

predictions in the reduced space, which are generally computationally less expensive to obtain than predictions using the full order model. These predictions in the reduced space can be mapped back to the full space using

$$\boldsymbol{q}(t) = \boldsymbol{v}(\boldsymbol{z}(t)),$$

where $\boldsymbol{v} : \mathbb{R}^d \to \mathbb{R}^n$ represents the mapping back to the full space. The following subsections discuss the current relevant literature on both the dimensionality reduction and the discovery of reduced dynamics.

1.2.1. Dimensionality Reduction

The first step of MOR, dimensionality reduction, can be done using both linear and non-linear approaches. Linear methods reduce the dimensionality of the data using projections to a linear subspace. Non-linear techniques can capture more complex patterns and variations by learning a mapping to a non-linear manifold.

1.2.1.1. Linear Dimensionality Reduction

An example of an established intrusive linear dimensionality reduction technique involves using modal projection. It is based on the observation that, for many structures, the lower-frequency modes dominate the dynamic response, especially in lightly damped systems [49].

If, for instance, the dynamic behaviour is primarily influenced by the first eigenmode $\boldsymbol{\varphi}_1 \in \mathbb{R}^n$, the system its response can be approximated using this mode. The eigenmode can be obtained by solving the undamped eigenvalue problem

$$\boldsymbol{K}\boldsymbol{\varphi}_1 = \omega_1^2 \boldsymbol{M}\boldsymbol{\varphi}_1.$$

Here $\boldsymbol{K}$ and $\boldsymbol{M}$ are the stiffness and mass matrices from equation (1.1), and ω_1 is the (undamped) natural frequency associated with the first eigenmode $\boldsymbol{\varphi}_1$. When $\boldsymbol{M}$ is positive definite, the eigenmode $\boldsymbol{\varphi}_1$ is typically mass normalized such that

$$\boldsymbol{\varphi}_1^\top \boldsymbol{M}\boldsymbol{\varphi}_1 = 1.$$

The high-dimensional model $\boldsymbol{q}(t) \in \mathbb{R}^n$ can be reduced to a lower-dimensional model $z(t) \in \mathbb{R}$, by projecting the full-order system onto the subspace spanned by $\boldsymbol{\varphi}_1$ (a so-called spectral subspace). This is done using

$$z(t) = \boldsymbol{\varphi}_1^\top \boldsymbol{M} \cdot \boldsymbol{q}(t).$$

In this subspace spanned by $\boldsymbol{\varphi}_1$, some reduced dynamics describe the response of $z(t)$. These reduced dynamics can be utilized to make predictions in the reduced space, which can be mapped back to the full space using the equation

$$\boldsymbol{q}(t) = \boldsymbol{\varphi}_1 \cdot z(t).$$

Note that this way,

$$z(t) = \boldsymbol{\varphi}_1^\top \boldsymbol{M} \cdot \boldsymbol{q}(t) = \underbrace{\boldsymbol{\varphi}_1^\top \boldsymbol{M}\boldsymbol{\varphi}_1}_{=1} \cdot z(t) = z(t).$$

An established non-intrusive linear dimensionality reduction technique is proper orthogonal decomposition (POD) [2]. POD identifies spatial structures that capture the most variance in the time-series data of the FE model using principal component analysis (PCA). By decomposing the system into orthogonal modes, POD can efficiently reduce the dimensionality of the data while preserving significant features [29]. POD can hence be considered a data-driven variant of modal projection, enabling a lower-dimensional representation that retains essential system behaviour, but now derived directly from data rather than from the governing equations.

POD may however not be optimal for modeling dynamical systems as its modes are independent of the time evolution or dynamics encoded in the data [19]. This limitation arises because POD focuses solely on spatial information and does not consider how the modes evolve over time, potentially missing crucial dynamic interactions. As a result, the reduced model may fail to accurately capture transient behaviour and temporal dynamics that could be critical in some applications.

To address these shortcomings, methods such as dynamic mode decomposition (DMD) have been developed. DMD offers a more robust framework for analyzing the time evolution of dynamical systems by directly linking spatial modes with their temporal behaviour [44]. Unlike POD, DMD identifies modes that evolve according to a linear dynamic system, making it particularly effective for capturing both spatial and temporal dynamics. This approach allows for the extraction of dynamic features, such as oscillatory behaviour or growth/decay patterns, which are essential for understanding complex systems. POD and DMD are however not in the scope of this thesis and will not be utilized any further.

1.2.1.2. Non-linear Dimensionality Reduction

Linear dimensionality reduction techniques generally fall short in accurately capturing the relevant behaviour of more complex non-linear systems [37]. To overcome this limitation, researchers have delved into non-linear techniques [34]. Non-linear dimensionality reduction (NLDR), which is also known as manifold learning, encompasses various techniques aimed at projecting high-dimensional data onto lower-dimensional latent manifolds. This approach differs from the projection-based methods like modal projection and POD, where the generated mappings do not correspond to a manifold, but a linear subspace.

An approach based in dynamical systems theory offers MOR using spectral submanifolds (SSMs). An SSM is essentially the smoothest possible non-linear manifold that captures the behaviour of the non-linear system, which asymptotically behaves like a spectral subspace [20]. Using these SSMs, an intrusive, mathematically rigorous non-linear MOR technique for very high-dimensional problems can be obtained [24]. The numerical implementation of this methodology, which is called SSMTool, is available in an open-source MATLAB package [26].

Building on the SSM-based framework of SSMTool, the intrusive SSMLearn has emerged as a data-assisted alternative for constructing SSMs [9]. SSMLearn seeks to learn SSMs directly from data using polynomial regression in a graph-style parametrization, enabling MOR for high-dimensional numerical datasets as well as experimental measurements.

Remarkably, SSM reductions trained on unforced data have also demonstrated the ability to accurately predict non-linear responses when additional external forcing is applied, as the SSMs persist under small-amplitude forcing [8]. The graph-style parametrization however breaks down when the manifold folds over the graphing subspace. This scenario can occur, for instance, when a cantilevered beam folds over itself during large-amplitude oscillations along a two-dimensional SSM corresponding to the beam its fundamental vibration mode [7].

Recently many advancements in artificial intelligence and neural networks (NNs) have been made. These NNs, inspired by the workings of the human brain, also lead to the development of new MOR techniques, as they are capable of processing vast datasets to recognize non-linear patterns [33]. Autoencoders are a type of NN that can be used for NLDR [22]. They can be utilized for MOR by projecting the high-dimensional data of a dynamical system onto a lower-dimensional manifold. While in general, autoencoders result in non-linear techniques, it can be shown that under specific conditions, the use of autoencoders can be equivalent to PCA [1]. This indicates that autoencoders serve as a non-linear extension of linear dimensionality reduction.

An autoencoder consists of two main components: an encoder $\boldsymbol{w} : \mathbb{R}^n \to \mathbb{R}^d$ and a decoder $\boldsymbol{v} : \mathbb{R}^d \to \mathbb{R}^n$. The encoder maps the high-dimensional time-series data of $\boldsymbol{q}(t)$ to a lower-dimensional latent space with some reduced model $\boldsymbol{z}(t)$. The decoder attempts to reconstruct the original data from this reduced representation, resulting in the reconstruction $\hat{\boldsymbol{q}}(t) \approx \boldsymbol{q}(t)$. This is illustrated in Figure 1.1. The network adjusts its weights and biases to minimize the reconstruction error, defined as the mean squared

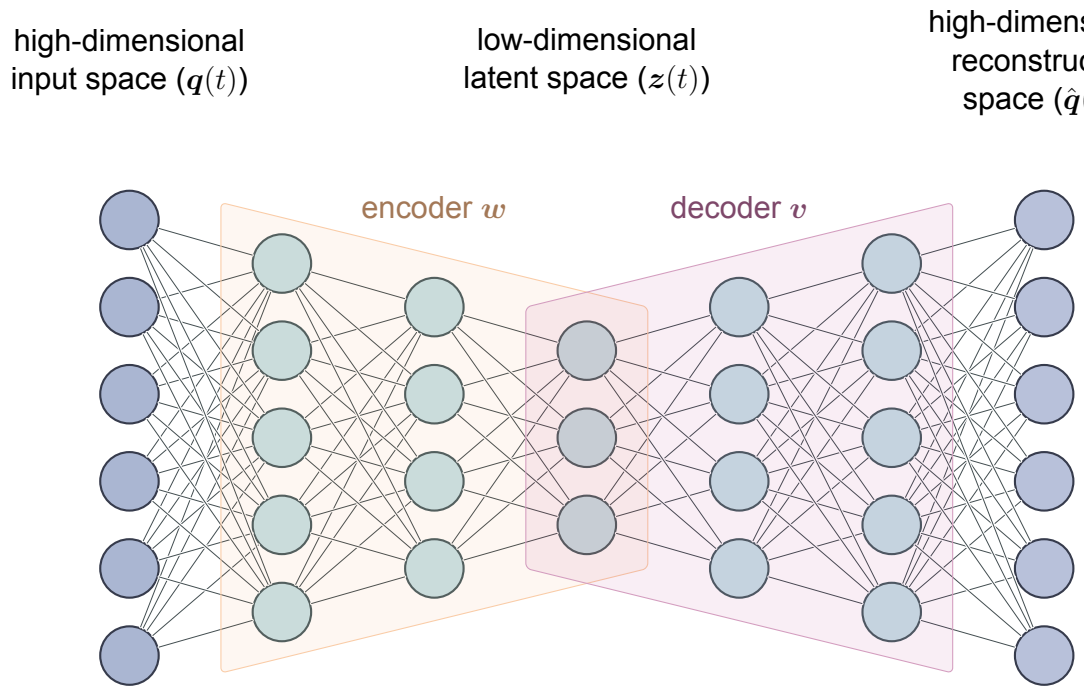


Figure 1.1: Example of an autoencoder consisting of an encoder $\boldsymbol{w}$ and a decoder $\boldsymbol{v}$. The encoder maps the high-dimensional input data from $\boldsymbol{q}(t)$ to a lower-dimensional latent space with $\boldsymbol{z}(t)$, while the decoder attempts to reconstruct the original data from this reduced representation, resulting in $\hat{\boldsymbol{q}}(t)$.

error between the original input and its reconstruction, i.e.

$$\mathcal{L}\left(\boldsymbol{q}, \hat{\boldsymbol{q}}\right) = ||\boldsymbol{q} - \hat{\boldsymbol{q}}||_2^2 .$$

This process, known as training, is done by iterating over the available data. Several studies have demonstrated the efficacy of using autoencoders for model order reduction [40, 47], even in complex situations, for example systems with unsteady flow [12].

Moreover, ongoing research explore various extensions of autoencoders to enhance their capabilities in MOR. The most common extension involves the use of convolutional autoencoders, which leverage convolutional neural networks (CNNs) to capture spatial dependencies within the data [18, 42]. CNNs, commonly used in image processing tasks, excel at extracting hierarchical features, making convolutional autoencoders well-suited for MOR tasks involving spatially distributed systems. The CNNs however extend beyond the scope of this thesis and will therefore not be utilized.

1.2.2. Discovery of Reduced Dynamics

In addition to determining a mapping to the latent space (whether it be a linear subspace or a non-linear manifold), the dynamics associated with this reduced representation must be learned as well. These reduced dynamics are crucial for describing the time evolution of the system within the lower-dimensional space. Without accurate reduced dynamics, a ROM may fail to reproduce the correct time evolution.

Specifically, the model in reduced coordinates, denoted as $\boldsymbol{z}(t)$, can be represented by a system of ODEs, as

$$\dot{\boldsymbol{z}}(t) = \boldsymbol{r}(\boldsymbol{z}(t)),$$

where $\boldsymbol{r} : \mathbb{R}^d \to \mathbb{R}^d$ is the function that describes the reduced dynamics. The challenge is to find an appropriate form of the function $\boldsymbol{r}$ that accurately reflects the dynamics in the reduced space.

A common method for approximating (reduced) dynamics is polynomial regression [14]. In this approach, the reduced dynamics are approximated using a linear combination of monomials [36]. Using a coefficient matrix $\boldsymbol{R} \in \mathbb{R}^{d \times l}$ and a library of monomials up to degree p, expressed as

$$\boldsymbol{\Phi}(\boldsymbol{z}(t)) = \begin{pmatrix} z_1(t) & z_2(t) & \cdots & z_d(t) & z_1^2(t) & z_1(t)z_2(t) & \cdots & \cdots & z_d^p(t) \end{pmatrix},$$

the reduced dynamics can be approximated using the system of ODEs

$$\dot{\boldsymbol{z}}(t) = \boldsymbol{R}\, \boldsymbol{\Phi}(\boldsymbol{z}(t)).$$

The coefficient matrix $\boldsymbol{R}$ is determined through regression, for example using ordinary least squares [36]. By including higher-order terms, polynomial regression can model non-linear interactions in the reduced space. The advantage of polynomial regression is its simplicity and the ability to capture basic non-linearity's. However, it has limitations for more complex systems, especially when the true dynamics are highly non-linear and cannot be well approximated by polynomials of low degree [11].

An alternative method for identifying reduced dynamics is using sparse identification of non-linear dynamics (SINDy) [5]. The approach is similar to polynomial regression, but the library of candidate functions may now also contain more complex terms, for example trigonometric functions. The key idea in SINDy is that most of the coefficients of $\boldsymbol{R}$ will be zero, resulting in a sparse model that selects only a few important terms to describe the dynamics. The selection of the terms is typically done using a sparsity promoting regression. SINDy is particularly useful when the dynamics are complex and cannot be easily represented by simple polynomials. By selecting only the most relevant terms, it produces parsimonious models that are both interpretable and efficient [10].

A powerful MOR technique involves integrating SINDy with autoencoders to jointly learn the mapping to the latent space and the reduced dynamics. Known as SINDy Autoencoders, this framework uses the autoencoder to capture the non-linear mapping from the high-dimensional input space to the low-dimensional latent space, while SINDy operates within this latent space to identify the reduced dynamics [10]. The aim is to establish one single optimization process that simultaneously adjusts the autoencoder parameters and the coefficients of the sparse dynamics.

Recent advancements in machine learning have also introduced novel methods for fitting dynamical systems to data, leveraging the power of NNs and other learning algorithms. For example, long short-term memory networks are a type of NN designed to handle sequential data and capture long-term dependencies. They are particularly effective for time-series prediction and sequence modeling, making them suitable and widely used for fitting dynamical systems to data [12, 18, 40, 47]. These results however extend beyond the scope of this thesis and will therefore not be explored or utilized any further.

1.3. Contribution and Research Outline

This thesis explores how autoencoders can improve MOR for non-linear dynamical systems. To achieve this, a novel MOR method is developed, incorporating the data-assisted discovery of attracting invariant manifolds, enabling rigorous MOR from a dynamical systems perspective. The method leverages unforced trajectory data to learn these manifolds.

The learning of the manifold and the discovery of the reduced dynamics is done in a joint style, inspired by the SINDy Autoencoders [10]. Similar to the SINDy Autoencoders, one single optimization process is used to simultaneously learn both the manifold and its reduced dynamics. The primary distinction of the proposed approach lie in the elimination of sparsity constraints. Unlike the original SINDy Autoencoder methodology, which emphasizes on constructing a sparse reduced model, the technique presented here does not prioritize sparsity. This adjustment allows the model to potentially capture more complex interactions and non-linearity's, providing a richer representation of the system dynamics.

Furthermore, by building on concepts from SSMLearn [8], the linear terms of the reduced dynamics are incorporated in the optimization, to assist the identification the ROM. When the linear terms of the FOM are known from the FE model, the linear terms

for the ROM can be derived using modal projection. Including these linear terms in the ROM discovery process ensures that the underlying linear behaviour is accurately represented, allowing the non-linear terms to focus on capturing deviations from linearity. Unlike SSMLearn, which uses a graph-based parameterization to obtain a manifold, the proposed approach employs an autoencoder, enabling the representation of manifolds that can fold over the graphing subspace.

The proposed method enables the prediction of non-linear forced responses, despite being trained on unforced data, as long as the resulting displacements are within the training displacements. This capability, which arises from its foundation in dynamical systems theory and its use of deep learning to approximate spectral submanifolds that persist under small-amplitude forcing [8], is a feature that is not achievable with SINDy Autoencoders [10].

The proposed method will be referred to as simultaneous projection and linear informed training (SPLIT). To evaluate SPLIT and demonstrate the limitations of existing approaches, a model of a 2D cantilever beam is employed. The non-linear behaviour of this beam serves as an excellent test case for assessing the performance of MOR techniques [30, 46, 50]. Moreover, this test case highlights the shortcomings of current methods, such as the folding of the manifold when the beam undergoes large deflections and bends over itself, an instance where approaches like SSMLearn fail [7]. The implementation of SPLIT and the test case can be found on GitHub [45].

This research aims to answer the following research questions and sub-questions.

How can autoencoders enhance non-intrusive model order reduction for non-linear dynamical systems described by high-dimensional finite element models?

- *What are the limitations of the established data-driven and data-assisted techniques for model order reduction?*
- *Are autoencoders able to correctly identify the non-linear manifolds that dominate the systems its response and learn their reduced dynamics?*
- *How well can the reduced dynamics on these manifolds make predictions for unseen data?*

The rest of this thesis is structured as follows: First, chapter 2 presents the methodologies utilized throughout this study. It introduces the 2D cantilever beam test case, describes various established MOR techniques, and explains the proposed method SPLIT, along with an extension called SPLIT+. Then chapter 3 details the experimental setup and results. It discusses the parameters used for data collection and model training, and compares the performance of the proposed methods against existing techniques. A critical discussion of the findings is provided in chapter 4. The chapter analyzes the observed results in the context of theoretical expectations, and identifies potential areas for further research and improvement. Finally chapter 5 concludes the thesis by summarizing the key insights gained from the research. It revisits the research questions, presents final answers, and highlights the contributions of this work to the field of model order reduction.

2

Methods

This chapter contains the methodologies that will be used in this study. First, the test case of this study is introduced: A 2D cantilever beam. Next, six different model order reduction (MOR) techniques are presented. Finally, the chapter concludes with a discussion of the evaluation metrics applied throughout the report.

2.1. Test Case: A 2D Cantilever Beam

While the methodologies presented in this study apply to general high-dimensional mechanics problems described by equation (1.1), this report utilizes a two-dimensional continuum-based finite element model of a geometrically non-linear beam as a test case. Cantilever beams are a widely used structural component in engineering applications, spanning from buildings and bridges to aerospace engineering [15]. The cantilever beam, fixed at one end and free at the other, is designed to withstand loads applied perpendicular to its length. The non-linear behaviour of this beam makes it an ideal test case for evaluating the performance of MOR techniques [30, 46, 50].

The finite element (FE) method is employed to obtain a numerical model for the beam. The FE model is built using the open-source package YetAnotherFEcode [25]. The general form of the FE model for the cantilever beam is given by a system of second-order ordinary differential equations (ODEs) for the displacement vector $\boldsymbol{q}(t) \in \mathbb{R}^n$, similar to equation (1.1), expressed as

$$\boldsymbol{M}\ddot{\boldsymbol{q}} + \boldsymbol{C}\dot{\boldsymbol{q}} + \boldsymbol{K}\boldsymbol{q} + \boldsymbol{f}^{int}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \varepsilon \boldsymbol{f}^{ext}(\Omega t)\,.$$

Here $\boldsymbol{M}, \boldsymbol{C}, \boldsymbol{K} \in \mathbb{R}^{n\times n}$ are the mass, damping and stiffness matrices, $\boldsymbol{f}^{int}(\boldsymbol{q}, \dot{\boldsymbol{q}}) \in \mathbb{R}^n$ is the purely non-linear internal force and $\boldsymbol{f}^{ext}(\Omega t) \in \mathbb{R}^n$ is the external force, which will be periodic with frequency Ω. The parameter ε is used to control the amplitude of the external forcing.

If the displacement of the beam is small enough, the model will exhibit linear behaviour. However, as the beam deflects further, it will start to behave more and more non-linear. If the beam is displaced far enough, it will bend over itself. This will result in very non-linear dynamics, which will be challenging to accurately capture in a reduced order model (ROM). In particular, a manifold describing this folding behaviour will bend

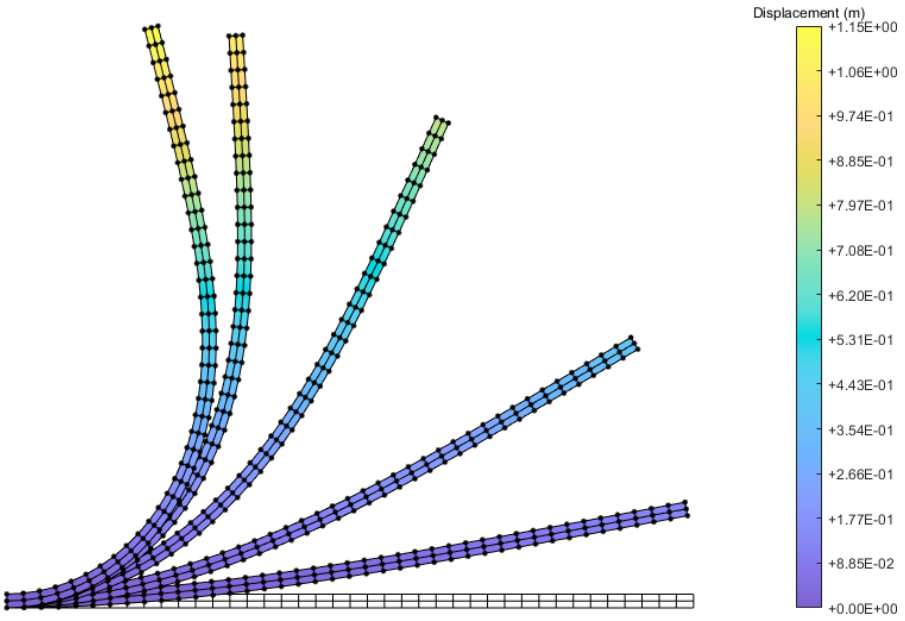


Figure 2.1: Displacement of a 2D cantilever beam in five different configurations. The colour coding indicates the displacement magnitude, with yellow representing maximum displacement and blue representing minimal displacement. The figure contains two beams in a folding configuration. Having a displacement this large leads to highly non-linear behaviour in the dynamical system.

over itself, causing the graph-style parameterization method, used in e.g. SSMLearn, to fail [7]. An example of a cantilever beam with five different displacements is shown in Figure 2.1. The figure contains two beams in a folding configuration. The displacement field is colour-coded, with areas of maximum displacement shown in yellow and areas of minimal displacement shown in blue.

With the 2D model, high-dimensional data can be generated, to serve as input for the training of the data-assisted or data-driven ROMs. Using the autonomous system, i.e. equation (1.1) without any forcing,

$$\varepsilon \boldsymbol{f}^{ext}(\Omega t) = 0,$$

decaying trajectory data can be collected over m time-steps and arranged into an $n \times m$ matrix

$$\boldsymbol{Q} = \begin{pmatrix} q_1(t_1) & q_1(t_2) & \cdots & q_1(t_m) \\ q_2(t_1) & q_2(t_2) & \cdots & q_2(t_m) \\ \vdots & \vdots & \ddots & \vdots \\ q_n(t_1) & q_n(t_2) & \cdots & q_n(t_m) \end{pmatrix}.$$

In this matrix each column corresponds to a time-step and each row to a system variable. Similarly, the velocity and acceleration data can be arranged in the matrices $\dot{\boldsymbol{Q}}, \ddot{\boldsymbol{Q}} \in \mathbb{R}^{n \times m}$. This data can be utilized during the training of the data-assisted and fully data-driven MOR techniques, which are discussed in the next section.

2.2. Model Order Reduction Methods

This research employs six distinct MOR techniques, each of which is explored in detail in the following subsections. These include techniques using modal projection, autoencoders and polynomial regression, and more advanced methods, utilizing SINDy Autoencoders and SSMLearn. Finally the proposed method SPLIT and an extension SPLIT+ are introduced. Each methodology offers a unique approach to dimensionality reduction and model simplification.

2.2.1. Modal Projection

First a simple ROM based on modal projection and polynomial regression is introduced. The methodology involves projecting the high-dimensional data $\boldsymbol{Q}$ onto a lower-dimensional subspace spanned by the first d eigenmodes. Then, a polynomial model is fitted to this reduced data using ordinary least squares (OLS). With this reduced polynomial model, predictions can be made in the lower-dimensional subspace, which are then mapped back to the full space using the set of eigenmodes. This approach will be referred to as 'modal projection' throughout the remainder of this thesis.

2.2.1.1. Reducing the Dimensionality

Modal projection is a widely used technique in model reduction that approximates system behaviour by leveraging a subset of dominant eigenmodes. In this method, it is assumed that the dynamics of the system are primarily governed by its first d modes, capturing the most significant features of the system its behaviour. The first step is to compute these d eigenmodes $\boldsymbol{\varphi}_1, \boldsymbol{\varphi}_2, \ldots, \boldsymbol{\varphi}_d \in \mathbb{R}^n$ by solving the undamped eigenvalue problems

$$\boldsymbol{K}\boldsymbol{\varphi}_i = \omega_i^2 \boldsymbol{M}\boldsymbol{\varphi}_i, \quad i = 1, 2, \ldots, d.$$

Here $\boldsymbol{M}, \boldsymbol{K} \in \mathbb{R}^{n \times n}$ are the mass and stiffness matrices of the system, and ω_i is the undamped natural frequency corresponding to the mode $\boldsymbol{\varphi}_i$. Since the mass matrix $\boldsymbol{M}$ is positive definite, the eigenmodes are mass-normalized as

$$\boldsymbol{\varphi}_i^\top \boldsymbol{M}\boldsymbol{\varphi}_i = 1, \quad i = 1, 2, \ldots, d. \tag{2.1}$$

Once the first d eigenmodes are computed, the high-dimensional data $\boldsymbol{Q}$ can be projected onto the subspace spanned by these modes. The projection of the displacement data onto the reduced subspace is performed by the mapping

$$\boldsymbol{z}(t) = \boldsymbol{\varphi}_i^\top \boldsymbol{M} \cdot \boldsymbol{q}(t).$$

The data in the reduced coordinates, denoted by $\boldsymbol{Z} \in \mathbb{R}^{2d \times m}$, is obtained by projecting the displacement and velocity data $\boldsymbol{Q}$ and $\dot{\boldsymbol{Q}}$ onto the first d eigenmodes,

$$\boldsymbol{Z} = \begin{pmatrix} \boldsymbol{\varphi}_1^\top \boldsymbol{M} \cdot \boldsymbol{Q} \\ \vdots \\ \boldsymbol{\varphi}_d^\top \boldsymbol{M} \cdot \boldsymbol{Q} \\ \boldsymbol{\varphi}_1^\top \boldsymbol{M} \cdot \dot{\boldsymbol{Q}} \\ \vdots \\ \boldsymbol{\varphi}_d^\top \boldsymbol{M} \cdot \dot{\boldsymbol{Q}} \end{pmatrix} = \begin{pmatrix} z_1(t_1) & z_1(t_2) & \cdots & z_1(t_m) \\ \vdots & \vdots & \ddots & \vdots \\ z_d(t_1) & z_d(t_2) & \cdots & z_d(t_m) \\ \dot{z}_1(t_1) & \dot{z}_1(t_2) & \cdots & \dot{z}_1(t_m) \\ \vdots & \vdots & \ddots & \vdots \\ \dot{z}_d(t_1) & \dot{z}_d(t_2) & \cdots & \dot{z}_d(t_m) \end{pmatrix} = \begin{pmatrix} \boldsymbol{z}(t_1) & \boldsymbol{z}(t_2) & \cdots & \boldsymbol{z}(t_m) \\ \dot{\boldsymbol{z}}(t_1) & \dot{\boldsymbol{z}}(t_2) & \cdots & \dot{\boldsymbol{z}}(t_m) \end{pmatrix},$$

and similarly, its derivative $\dot{\boldsymbol{Z}} \in \mathbb{R}^{2d \times m}$ can be obtained by projecting the velocity and acceleration data $\dot{\boldsymbol{Q}}$ and $\ddot{\boldsymbol{Q}}$ onto the first d eigenmodes,

$$\dot{\boldsymbol{Z}} = \begin{pmatrix} \boldsymbol{\varphi}_1^\top \boldsymbol{M} \cdot \dot{\boldsymbol{Q}} \\ \vdots \\ \boldsymbol{\varphi}_d^\top \boldsymbol{M} \cdot \dot{\boldsymbol{Q}} \\ \boldsymbol{\varphi}_1^\top \boldsymbol{M} \cdot \ddot{\boldsymbol{Q}} \\ \vdots \\ \boldsymbol{\varphi}_d^\top \boldsymbol{M} \cdot \ddot{\boldsymbol{Q}} \end{pmatrix} = \begin{pmatrix} \dot{z}_1(t_1) & \dot{z}_1(t_2) & \cdots & \dot{z}_1(t_m) \\ \vdots & \vdots & \ddots & \vdots \\ \dot{z}_d(t_1) & \dot{z}_d(t_2) & \cdots & \dot{z}_d(t_m) \\ \ddot{z}_1(t_1) & \ddot{z}_1(t_2) & \cdots & \ddot{z}_1(t_m) \\ \vdots & \vdots & \ddots & \vdots \\ \ddot{z}_d(t_1) & \ddot{z}_d(t_2) & \cdots & \dot{z}_d(t_m) \end{pmatrix} = \begin{pmatrix} \dot{\boldsymbol{z}}(t_1) & \dot{\boldsymbol{z}}(t_2) & \cdots & \dot{\boldsymbol{z}}(t_m) \\ \ddot{\boldsymbol{z}}(t_1) & \ddot{\boldsymbol{z}}(t_2) & \cdots & \ddot{\boldsymbol{z}}(t_m) \end{pmatrix}.$$

Note that $\boldsymbol{Z}$ includes both displacement and velocity data, effectively doubling the dimensionality of the latent space to $2d$. This allows for fitting a first-order system of ODEs of size $2d$. The procedure for the discovery of the reduced dynamics is detailed below.

2.2.1.2. Discovering the Reduced Dynamics

After the data has been projected onto the reduced subspace, the next step is to fit a ROM to this reduced data, to approximate the system its dynamics in the reduced space. Polynomial regression is used for this purpose as it provides a simple yet flexible way to model dynamics. The general form of the model is a first-order system of ODEs,

$$\dot{\boldsymbol{Z}} = \boldsymbol{R} \cdot \boldsymbol{\Phi}(\boldsymbol{Z}). \tag{2.2}$$

Here $\boldsymbol{\Phi}(\boldsymbol{Z}) \in \mathbb{R}^{l \times m}$ is a library of monomials of $\boldsymbol{Z}$ up to degree p, expressed as

$$\boldsymbol{\Phi}(\boldsymbol{Z}) = \begin{pmatrix} \boldsymbol{Z} & \boldsymbol{Z}^2 & \boldsymbol{Z}^3 & \dots & \boldsymbol{Z}^p \end{pmatrix}^\top,$$

and $\boldsymbol{R} \in \mathbb{R}^{2d \times l}$ is a coefficient matrix that represents the parameters of the polynomial model. Note that l, which is the dimension of the library, depends on p and d.

The next step is to fit the coefficient matrix $\boldsymbol{R}$ by performing OLS regression. The goal of OLS is to find an $\boldsymbol{R}$ such that the difference between the observed time derivatives $\dot{\boldsymbol{Z}}$ and the predicted time derivatives $\boldsymbol{R} \cdot \boldsymbol{\Phi}(\boldsymbol{Z})$ is minimized. This is achieved by minimizing the residual sum of squares,

$$\left|\left| \dot{\boldsymbol{Z}} - \boldsymbol{R} \cdot \boldsymbol{\Phi}\left(\boldsymbol{Z}\right) \right|\right|_2^2 = \left(\dot{\boldsymbol{Z}} - \boldsymbol{R} \cdot \boldsymbol{\Phi}(\boldsymbol{Z}) \right)^\top \left(\dot{\boldsymbol{Z}} - \boldsymbol{R} \cdot \boldsymbol{\Phi}(\boldsymbol{Z}) \right). \tag{2.3}$$

This minimization problem has a unique solution, provided that the columns of the library $\boldsymbol{\Phi}(\boldsymbol{Z})$ are linearly independent [17]. It can be found by solving the so-called normal equations

$$\boldsymbol{\Phi}(\boldsymbol{Z})^\top \boldsymbol{\Phi}(\boldsymbol{Z}) \cdot \boldsymbol{R} = \boldsymbol{\Phi}(\boldsymbol{Z})^\top \dot{\boldsymbol{Z}}, \tag{2.4}$$

which can be derived when equation (2.3) is differentiated with respect to $\boldsymbol{R}$, and set to zero. Solving equation (2.4) gives the OLS solution for $\boldsymbol{R}$, denoted by

$$\boldsymbol{R} = \left(\boldsymbol{\Phi}(\boldsymbol{Z})^\top \boldsymbol{\Phi}(\boldsymbol{Z}) \right)^{-1} \boldsymbol{\Phi}(\boldsymbol{Z})^\top \dot{\boldsymbol{Z}}.$$

This matrix equation provides the optimal coefficient matrix that minimizes the error in approximating the reduced dynamics.

Once the coefficient matrix $\boldsymbol{R}$ has been determined, the ROM can be used to make predictions in the reduced space. The predicted dynamics are computed by solving the system of ordinary differential equations

$$\begin{pmatrix} \dot{\boldsymbol{z}}(t) \\ \ddot{\boldsymbol{z}}(t) \end{pmatrix} = \boldsymbol{R} \cdot \boldsymbol{\Phi} \left(\begin{pmatrix} \boldsymbol{z}(t) \\ \dot{\boldsymbol{z}}(t) \end{pmatrix} \right),$$

where $\boldsymbol{z}(t)$ represents the reduced coordinates corresponding to the first d eigenmodes. Numerical integration methods can be applied to solve this system over time. Once the trajectory $\boldsymbol{z}(t)$ has been predicted in the reduced space, the results must be mapped back to the full space for interpretation. To reconstruct the high-dimensional state $\boldsymbol{q}(t)$ from the reduced state $\boldsymbol{z}(t)$, the inverse of the projection step is used. The reduced state $\boldsymbol{z}(t)$ is mapped back to the full state $\boldsymbol{q}(t)$ using the first d eigenmodes as,

$$\boldsymbol{q}(t) = \sum_{i=1}^{d} \boldsymbol{\varphi}_i \cdot z_i(t).$$

2.2.2. A Decoupled Autoencoder

The next MOR technique is similar to the previous. However, instead of using modal projection to map to a linear subspace, an autoencoder is used to obtain a mapping to a non-linear manifold. The fitting of the reduced dynamics on this manifold remains the same, i.e. using polynomial regression. This autoencoder-based approach is expected to capture the non-linear behaviour of the system more accurately than the linear modal projection, particularly in scenarios involving large displacements and thus significant non-linearity's. This approach will be referred to as the 'decoupled autoencoder', distinguishing it from other methods involving autoencoders that employ a joint learning strategy, which are introduced later.

2.2.2.1. Reducing the Dimensionality

An autoencoder consists an encoder $\boldsymbol{w} : \mathbb{R}^n \to \mathbb{R}^d$ and a decoder $\boldsymbol{v} : \mathbb{R}^d \to \mathbb{R}^n$ (recall Figure 1.1). The encoder and decoder are composed of multiple layers, each with so-called weights and biases that are learned during the training process. Specifically, each layer performs a linear transformation on the input, followed by a non-linear activation function, which is applied element-wise. This combination allows the network to capture complex, non-linear relationships in the data.

In the encoding phase, the high-dimensional input data $\boldsymbol{Q}$ is passed through successive layers, gradually reducing its dimensionality until it reaches the lower-dimensional latent representation $\boldsymbol{Z}$. Mathematically, each layer applies a transformation of the form

$$\boldsymbol{y}_i = \sigma \left(\boldsymbol{W}_i \boldsymbol{y}_{i-1} + \boldsymbol{b}_i \right),$$

where $\boldsymbol{W}_i$ represents the weight matrix, $\boldsymbol{b}_i$ is the bias vector, σ the activation function, and $\boldsymbol{y}_i$ the output at layer i. The final output $\boldsymbol{Z}$ is the result of these transformations.

In the decoding phase, the process is reversed. The latent representation $\boldsymbol{Z}$ is passed through a series of decoding layers to reconstruct the original data, resulting

in $\hat{\boldsymbol{Q}}$. The decoder layers are similarly constructed, with weights, biases, and activation functions guiding the transformation back to the higher-dimensional space. The goal is to minimize the reconstruction error, usually measured using a loss function such as the mean squared error (MSE), which quantifies the difference between the reconstructed data $\hat{\boldsymbol{Q}}$ and the original data $\boldsymbol{Q}$. When all m data points of $\boldsymbol{Q}$ are used for training, the loss function is expressed as

$$\mathcal{L}\left(\boldsymbol{Q}, \hat{\boldsymbol{Q}}\right) = \frac{1}{m} \left|\left|\boldsymbol{Q} - \hat{\boldsymbol{Q}}\right|\right|_2^2.$$

Training an autoencoder requires updating the weights and biases in both the encoder and decoder to minimize the loss function. This is achieved through optimization algorithms, which process the training data in small batches. A complete pass over the entire dataset is called an epoch. As the training progresses, the autoencoder learns to extract the most significant features of the high-dimensional data, compressing it into a low-dimensional representation.

Once the training is done, the high-dimensional displacement, velocity and acceleration data can be mapped to the manifold using the learned encoder $\boldsymbol{w}$ and the chain rule,

$$z(t) = \boldsymbol{w}(\boldsymbol{q}(t)), \quad \dot{z}(t) = \nabla_{\boldsymbol{q}} \boldsymbol{w}(\boldsymbol{q}(t)) \dot{\boldsymbol{q}}(t), \quad \ddot{z}(t) = \nabla_{\boldsymbol{q}}^2 \boldsymbol{w}(\boldsymbol{q}(t))\, \dot{\boldsymbol{q}}^2(t) + \nabla_{\boldsymbol{q}} \boldsymbol{w}(\boldsymbol{q}(t))\, \ddot{\boldsymbol{q}}(t).$$

The data in the reduced coordinates, denoted by $\boldsymbol{Z}$, is obtained by projecting the displacement and velocity data $\boldsymbol{Q}$ and $\dot{\boldsymbol{Q}}$, as

$$\boldsymbol{Z} = \begin{pmatrix} \boldsymbol{w}(\boldsymbol{Q}) \\ \nabla_{\boldsymbol{q}} \boldsymbol{w}(\boldsymbol{Q}) \dot{\boldsymbol{Q}} \end{pmatrix} = \begin{pmatrix} \boldsymbol{z}(t_1) & \boldsymbol{z}(t_2) & \cdots & \boldsymbol{z}(t_m) \\ \dot{\boldsymbol{z}}(t_1) & \dot{\boldsymbol{z}}(t_2) & \cdots & \dot{\boldsymbol{z}}(t_m) \end{pmatrix},$$

and similarly, its derivative $\dot{\boldsymbol{Z}}$ can be obtained by projecting the velocity and acceleration data $\dot{\boldsymbol{Q}}$ and $\ddot{\boldsymbol{Q}}$, as

$$\dot{\boldsymbol{Z}} = \begin{pmatrix} \nabla_{\boldsymbol{q}} \boldsymbol{w}(\boldsymbol{Q}) \dot{\boldsymbol{Q}} \\ \nabla_{\boldsymbol{q}}^2 \boldsymbol{w}(\boldsymbol{Q})\, \dot{\boldsymbol{Q}}^2 + \nabla_{\boldsymbol{q}} \boldsymbol{w}(\boldsymbol{Q})\, \ddot{\boldsymbol{Q}} \end{pmatrix} = \begin{pmatrix} \dot{\boldsymbol{z}}(t_1) & \dot{\boldsymbol{z}}(t_2) & \cdots & \dot{\boldsymbol{z}}(t_m) \\ \ddot{\boldsymbol{z}}(t_1) & \ddot{\boldsymbol{z}}(t_2) & \cdots & \ddot{\boldsymbol{z}}(t_m) \end{pmatrix}.$$

2.2.2.2. Discovering the Reduced Dynamics

To the projected data, again a first-order system of ODEs (recall equation (2.2)) can be fitted using polynomial regression, by following the approach from subsection 2.2.1. This first-order system can again be used to make predictions in the reduced space, $\boldsymbol{z}(t)$, which can be mapped to the full space using the learned decoder

$$\boldsymbol{q}(t) = \boldsymbol{v}(\boldsymbol{z}(t)),$$

to obtain a prediction in the original high-dimensional space.

2.2.3. SINDy Autoencoders

The third MOR method uses an autoencoder, in combination with sparse identification of non-linear dynamics (SINDy). The approach simultaneously learns a low-dimensional representation of the system and identifies the underlying governing dynamics. Before introducing this method, called SINDy Autoencoders, the fundamentals SINDy are explained first.

2.2.3.1. SINDy

SINDy is a method for discovering governing equations of dynamical systems directly from data [5]. SINDy identifies the underlying sparse structure in the dynamical system, enabling the reconstruction of the system its behaviour with a minimal number of terms. The primary objective of SINDy is to identify a sparse representation of the dynamics of a first-order system of ODEs,

$$\dot{\boldsymbol{z}}(t) = \boldsymbol{r}(\boldsymbol{z}(t)),$$

where $\boldsymbol{z}(t) \in \mathbb{R}^d$ represents the state variables at time t and $\boldsymbol{r} : \mathbb{R}^d \to \mathbb{R}^d$ a non-linear function describing the system its dynamics.

Similar to polynomial regression, first m samples of $\boldsymbol{z}(t)$ are arranged in a matrix

$$\boldsymbol{Z} = \begin{pmatrix} z_1(t_1) & z_1(t_2) & \cdots & z_1(t_m) \\ z_2(t_1) & z_2(t_2) & \cdots & z_2(t_m) \\ \vdots & \vdots & \ddots & \vdots \\ z_d(t_1) & z_d(t_2) & \cdots & z_d(t_m) \end{pmatrix},$$

and similarly m samples of $\dot{\boldsymbol{z}}(t)$ are arranged in $\dot{\boldsymbol{Z}}$. Then a library $\boldsymbol{\Theta}(\boldsymbol{Z}) \in \mathbb{R}^{l \times m}$ of candidate functions is constructed, which may contain constant, monomial, or even trigonometric and rational terms, i.e.

$$\boldsymbol{\Theta}(\boldsymbol{Z}) = \begin{pmatrix} 1 & \boldsymbol{Z} & \boldsymbol{Z}^2 & \boldsymbol{Z}^3 & \cdots & \sin(\boldsymbol{Z}) & \cos(\boldsymbol{Z}) & \cdots \end{pmatrix}^\top.$$

This library is a sort of extension of the library of monomials $\boldsymbol{\Phi}(\boldsymbol{Z})$ used in polynomial regression. It is used to create the linear system

$$\dot{\boldsymbol{Z}} = \boldsymbol{\Xi} \cdot \boldsymbol{\Theta}(\boldsymbol{Z}), \tag{2.5}$$

where the matrix $\boldsymbol{\Xi} \in \mathbb{R}^{d \times l}$ is a set of coefficients that is learned using sparsity-promoting regression. Least squares with a sparsity-promoting L_1 regularization term is performed on the system from equation (2.5), resulting in

$$\boldsymbol{\Xi} = \underset{\boldsymbol{\Xi}'}{\arg\min} \left(\left|\left| \dot{\boldsymbol{Z}} - \boldsymbol{\Xi}' \boldsymbol{\Theta}(\boldsymbol{Z}) \right|\right|_2^2 + \lambda \left|\left| \boldsymbol{\Xi}' \right|\right|_1 \right),$$

for some regularization weight parameter λ.

2.2.3.2. SINDy Autoencoders

The SINDy Autoencoder combines this SINDy algorithm with an autoencoder to capture dynamics from scientific data. The autoencoder is enforced to learn coordinates associated with sparse dynamics by simultaneously learning a SINDy model for the dynamics of the reduced coordinates [10].

Similar to the decoupled autoencoder-based MOR method from subsection 2.2.2, the encoder and decoder of the autoencoder are learned by minimizing the reconstruction loss

$$\mathcal{L}_{recon}\left(\boldsymbol{Q}, \hat{\boldsymbol{Q}}\right) = \frac{1}{m} \left|\left| \boldsymbol{Q} - \hat{\boldsymbol{Q}} \right|\right|_2^2.$$

However, this loss function is augmented by additional terms that enforce the learned latent variables to follow sparse dynamics, modeled by SINDy. A library $\boldsymbol{\Theta}(\boldsymbol{Z}) \in \mathbb{R}^{l \times m}$ of candidate functions and a set of coefficients $\boldsymbol{\Xi} \in \mathbb{R}^{d \times l}$ are used to create the linear system

$$\dot{\boldsymbol{Z}} = \boldsymbol{\Xi} \cdot \boldsymbol{\Theta}(\boldsymbol{Z}),$$

similar to SINDy. The coefficients $\boldsymbol{\Xi}$ are again learned using sparsity-promoting regression. This is done in both the original and latent space. By using the chain rule, the losses in the reduced space

$$\begin{aligned}\mathcal{L}_{d\boldsymbol{z}/dt} &= \frac{1}{m}\left|\left|\dot{\boldsymbol{Z}} - \boldsymbol{\Xi} \cdot \boldsymbol{\Theta}(\boldsymbol{Z})\right|\right|_2^2 \\ &= \frac{1}{m}\left|\left|\nabla_{\boldsymbol{q}} \boldsymbol{w}(\boldsymbol{Q})\dot{\boldsymbol{Q}} - \boldsymbol{\Xi} \cdot \boldsymbol{\Theta}(\boldsymbol{w}(\boldsymbol{Q}))\right|\right|_2^2\end{aligned}$$

and the full space

$$\begin{aligned}\mathcal{L}_{d\boldsymbol{q}/dt} &= \frac{1}{m}\left|\left|\dot{\boldsymbol{Q}} - \nabla_z \boldsymbol{v}(\boldsymbol{Z})\,\dot{\boldsymbol{Z}}\right|\right|_2^2 \\ &= \frac{1}{m}\left|\left|\dot{\boldsymbol{Q}} - \left(\nabla_{\boldsymbol{z}} \boldsymbol{v}(\boldsymbol{w}(\boldsymbol{Q}))\right)\left(\boldsymbol{\Xi} \cdot \boldsymbol{\Theta}(\boldsymbol{Z})\right)\right|\right|_2^2 \\ &= \frac{1}{m}\left|\left|\dot{\boldsymbol{Q}} - \left(\nabla_{\boldsymbol{z}} \boldsymbol{v}(\boldsymbol{w}(\boldsymbol{Q}))\right)\left(\boldsymbol{\Xi} \cdot \boldsymbol{\Theta}(\boldsymbol{w}(\boldsymbol{Q}))\right)\right|\right|_2^2\end{aligned}$$

are constructed. The two loss functions, together with the L_1 regularization

$$\mathcal{L}_{reg} = ||\boldsymbol{\Xi}||_1,$$

to promote sparsity, are combined with the standard autoencoder loss $\mathcal{L}_{recon}$ to obtain the overall loss function

$$\mathcal{L}_{recon} + \lambda_1 \mathcal{L}_{d\boldsymbol{q}/dt} + \lambda_2 \mathcal{L}_{d\boldsymbol{z}/dt} + \lambda_3 \mathcal{L}_{reg}.$$

The hyperparameters λ_1, λ_2 and λ_3 determine the relative weight of the 4 terms in the loss function.

Once the training is done, the learned ROM can be used to make predictions in the reduced space, $\boldsymbol{z}(t)$, which can again be mapped to the full space using the learned decoder

$$\boldsymbol{q}(t) = \boldsymbol{v}(\boldsymbol{z}(t)),$$

to obtain a prediction in the original high-dimensional space.

2.2.3.3. Fitting a Second-Order System

Given that this research focuses on a second-order system, the SINDy autoencoder as described above requires a slight modification. The library $\boldsymbol{\Theta}$ must now include terms of both $\boldsymbol{Z}$ and $\dot{\boldsymbol{Z}}$, instead of just $\boldsymbol{Z}$, i.e.

$$\boldsymbol{\Theta}(\boldsymbol{Z}, \dot{\boldsymbol{Z}}) = \begin{pmatrix} 1 & \boldsymbol{Z} & \dot{\boldsymbol{Z}} & \boldsymbol{Z}^2 & \boldsymbol{Z}\dot{\boldsymbol{Z}} & \dot{\boldsymbol{Z}}^2 & \cdots & \sin(\boldsymbol{Z}) & \sin\left(\dot{\boldsymbol{Z}}\right) & \cos(\boldsymbol{Z}) & \cos\left(\dot{\boldsymbol{Z}}\right) & \cdots \end{pmatrix}^\top.$$

With this library, the second-order linear system of ODEs

$$\ddot{\boldsymbol{Z}} = \boldsymbol{\Xi} \cdot \boldsymbol{\Theta}\left(\boldsymbol{Z}, \dot{\boldsymbol{Z}}\right)$$

can be formulated, where $\boldsymbol{\Xi}$ is obtained by minimizing the loss in the reduced space

$$\begin{aligned}\mathcal{L}_{d^2\boldsymbol{z}/dt^2} &= \frac{1}{m}\left|\left|\ddot{\boldsymbol{Z}} - \boldsymbol{\Xi} \cdot \boldsymbol{\Theta}\left(\boldsymbol{Z}, \dot{\boldsymbol{Z}}\right)\right|\right|_2^2 \\ &= \frac{1}{m}\left|\left|\nabla_{\boldsymbol{q}}^2 \boldsymbol{w}(\boldsymbol{Q})\,\dot{\boldsymbol{Q}} + \nabla_{\boldsymbol{q}} \boldsymbol{w}(\boldsymbol{Q})\,\ddot{\boldsymbol{Q}} - \boldsymbol{\Xi} \cdot \boldsymbol{\Theta}\left(\boldsymbol{w}(\boldsymbol{Q}), \nabla_{\boldsymbol{q}} \boldsymbol{w}(\boldsymbol{Q})\,\dot{\boldsymbol{Q}}\right)\right|\right|_2^2\end{aligned}$$

and the loss in the full space

$$\begin{aligned}\mathcal{L}_{d^2\boldsymbol{q}/dt^2} &= \frac{1}{m}\left|\left|\ddot{\boldsymbol{Q}} - \left(\nabla_{\boldsymbol{z}}^2 \boldsymbol{v}(\boldsymbol{Z})\dot{\boldsymbol{Z}} + \nabla_{\boldsymbol{z}} \boldsymbol{v}(\boldsymbol{Z})\ddot{\boldsymbol{Z}}\right)\right|\right|_2^2 \\ &= \frac{1}{m}\left|\left|\ddot{\boldsymbol{Q}} - \left(\nabla_{\boldsymbol{z}}^2 \boldsymbol{v}(\boldsymbol{Z})\dot{\boldsymbol{Z}} + \nabla_{\boldsymbol{z}} \boldsymbol{v}(\boldsymbol{Z})\boldsymbol{\Xi} \cdot \boldsymbol{\Theta}\left(\boldsymbol{Z}, \dot{\boldsymbol{Z}}\right)\right)\right|\right|_2^2 \\ &= \frac{1}{m}\left|\left|\ddot{\boldsymbol{Q}} - \left(\nabla_{\boldsymbol{z}}^2 \boldsymbol{v}(\boldsymbol{w}(\boldsymbol{Q}))\nabla_{\boldsymbol{q}} \boldsymbol{w}(\boldsymbol{Q})\,\dot{\boldsymbol{Q}} + \nabla_{\boldsymbol{z}} \boldsymbol{v}(\boldsymbol{w}(\boldsymbol{Q}))\boldsymbol{\Xi} \cdot \boldsymbol{\Theta}\left(\boldsymbol{w}(\boldsymbol{Q}), \nabla_{\boldsymbol{q}} \boldsymbol{w}(\boldsymbol{Q})\,\dot{\boldsymbol{Q}}\right)\right)\right|\right|_2^2\end{aligned}$$

resulting in the new loss function

$$\mathcal{L}_{recon} + \lambda_1\,\mathcal{L}_{d^2\boldsymbol{q}/dt^2} + \lambda_2\,\mathcal{L}_{d^2\boldsymbol{z}/dt^2} + \lambda_3\,\mathcal{L}_{reg}.$$

Note that here the chain rule is used to obtain expressions for $\dot{\boldsymbol{Z}}$ and $\ddot{\boldsymbol{Z}}$ in terms of $\boldsymbol{Q}$.

It is important to highlight the differences between SINDy Autoencoder and the previous methods. In both the modal projection (subsection 2.2.1) and the decoupled autoencoder approach (subsection 2.2.2), the reduced space was derived first, and after that, the reduced dynamics within this space was identified by fitting a polynomial model using OLS. In contrast, the SINDy Autoencoder performs both the identification of the latent space and the reduced dynamics simultaneously within a single joint optimization process. As a result, the determination of the dynamics becomes part of the minimization problem rather than an exact computation using OLS. Additionally, in the modal projection and decoupled autoencoder methods, the reduced dynamics were represented by a first-order system of size $2d$, using the data matrix $\boldsymbol{Z}$, which included both the mapped displacement data $\boldsymbol{Q}$ and the mapped velocity data $\dot{\boldsymbol{Q}}$. In contrast, the SINDy Autoencoder uses an $\boldsymbol{Z}$ which contains only the displacement data $\boldsymbol{Q}$, and then fits a second-order system of size d to capture the dynamics. This approach reduces the computational load during training, as it avoids the need for an encoder of the form $w : \mathbb{R}^n \to \mathbb{R}^{2d}$ (instead of $w : \mathbb{R}^n \to \mathbb{R}^d$), which would increase the number of weights and biases that have to be learned.

2.2.4. SSMLearn

The fourth MOR technique is SSMLearn, which is based on spectral submanifold (SSM) theory. The SSM and its reduced dynamics are defined by a set of non-linear equations, which are determined through regression [21]. SSMLearn is captures the dominant modes and non-linearities in the dynamics, making it a powerful tool for

rigorous MOR of high-dimensional systems with significant computational efficiency [20].

To start, the second-order system

$$\boldsymbol{M}\ddot{\boldsymbol{q}} + \boldsymbol{C}\dot{\boldsymbol{q}} + \boldsymbol{K}\boldsymbol{q} + \boldsymbol{f}^{int}\left(\boldsymbol{q}, \dot{\boldsymbol{q}}\right) = \varepsilon \boldsymbol{f}^{ext}\left(\Omega t\right)$$

is transformed to the first-order system

$$\dot{\boldsymbol{x}} = \underbrace{\begin{pmatrix} \boldsymbol{0} & \boldsymbol{I} \\ -\boldsymbol{M}^{-1}\boldsymbol{K} & -\boldsymbol{M}^{-1}\boldsymbol{C} \end{pmatrix}}_{=\boldsymbol{A}} \boldsymbol{x} + \underbrace{\begin{pmatrix} \boldsymbol{0} \\ -\boldsymbol{M}^{-1}\boldsymbol{f}^{int}(\boldsymbol{q}, \dot{\boldsymbol{q}}) \end{pmatrix}}_{=\boldsymbol{f}_0(\boldsymbol{x})} + \varepsilon \underbrace{\begin{pmatrix} \boldsymbol{0} \\ \boldsymbol{M}^{-1}\boldsymbol{f}^{ext}(\Omega t) \end{pmatrix}}_{=\boldsymbol{f}_1(\Omega t)}. \tag{2.6}$$

SSMLearn uses an encoder $\boldsymbol{w} : [\mathbb{R}^{2n}, \mathbb{R}] \to \mathbb{R}^d$, which is a linear map,

$$\boldsymbol{z} = \boldsymbol{w}(\boldsymbol{x}, \Omega t) = \boldsymbol{W}_0 \boldsymbol{x}, \tag{2.7}$$

and a decoder $\boldsymbol{v} : \left[\mathbb{R}^d, \mathbb{R}\right] \to \mathbb{R}^{2n}$, which is a non-linear map

$$\boldsymbol{x} = \boldsymbol{v}(\boldsymbol{z}, \Omega t) = \boldsymbol{V}_0 \boldsymbol{z} + \boldsymbol{v}_{nl}(\boldsymbol{z}) + \varepsilon \boldsymbol{v}_1(\Omega t), \tag{2.8}$$

to map to and from a space with the reduced dynamics $\boldsymbol{r} : \mathbb{R}^d \to \mathbb{R}^d$,

$$\dot{\boldsymbol{z}} = \boldsymbol{r}(\boldsymbol{z}, \Omega t) = \boldsymbol{R}_0 \boldsymbol{z} + \boldsymbol{r}_{nl}(\boldsymbol{z}) + \varepsilon \boldsymbol{r}_1(\Omega t). \tag{2.9}$$

The linear parts of these mappings are defined using the first eigenmode $\boldsymbol{\varphi}_1$,

$$\boldsymbol{W}_0 = \begin{pmatrix} \boldsymbol{\varphi}_1^\top \boldsymbol{M} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{\varphi}_1^\top \boldsymbol{M} \end{pmatrix} \quad \text{and} \quad \boldsymbol{V}_0 = \begin{pmatrix} \boldsymbol{\varphi}_1 & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{\varphi}_1 \end{pmatrix},$$

meaning that $\boldsymbol{W}_0 \boldsymbol{V}_0 = \boldsymbol{I}$. The non-linear parts of these mappings are discovered by utilizing regression [8], i.e.

$$\boldsymbol{v}_{nl} = \underset{\boldsymbol{v}'_{nl}}{\arg\min} \sum_{j=1}^{m} ||\boldsymbol{x}_j - \boldsymbol{V}_0 \boldsymbol{z}_j - \boldsymbol{v}'_{nl}(\boldsymbol{z}_j)||_2^2, \quad \boldsymbol{r}_{nl} = \underset{\boldsymbol{r}'_{nl}}{\arg\min} \sum_{j=1}^{m} ||\dot{\boldsymbol{z}}_j - \boldsymbol{R}_0 \boldsymbol{z}_j - \boldsymbol{r}'_{nl}(\boldsymbol{z}_j)||_2^2.$$

To discover the time-dependent parts of the mappings, first some intermediate steps have to be taken.

When decoding and encoding $\boldsymbol{z}$, one finds

$$\begin{aligned} \boldsymbol{z} &= \boldsymbol{w}(\boldsymbol{v}(\boldsymbol{z}, \Omega t), \Omega t) \\ \boldsymbol{z} &= \boldsymbol{W}_0(\boldsymbol{V}_0 \boldsymbol{z} + \boldsymbol{v}_{nl}(\boldsymbol{z}) + \varepsilon \boldsymbol{v}_1(\Omega t)) \\ \boldsymbol{z} &= \underbrace{\boldsymbol{W}_0 \boldsymbol{V}_0}_{=\boldsymbol{I}} \boldsymbol{z} + \boldsymbol{W}_0 \boldsymbol{v}_{nl}(\boldsymbol{z}) + \varepsilon \boldsymbol{W}_0 \boldsymbol{v}_1(\Omega t)) \\ \boldsymbol{0} &= \underbrace{\boldsymbol{W}_0 \boldsymbol{v}_{nl}(\boldsymbol{z})}_{\boldsymbol{z}\text{-dependent}} + \varepsilon \underbrace{\boldsymbol{W}_0 \boldsymbol{v}_1(\Omega t))}_{\Omega t\text{-dependent}} \end{aligned}$$

meaning that

$$\boldsymbol{W}_0 \boldsymbol{v}_{nl}(\boldsymbol{z}) = \boldsymbol{W}_0 \boldsymbol{v}_1(\Omega t) = \boldsymbol{0}. \tag{2.10}$$

Substituting equation (2.8) and (2.9) into equation (2.6) gives the invariance equation

$$\nabla_{\boldsymbol{z}}\boldsymbol{v}(\boldsymbol{z},\Omega t)\boldsymbol{r}(\boldsymbol{z},\Omega t)+\nabla_{\Omega t}\boldsymbol{v}(\boldsymbol{z},\Omega t)\Omega=\boldsymbol{A}\boldsymbol{v}(\boldsymbol{z},\Omega t)+\boldsymbol{f}_0(\boldsymbol{v}(\boldsymbol{z},\Omega t))+\varepsilon\boldsymbol{f}_1(\Omega t).$$

Substituting equation (2.7) and (2.6) into equation (2.9) gives a second invariance equation

$$\nabla_{\boldsymbol{x}}\boldsymbol{w}(\boldsymbol{x},\Omega t)(\boldsymbol{A}\boldsymbol{x}+\boldsymbol{f}_0(\boldsymbol{x})+\varepsilon\boldsymbol{f}_1(\Omega t))+\nabla_{\Omega t}\boldsymbol{w}(\boldsymbol{x},\Omega t)\Omega=\boldsymbol{r}(\boldsymbol{w}(\boldsymbol{x},\Omega t)).$$

By using the fact that $\boldsymbol{w}(\boldsymbol{x},\Omega t)=\boldsymbol{W}_0\boldsymbol{x}$, one obtains

$$\boldsymbol{W}_0(\boldsymbol{A}\boldsymbol{x}+\boldsymbol{f}_0(\boldsymbol{x})+\varepsilon\boldsymbol{f}_1(\Omega t))+0\Omega=\boldsymbol{r}(\boldsymbol{W}_0\boldsymbol{x}),$$

meaning that the reduced dynamics is simply the projection of the full dynamics via $\boldsymbol{W}_0$. In particular

$$\boldsymbol{W}_0\boldsymbol{A}=\boldsymbol{R}_0\boldsymbol{W}_0, \tag{2.11}$$

or when left multiplied by $\boldsymbol{V}_0$

$$\boldsymbol{W}_0\boldsymbol{A}\boldsymbol{V}_0=\boldsymbol{R}_0.$$

With these results, the time-dependent parts of the mappings $\boldsymbol{v}$ and $\boldsymbol{w}$ can be found. First, equation (2.8) and (2.9) are substituted into the first invariance equation. Collecting the $\mathcal{O}(\varepsilon)$-terms gives

$$\boldsymbol{V}_0\boldsymbol{r}_1(\Omega t)+\nabla_{\Omega t}\boldsymbol{v}_1(\Omega t)\Omega=\boldsymbol{A}\boldsymbol{v}_1(\Omega t)+\boldsymbol{f}_1(\Omega t).$$

Multiplying this equation by $\boldsymbol{W}_0$ gives

$$\underbrace{\boldsymbol{W}_0\boldsymbol{V}_0}_{=\boldsymbol{I}}\boldsymbol{r}_1(\Omega t)+\nabla_{\Omega t}\underbrace{\boldsymbol{W}_0\boldsymbol{v}_1(\Omega t)}_{=\boldsymbol{0}\ (2.10)}\Omega=\underbrace{\boldsymbol{W}_0\boldsymbol{A}\boldsymbol{v}_1(\Omega t)}_{=\boldsymbol{R}_0\underbrace{\boldsymbol{W}_0\boldsymbol{v}(\Omega t)}_{=\boldsymbol{0}\ (2.10)}\ (2.11)}+\varepsilon\boldsymbol{W}_0\boldsymbol{f}_1(\Omega t),$$

so

$$\boldsymbol{r}_1(\Omega t)=\boldsymbol{W}_0\boldsymbol{f}_1(\Omega t), \tag{2.12}$$

meaning that the time-dependent part of the reduced dynamics, is simply the mapped external forcing function.

2.2.5. SPLIT

The proposed simultaneous projection and linear informed training (SPLIT) method combines elements of autoencoder-based manifold learning with dynamics modeling, inspired by the approaches SSMlearn [9] and SINDy Autoencoders [10]. The method jointly learns a low-dimensional manifold and the associated reduced dynamics by incorporating linear terms from the full order model (FOM) to guide the optimization. This ensures that the ROM captures the linear behaviour accurately, allowing the nonlinear components to focus on modeling deviations from this linear foundation.

SPLIT utilizes an autoencoder, with an encoder $\boldsymbol{w}$ which maps the full-order data $\boldsymbol{Q}$ to the low-dimensional coordinates $\boldsymbol{Z}$, while the decoder $\boldsymbol{v}$ reconstructs the full-

order data from the reduced coordinates. The encoder and decoder are trained by minimizing the reconstruction loss, expressed as

$$\mathcal{L}_{\text{recon}}\left(\boldsymbol{Q}, \hat{\boldsymbol{Q}}\right) = \frac{1}{m}\left|\left|\boldsymbol{Q} - \hat{\boldsymbol{Q}}\right|\right|_2^2,$$

where $\hat{\boldsymbol{Q}} = \boldsymbol{v}(\boldsymbol{w}(\boldsymbol{Q}))$. Simultaneously, the reduced dynamics, described by a system of second-order ODEs

$$\ddot{\boldsymbol{z}}(t) = \boldsymbol{R}_0 \begin{pmatrix} \boldsymbol{z}(t) \\ \dot{\boldsymbol{z}}(t) \end{pmatrix} + \boldsymbol{r}_{nl}(\boldsymbol{z}, \dot{\boldsymbol{z}})$$

is learned. Similar to SSMLearn, the linear part of these dynamics are given by the linear part of the FOM, mapped using the first eigenmode, resulting in

$$\boldsymbol{R}_0 = \begin{pmatrix} \boldsymbol{\varphi}^\top \boldsymbol{C} \boldsymbol{\varphi} & \boldsymbol{\varphi}^\top \boldsymbol{K} \boldsymbol{\varphi} \end{pmatrix}. \tag{2.13}$$

It might seem that these linear dynamics are independent of the mass matrix $\boldsymbol{M}$. However, it is important to note that $\boldsymbol{M}$ is already incorporated into the first eigenmode φ_1 through mass normalization (recall equation (2.1)). The non-linear part of the reduced dynamics $\boldsymbol{r}_{nl} : \left[\mathbb{R}^d, \mathbb{R}^d\right] \to \mathbb{R}^d$ are leaned in a style similar to the SINDy Autoencoders. Given a library $\boldsymbol{\Theta}(\boldsymbol{Z}) \in \mathbb{R}^{l \times m}$, which includes the non-linear monomials of $\boldsymbol{Z}$ and $\dot{\boldsymbol{Z}}$ up to order p, i.e.

$$\boldsymbol{\Theta}(\boldsymbol{Z}, \dot{\boldsymbol{Z}}) = \begin{pmatrix} \boldsymbol{Z}^2 & \boldsymbol{Z}\dot{\boldsymbol{Z}} & \dot{\boldsymbol{Z}}^2 & \boldsymbol{Z}^3 & \cdots & \dot{\boldsymbol{Z}}^p \end{pmatrix}^\top,$$

the second-order linear system of ODEs

$$\ddot{\boldsymbol{Z}} = \boldsymbol{R}_0 \begin{pmatrix} \boldsymbol{Z} \\ \dot{\boldsymbol{Z}} \end{pmatrix} + \boldsymbol{\Xi} \cdot \boldsymbol{\Theta}\left(\boldsymbol{Z}, \dot{\boldsymbol{Z}}\right)$$

can be formulated. The coefficient matrix $\boldsymbol{\Xi} \in \mathbb{R}^{d \times l}$ is obtained by minimizing the loss in the reduced space

$$\begin{aligned}
\mathcal{L}_{d^2\boldsymbol{z}/dt^2} &= \frac{1}{m}\left|\left|\ddot{\boldsymbol{Z}} - \left(\boldsymbol{R}_0 \begin{pmatrix} \boldsymbol{Z} \\ \dot{\boldsymbol{Z}} \end{pmatrix} + \boldsymbol{\Xi} \cdot \boldsymbol{\Theta}\left(\boldsymbol{Z}, \dot{\boldsymbol{Z}}\right)\right)\right|\right|_2^2 \\
&= \frac{1}{m}\left|\left|\nabla_{\boldsymbol{q}}^2 \boldsymbol{w}(\boldsymbol{Q})\, \dot{\boldsymbol{Q}} + \nabla_{\boldsymbol{q}} \boldsymbol{w}(\boldsymbol{Q})\, \ddot{\boldsymbol{Q}} - \boldsymbol{R}_0 \begin{pmatrix} \boldsymbol{w}(\boldsymbol{Q}) \\ \nabla_{\boldsymbol{q}} \boldsymbol{w}(\boldsymbol{Q})\, \dot{\boldsymbol{Q}} \end{pmatrix}\right.\right. \\
&\qquad \left.\left. - \boldsymbol{\Xi} \cdot \boldsymbol{\Theta}\left(\boldsymbol{w}(\boldsymbol{Q}), \nabla_{\boldsymbol{q}} \boldsymbol{w}(\boldsymbol{Q})\, \dot{\boldsymbol{Q}}\right)\right|\right|_2^2
\end{aligned}$$

and the loss in the full space

$$\begin{aligned}
\mathcal{L}_{d^2\boldsymbol{q}/dt^2} &= \frac{1}{m}\left|\left|\ddot{\boldsymbol{Q}} - \left(\nabla_{\boldsymbol{z}}^2 \boldsymbol{v}(\boldsymbol{Z})\dot{\boldsymbol{Z}} + \nabla_{\boldsymbol{z}} \boldsymbol{v}(\boldsymbol{Z})\ddot{\boldsymbol{Z}}\right)\right|\right|_2^2 \\
&= \frac{1}{m}\left|\left|\ddot{\boldsymbol{Q}} - \left(\nabla_{\boldsymbol{z}}^2 \boldsymbol{v}(\boldsymbol{Z})\dot{\boldsymbol{Z}} + \nabla_{\boldsymbol{z}} \boldsymbol{v}(\boldsymbol{Z}) \left(\boldsymbol{R}_0 \begin{pmatrix} \boldsymbol{Z} \\ \dot{\boldsymbol{Z}} \end{pmatrix} + \boldsymbol{\Xi} \cdot \boldsymbol{\Theta}\left(\boldsymbol{Z}, \dot{\boldsymbol{Z}}\right)\right)\right)\right|\right|_2^2 \\
&= \frac{1}{m}\left|\left|\ddot{\boldsymbol{Q}} - \left(\nabla_{\boldsymbol{z}}^2 \boldsymbol{v}(\boldsymbol{w}(\boldsymbol{Q}))\nabla_{\boldsymbol{q}} \boldsymbol{w}(\boldsymbol{Q})\, \dot{\boldsymbol{Q}} + \nabla_{\boldsymbol{z}} \boldsymbol{v}(\boldsymbol{w}(\boldsymbol{Q}))\left(\boldsymbol{R}_0 \begin{pmatrix} \boldsymbol{w}(\boldsymbol{Q}) \\ \nabla_{\boldsymbol{q}} \boldsymbol{w}(\boldsymbol{Q})\, \dot{\boldsymbol{Q}} \end{pmatrix}\right.\right.\right. \\
&\qquad \left.\left.\left. + \boldsymbol{\Xi} \cdot \boldsymbol{\Theta}\left(\boldsymbol{w}(\boldsymbol{Q}), \nabla_{\boldsymbol{q}} \boldsymbol{w}(\boldsymbol{Q})\, \dot{\boldsymbol{Q}}\right)\right)\right)\right|\right|_2^2.
\end{aligned}$$

Note that here the chain rule is used to obtain expressions for $\dot{\boldsymbol{Z}}$ and $\ddot{\boldsymbol{Z}}$ in terms of $\dot{\boldsymbol{Q}}$, $\boldsymbol{Q}$ and $\ddot{\boldsymbol{Q}}$. The two loss functions are combined with the standard autoencoder loss $\mathcal{L}_{recon}$ to obtain the overall loss function

$$\mathcal{L}_{recon} + \lambda_1 \mathcal{L}_{d^2\boldsymbol{q}/dt^2} + \lambda_2 \mathcal{L}_{d^2\boldsymbol{z}/dt^2}.$$

The hyperparameters λ_1 and λ_2 determine the relative weight of the 3 terms in the loss function. Note that for SPLIT, no regularization is used, as the sparsity that was utilized in the SINDy Autoencoder, is no longer of interest for SPLIT.

Once the training is done, the learned ROM can be used to make predictions in the reduced space, $\boldsymbol{z}(t)$, which can again be mapped to the full space using the learned decoder

$$\boldsymbol{q}(t) = \boldsymbol{v}(\boldsymbol{z}(t)),$$

to obtain a prediction in the original high-dimensional space.

2.2.6. SPLIT+

An extension of the SPLIT methodology, referred to as SPLIT+, introduces a new step after the joint training of the mapping and dynamics. In SPLIT+, once the low-dimensional manifold and corresponding dynamics have been learned through the joint optimization process, the learned dynamics are discarded. Instead, the trained autoencoder is only used to encode the data to the reduced space. In this reduced space, a first-order system is then fitted using polynomial regression. By discarding the learned coefficients, this approach allows for the possibility of capturing non-linearity's that the joint optimization might have missed, potentially due to incorrect training parameters. By still utilizing the joint learning of the mapping and the dynamics (even though these dynamics are never used), the autoencoder is ensured to project to a space where the dynamic system can operate effectively. Keeping this step guarantees that the learned manifold has a meaningful representation.

More specifically, after training, the encoder $\boldsymbol{w}$ is used to map the high-dimensional displacement and velocity data $\boldsymbol{Q}$ and $\dot{\boldsymbol{Q}}$ in the reduced coordinates, denoted by $\boldsymbol{Z}$, by utilizing

$$\boldsymbol{Z} = \begin{pmatrix} \boldsymbol{w}(\boldsymbol{Q}) \\ \nabla_{\boldsymbol{q}} \boldsymbol{w}(\boldsymbol{Q}) \dot{\boldsymbol{Q}} \end{pmatrix} = \begin{pmatrix} \boldsymbol{z}(t_1) & \boldsymbol{z}(t_2) & \cdots & \boldsymbol{z}(t_m) \\ \dot{\boldsymbol{z}}(t_1) & \dot{\boldsymbol{z}}(t_2) & \cdots & \dot{\boldsymbol{z}}(t_m) \end{pmatrix}.$$

Similarly, its derivative $\dot{\boldsymbol{Z}}$ can be obtained by projecting the velocity and acceleration data $\dot{\boldsymbol{Q}}$ and $\ddot{\boldsymbol{Q}}$, as

$$\dot{\boldsymbol{Z}} = \begin{pmatrix} \nabla_{\boldsymbol{q}} \boldsymbol{w}(\boldsymbol{Q}) \dot{\boldsymbol{Q}} \\ \nabla^2_{\boldsymbol{q}} \boldsymbol{w}(\boldsymbol{Q}) \, \dot{\boldsymbol{Q}}^2 + \nabla_{\boldsymbol{q}} \boldsymbol{w}(\boldsymbol{Q}) \, \ddot{\boldsymbol{Q}} \end{pmatrix} = \begin{pmatrix} \dot{\boldsymbol{z}}(t_1) & \dot{\boldsymbol{z}}(t_2) & \cdots & \dot{\boldsymbol{z}}(t_m) \\ \ddot{\boldsymbol{z}}(t_1) & \ddot{\boldsymbol{z}}(t_2) & \cdots & \ddot{\boldsymbol{z}}(t_m) \end{pmatrix}.$$

Instead of relying on the previously learned second-order dynamics from SPLIT, a first-order system of the form

$$\dot{\boldsymbol{Z}} = \boldsymbol{R}_0 \begin{pmatrix} \boldsymbol{Z} \\ \dot{\boldsymbol{Z}} \end{pmatrix} + \boldsymbol{R} \cdot \boldsymbol{\Phi}(\boldsymbol{Z})$$

is fitted to the encoded data. Here $\boldsymbol{R}_0 \in \mathbb{R}^2$ contains the linear parts of the FOM mapped using the first eigenmode (recall equation (2.13)), $\boldsymbol{\Phi}(\boldsymbol{Z}) \in \mathbb{R}^{l \times m}$ represents a library of non-linear monomials that describe the dynamics in the reduced space,

$$\boldsymbol{\Phi}(\boldsymbol{Z}) = \begin{pmatrix} \boldsymbol{Z}^2 & \boldsymbol{Z}^3 & \dots & \boldsymbol{Z}^p \end{pmatrix}^\top,$$

and $\boldsymbol{R} \in \mathbb{R}^{d \times l}$ contains the coefficients that need to be determined via OLS. This is again achieved by minimizing the residual sum of squares,

$$\left\| \left(\dot{\boldsymbol{Z}} - \boldsymbol{R}_0 \begin{pmatrix} \boldsymbol{Z} \\ \dot{\boldsymbol{Z}} \end{pmatrix} \right) - \boldsymbol{R} \cdot \boldsymbol{\Phi}\left(\boldsymbol{Z}\right) \right\|_2^2.$$

This minimization problem has a unique solution, provided that the columns of the library $\boldsymbol{\Phi}(\boldsymbol{Z})$ are linearly independent [17], given by

$$\boldsymbol{R} = \left(\boldsymbol{\Phi}(\boldsymbol{Z})^\top \boldsymbol{\Phi}(\boldsymbol{Z}) \right)^{-1} \left(\boldsymbol{\Phi}(\boldsymbol{Z})^\top \dot{\boldsymbol{Z}} - \Phi(\boldsymbol{Z})^\top \boldsymbol{R}_0 \begin{pmatrix} \boldsymbol{Z} \\ \dot{\boldsymbol{Z}} \end{pmatrix} \right).$$

This matrix equation provides the optimal coefficient matrix that minimizes the approximation error of the reduced dynamics, while simultaneously incorporating the linear components mapped from the FOM.

Once the first-order dynamics are fit, predictions in the reduced space can be made using this first-order polynomial system of ODEs. These predictions can be mapped back to the full space via the leaned decoder, as

$$\boldsymbol{q}(t) = \boldsymbol{v}(\boldsymbol{z}(t)).$$

2.3. Evaluation Metrics

To evaluate the performance of the ROMs, three evaluation metrics are used. One of the metrics is the normalized mean-trajectory-error, which gives the error between two trajectories in terms of a percentage. In addition to this error, the backbone curves are utilized, which give an indication of how the system behaves when no forcing is present. Finally, also forced response curves are used to evaluate the performance of the system when external forcing is introduced. The three metrics are explained in more detail below.

2.3.1. Normalized Mean-Trajectory-Error

The normalized mean-trajectory-error (NMTE) is employed as the primary metric. This metric effectively quantifies the accuracy of ROM-based predictions in capturing the dynamic behaviour of the full system [7, 9].

The NMTE of a displacement vector $\tilde{\boldsymbol{q}}$ is given by

$$\text{NMTE} = \frac{100}{m \cdot \max_i ||\boldsymbol{q}_i||_2} \sum_{i=1}^{m} ||\boldsymbol{q}_i - \tilde{\boldsymbol{q}}_i||_2,$$

where $\boldsymbol{q}$ is the displacement vector of the FOM. The NMTE metric is normalized by the maximum displacement magnitude observed in the dataset, ensuring that the error measure is scale-invariant and can be interpreted as a percentage of the largest

displacement. This normalization allows for a consistent comparison across different scenarios and scales of motion.

The NMTE can be used to calculate a reconstruction or reproduction error, by using the reconstructed displacement vector $\hat{\boldsymbol{q}} = \boldsymbol{v}(\boldsymbol{w}(\boldsymbol{q}))$ or the decoded displacement vector of the ROM $\boldsymbol{v}(\boldsymbol{z}(t))$ respectively. A lower NMTE indicates a better performance of the ROM in capturing the essential dynamics and characteristics of the dynamical system [8].

2.3.2. Backbone Curves

In addition to the NMTE, backbone curves are utilized as an evaluation metric to assess the performance of the ROMs. Backbone curves describe the relationship between the amplitude and frequency of oscillations in a non-linear system in the absence of external forcing. These curves are essential for understanding the non-linear characteristics and the resonance behaviour of the cantilever beam [6].

To extract the frequencies from the unforced time simulations, the method of Peak Finding and Fitting is employed [27]. As the name of the methods suggests, this method first finds the peaks in the time simulation. The time intervals between successive peaks are then calculated to determine the period, and thus the frequency of the oscillation. The extracted frequency data is plotted against the amplitude, to construct the backbone curve, which shows the dependence of frequency on the oscillation amplitude.

In linear systems, the natural frequency is constant regardless of the oscillation amplitude. In non-linear systems however, the natural frequency often varies with the amplitude of oscillation due to non-linear stiffness or other non-linear effects. This is illustrated in Figure 2.2, where an example of a backbone curve is given.

The backbone curves are used to evaluate the ROMs by comparing the ROM-

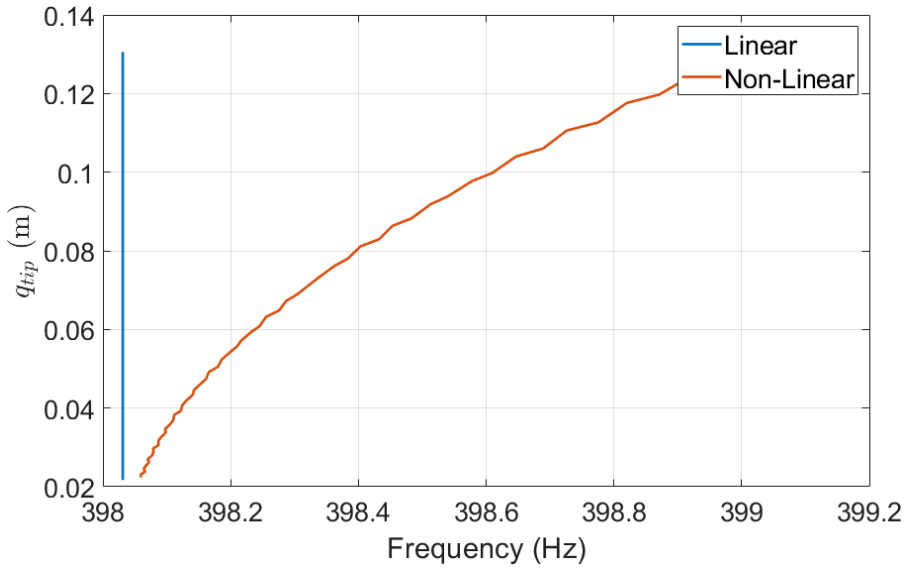


Figure 2.2: Illustration of two backbone curves. The linear backbone curve (blue) shows a constant frequency, whereas the non-linear (orange) backbone curve shows variation in frequency with oscillation amplitude.

generated curves against those obtained from the simulations of the FOM. This comparison helps in verifying the ability of the ROMs to capture the non-linear frequency-amplitude relationship inherent in the dynamic behaviour of the cantilever beam. A close match between the ROM-derived backbone curve and that from the FOM simulation indicates a high-fidelity ROM that can accurately reproduce the system its dynamics.

2.3.3. Forced Response Curves

Finally, forced response curves (FRCs) are employed to evaluate the predictive accuracy of ROMs. FRCs provide insights into how well the ROM predicts the behaviour of the system when external forcing is present [35]. FRCs are utilized to analyze both the amplitude and phase. Before detailing the process of generating the FRCs, it is essential to clarify how external forcing is integrated into the ROMs.

2.3.3.1. Treatment of External Forcing in ROM

For modal projection and SSMLearn, which use a linear encoder, the external forcing function can simply be mapped using $\boldsymbol{\varphi}_1^\top$ to obtain the time-dependent part of the reduced dynamics (recall e.g. equation (2.12)). However, for the autoencoder-based MOR techniques, which utilize a non-linear encoder, incorporating external forcing requires a different approach.

First, assume that the decoder and reduced dynamics now also include a time-dependent part, expressed as

$$\boldsymbol{q} = \boldsymbol{v}(\boldsymbol{z}) + \varepsilon \boldsymbol{v}_1(\Omega t) \quad \text{and} \quad \dot{\boldsymbol{z}} = \boldsymbol{r}(\boldsymbol{z}) + \varepsilon \boldsymbol{r}_1(\Omega t). \tag{2.14}$$

This can be justified using SSM theory, which states that the underlying manifold remains intact even with the inclusion of small-amplitude time-periodic forcing [20]. By substituting the time-dependent decoder and dynamics from equation (2.14) into the system defined in equation (1.1), it follows that

$$\begin{aligned}
\boldsymbol{M}\ddot{\boldsymbol{q}} + \boldsymbol{C}\dot{\boldsymbol{q}} + \boldsymbol{K}\boldsymbol{q} + \boldsymbol{f}^{int}(\boldsymbol{q}, \dot{\boldsymbol{q}}) &= \varepsilon \boldsymbol{f}^{ext}(\Omega t) \\
\ddot{\boldsymbol{q}} + \boldsymbol{M}^{-1}\boldsymbol{C}\dot{\boldsymbol{q}} + \boldsymbol{M}^{-1}\boldsymbol{K}\boldsymbol{q} + \boldsymbol{M}^{-1}\boldsymbol{f}^{int}(\boldsymbol{q}, \dot{\boldsymbol{q}}) &= \varepsilon \boldsymbol{M}^{-1}\boldsymbol{f}^{ext}(\Omega t) \\
\boldsymbol{v}''(\boldsymbol{z})\dot{\boldsymbol{z}}^2 + \boldsymbol{v}'(\boldsymbol{z})\ddot{\boldsymbol{z}} + \varepsilon\Omega^2 \boldsymbol{v}_1''(\Omega t) + \boldsymbol{M}^{-1}\boldsymbol{C}(\boldsymbol{v}'(\boldsymbol{z})\dot{\boldsymbol{z}} + \varepsilon\Omega \boldsymbol{v}_1'(\Omega t)) & \\
+\boldsymbol{M}^{-1}\boldsymbol{K}(\boldsymbol{v}(\boldsymbol{z}) + \varepsilon \boldsymbol{v}_1(\Omega t)) + \boldsymbol{M}^{-1}\boldsymbol{f}^{int}(\boldsymbol{q}, \dot{\boldsymbol{q}}) &= \varepsilon \boldsymbol{M}^{-1}\boldsymbol{f}^{ext}(\Omega t) \\
\boldsymbol{v}''(\boldsymbol{z})\dot{\boldsymbol{z}}^2 + \boldsymbol{v}'(\boldsymbol{z})(\boldsymbol{r}(\boldsymbol{z}) + \varepsilon \boldsymbol{r}_1(\Omega t)) + \varepsilon\Omega^2 \boldsymbol{v}_1''(\Omega t) + \boldsymbol{M}^{-1}\boldsymbol{C}(\boldsymbol{v}'(\boldsymbol{z})\dot{\boldsymbol{z}} & \\
+\varepsilon\Omega \boldsymbol{v}_1'(\Omega t)) + \boldsymbol{M}^{-1}\boldsymbol{K}(\boldsymbol{v}(\boldsymbol{z}) + \varepsilon \boldsymbol{v}_1(\Omega t)) + \boldsymbol{M}^{-1}\boldsymbol{f}^{int}(\boldsymbol{q}, \dot{\boldsymbol{q}}) &= \varepsilon \boldsymbol{M}^{-1}\boldsymbol{f}^{ext}(\Omega t)
\end{aligned}$$

Collecting the $\mathcal{O}(\varepsilon)$-terms leaves

$$\boldsymbol{v}'(\boldsymbol{z})\boldsymbol{r}_1(\Omega t) + \Omega^2 \boldsymbol{v}_1''(\Omega t) + \Omega \boldsymbol{v}_1'(\Omega t) + \boldsymbol{v}_1(\Omega t) = \boldsymbol{M}^{-1}\boldsymbol{f}^{ext}(\Omega t),$$

and by choosing $\boldsymbol{v}_1 = \boldsymbol{0}$, one finds

$$\boldsymbol{v}'(\boldsymbol{z})\boldsymbol{r}_1(\Omega t) = \boldsymbol{M}^{-1}\boldsymbol{f}^{ext}(\Omega t). \tag{2.15}$$

Now let $\boldsymbol{\alpha}$ be such that $\boldsymbol{\alpha}(\boldsymbol{z}) \cdot \boldsymbol{v}'(\boldsymbol{z}) = 1$, i.e.

$$\boldsymbol{\alpha}(\boldsymbol{z}) = \frac{\boldsymbol{v}'(\boldsymbol{z})^\top}{||\boldsymbol{v}'(\boldsymbol{z})||_2^2}.$$

Then equation (2.15) can be multiplied by $\boldsymbol{\alpha}$ to obtain

$$\boldsymbol{r}_1(\Omega t) = \boldsymbol{\alpha}(\boldsymbol{z}) \cdot \boldsymbol{M}^{-1} \boldsymbol{f}^{ext}(\Omega t),$$

which means that the reduced dynamics with external forcing can be described by

$$\ddot{\boldsymbol{z}} = \boldsymbol{r}(\boldsymbol{z}) + \boldsymbol{\alpha}(\boldsymbol{z}) \cdot \boldsymbol{M}^{-1} \boldsymbol{f}^{ext}(\Omega t). \tag{2.16}$$

Note that the external forcing now not only depends on the time t, but also the reduced state $\boldsymbol{z}$.

2.3.3.2. Construction of Forced Response Curves

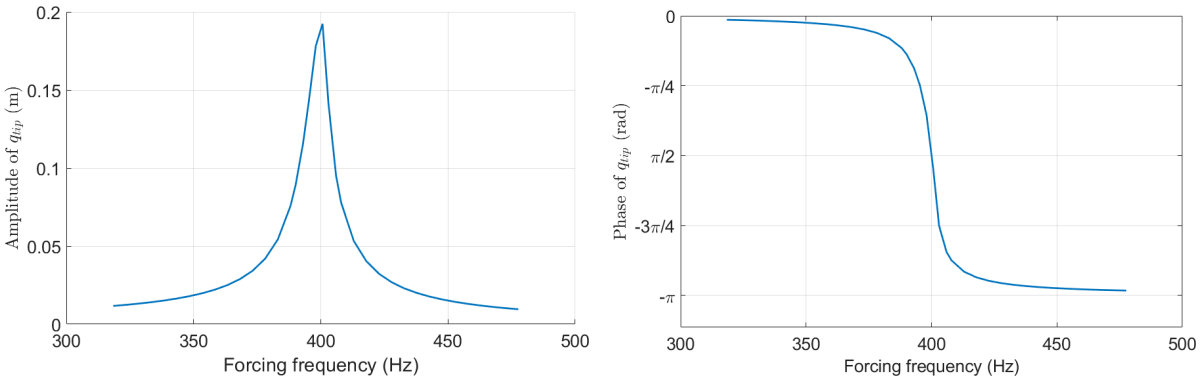
To construct the FRCs, the system is subjected to a periodic external force with varying frequencies Ω. The forced simulation is run until the fast transients die out, ensuring a steady-state forced response is achieved. The displacement of the tip of the beam q_{tip} during the last period, denoted by $[\tau, \tau + T)$, is then used to extract the amplitude and phase of the response. The underlying idea is that with the amplitude and phase, the response at a particular degree of freedom can be described as

$$\text{amplitude} \cdot \sin(\Omega t + \text{phase}).$$

The amplitude of the response is determined by simply finding the maximum value of q_{tip}. The phase of the response is computed by taking the arctangent of the integral of the response multiplied by the complex exponential $e^{-i2\pi t/T}$ [8], i.e.

$$\text{amplitude} = \max_{t \in [\tau, \tau+T)} |q_{tip}(t)|, \qquad \text{phase} = \angle \int_{\tau}^{\tau+T} q_{tip}(t) \cdot e^{-i2\pi t/T} \, dt.$$

Figure 2.3 provides an example of the FRCs for both the amplitude and phase under external forcing at different frequencies. The amplitude of the tip of the beam is expected to peak near its eigenfrequency. The phase shifts from 0 to $-\pi$, with a sharp decline, also occurring around the eigenfrequency.



(a) Amplitude **(b)** Phase

Figure 2.3: Example of forced response curves showing the amplitude and phase of the tip of the beam under external forcing at different frequencies. The amplitude peaks around the beam its eigenfrequency, while the phase drops sharply from 0 to $-\pi$ near that same frequency.

A ROM can be considered effective if its FRC closely aligns with that of the FOM. Achieving this indicates that the ROM captures the system its resonant behaviour and dynamic response under external periodic loading conditions. In addition, since ROMs are trained on unforced data, their ability to predict forced responses effectively would showcase a powerful predictive capability.

This approach is a straightforward way to extract FRCs, that lies within the scope of this thesis. However, it is worth noting that more sophisticated methods, such as numerical continuation [23], or analytical techniques which utilize the normal form of the reduced dynamics [9], exist.

3 Results

This chapter presents the findings of the study. It starts by outlining the parameter configuration of the experimental setup. Following this, an evaluation of the training outcomes for the model order reduction (MOR) techniques are discussed. The chapter then discusses the performance of the reduced order models (ROMs) during both unforced and forced simulations. Finally, the computational run times are analyzed.

3.1. Parameter Configuration

This section discusses the parameter configuration used for the full order model (FOM) and the MOR techniques. First, the setup of the finite element (FE) model and the simulation parameters for obtaining the full order data are described. Subsequently, the parameters for the MOR techniques are provided.

3.1.1. Parameters for Data Collection

Since this thesis focuses on data-driven and data-assisted ROMs, first data has to be collected. The FE model of the 2D cantilever beam is utilized to do so. The beam that is used, is constructed from titanium, and has a length of 1 meter, a height (along the bending direction) of 0.02 meter, and an out of plane width of 0.05 meter [52]. The geometric and material properties of the beam are summarized in Table 3.1.

Table 3.1: Material and geometric properties of the titanium cantilever beam.

Property	**Value**
Young's modulus	$104 \cdot 10^9$ Pa
Density	440 kg/m^3
Poisson's ratio	0.3
Length	1 m
Height (along bending direction)	0.02 m
Width (out of plane)	0.05 m

The beam is discretized using a finite element grid with 40 elements along the length and 2 elements along the height (recall Figure 2.1). This results in 246 degrees of freedom, of which 6 are fixed by the boundary condition. The mass matrix $\boldsymbol{M}$ and stiffness matrix $\boldsymbol{K}$ are obtained through the standard assembly routine in the YetAnotherFEcode package. For damping, a mass-proportional damping approach is used [52]. The damping matrix $\boldsymbol{C}$ is defined as

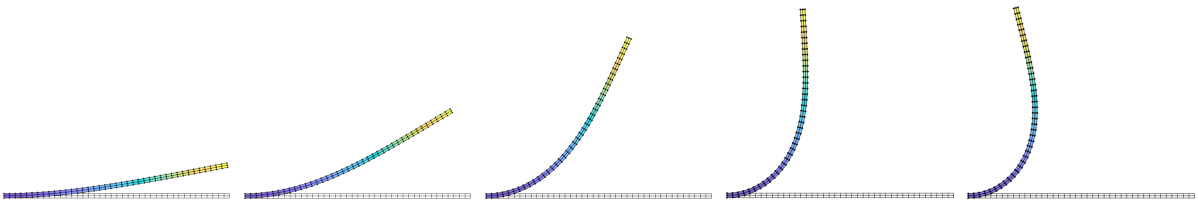
$$\boldsymbol{C} = \frac{\omega_1}{50}\boldsymbol{M},$$

where ω_1 is the first undamped natural frequency of the beam.

To generate trajectories for training and testing, unforced simulations are conducted using the autonomous system. The initial displacement is derived from a prior performed forced simulation, while the initial velocity and acceleration are set to zero. Five different initial displacement vectors are used, with increasing magnitudes. They are visualized in Figure 3.1. Note that in the two of the cases, the beam folds over itself. The five initial conditions result in five different trajectories, with increasing non-linearity. Using the trajectories, three different test cases are defined:

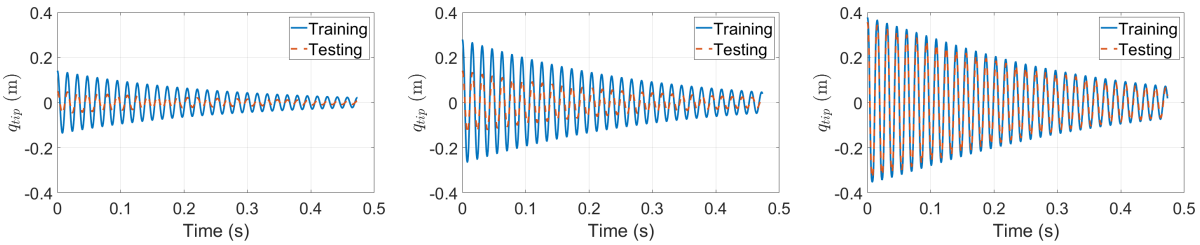
1. The first case uses **small displacements**. The ROM is trained with a trajectory where the tip of the beam has an initial displacement of 0.1412 m (Figure 3.1b), and tested with a trajectory where the tip has an initial displacement of 0.0496 m (Figure 3.1a). The corresponding trajectories, shown in Figure 3.2a, do not contain too much non-linear behaviour yet.
2. The second case uses **large displacements**. The ROM is trained with a trajectory where the tip of the beam has an initial displacement of 0.2797 m (Figure 3.1c), and tested with a trajectory where the tip has an initial displacement of 0.1412 m (Figure 3.1b). The corresponding trajectories are shown in Figure 3.2b. More non-linear behaviour is expected in these trajectories.
3. Finally also a scenario where the beam reaches a **folding configuration** is utilized. The ROM is trained with a trajectory where the tip of the beam has an initial displacement of 0.3777 m (Figure 3.1e), and tested with a trajectory where the tip has an initial displacement of 0.3568 m (Figure 3.1d). The trajectories in this case, shown in Figure 3.2c, exhibit very non-linear behaviour.

Note that each time, the ROM is tested with slightly smaller displacements than that it is trained on.



(a) 0.0496 m **(b)** 0.1412 m **(c)** 0.2797 m **(d)** 0.3568 m **(e)** 0.3777 m

Figure 3.1: Five different displacements, used as initial conditions, arranged in increasing order of magnitude. The displacements, specified in meters, correspond to the tip of the beam. The configurations of 3.1d and 3.1e display folding behaviour, with the beam displacing enough to fold over itself.



(a) Small displacements **(b)** Large displacements **(c)** Folding configuration

Figure 3.2: Trajectories of the displacement of the tip of the beam (perpendicular to the beam). These trajectories are used for training and testing the ROMs. No external forcing is present. During the 30 periods T, the displacement decays, converging to an equilibrium.

The Newmark-beta integration scheme is employed for time-integration, with $\beta = 0.275625$ and $\gamma = 0.55$ [13]. The time-step size of this implicit Newmark is set to be $h = T/1000$, where T is the period corresponding to the eigenfrequency of the beam. The simulation is run for 30 periods T to capture the dynamic response. This results in trajectories with $m = 29996$ time-steps. The displacement data $\boldsymbol{Q}$ and velocity data $\dot{\boldsymbol{Q}}$ vectors are directly obtained from these simulations. To calculate the acceleration data $\ddot{\boldsymbol{Q}}$, finite differencing with sixth-order accuracy is used, ensuring precise derivative estimates.

3.1.2. Parameters for Model Order Reduction

Besides the parameters of the FOM, also the parameters of the MOR techniques have to be specified. While parts of the six MOR methodologies from section 2.2 overlap, each method has some distinct configurations and parameters. The settings are as follows:

- The modal projection method (subsection 2.2.1) utilizes only the first eigenmode, $\boldsymbol{\varphi}_1$, for projecting the system to the reduced space, as the behaviour of the beam is primarily governed by this mode.
- The SSMLearn methodology (subsection 2.2.4) identifies a two-dimensional invariant manifold using a non-linear polynomial basis of order 3 for the decoder. The dynamics on this manifold, also of order 3, are then fitted using the linear part of the reduced dynamics.
- The autoencoder-based methods (subsection 2.2.2, 2.2.3, 2.2.5, and 2.2.6) all utilize the same autoencoder architecture. It consists of an input and output layer matching the number of unconstrained degrees of freedom, which is 240. Between these layers, there are five hidden layers with sizes 64, 32, 1, 32, and 64, respectively. This architecture ensures a compact and non-linear mapping of the full-order data to a low-dimensional space. The encoder $\boldsymbol{w} : \mathbb{R}^{240} \to \mathbb{R}$ maps the data to a single latent dimension, while the decoder $\boldsymbol{v} : \mathbb{R} \to \mathbb{R}^{240}$ reconstructs it back to the full space.

  To guarantee that the equilibrium state of the beam maps to zero in the latent space, the autoencoder does not include biases, relying solely on weights and activation functions. The chosen activation function is the exponential linear unit

(ELU), which also maps zero to zero. The weights of the network are initialized using Xavier initialization [16], and training is performed for 15,000 epochs using the Adam optimizer [31] with a batch size of 1024 and a learning rate of 10^{-4} [10].

- In addition to this autoencoder setup, the SINDy Autoencoder, SPLIT and SPLIT+ (subsection 2.2.3, 2.2.5, and 2.2.6) utilize hyperparameters in their loss functions. The loss weight in the full space, λ_1, is calculated as the ratio of the squared norms of the displacement data $\boldsymbol{Q}$ and the acceleration data $\ddot{\boldsymbol{Q}}$ [10], expressed as

$$\lambda_1 = \frac{||\boldsymbol{Q}||_2^2}{||\ddot{\boldsymbol{Q}}||_2^2}.$$

  The loss weight in the reduced space, λ_2, is set to be one order of magnitude lower than λ_1 [10]. This configuration ensures a slight prioritization of the reconstruction of $\boldsymbol{q}$ over the prediction of $\ddot{\boldsymbol{q}}$, ensuring that the encoder and decoder focus primarily on reconstructing the original data, while the reduced model emphasizes accurate prediction of the second-order dynamics.

  For the SINDy Autoencoder (subsection 2.2.3), the third hyperparameter, the sparsity coefficient λ_3, is set to 0 as sparsity is not of interest in this study. In addition, the library $\boldsymbol{\Theta}$ includes only monomials, so excluding any constants or trigonometric functions.

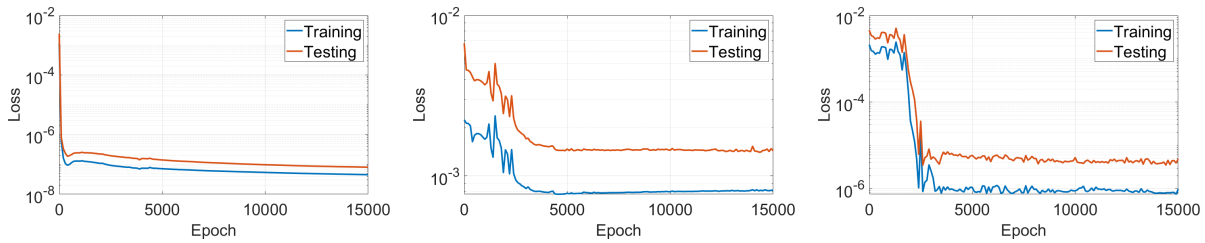
Note that these configurations ensure that all methods utilize a two-dimensional subspace for the reduced dynamics ($d = 2$). This is intentional, as the behaviour of the 2D cantilever beam is primarily governed by the first eigenmode, which has an attracting invariant manifold with a dimensionality of 2 [8]. Additionally, the dynamics within the reduced space are represented using a polynomial library of order 3 for all MOR techniques ($p = 3$). This is done since cubic non-linearity's are expected in the beam dynamics [39]. The time-integration of all reduced order models is carried out using the 8th-order Runge-Kutta method [51].

3.2. Evaluation of Training Outcome

The following section provides an evaluation of the training outcomes for the MOR techniques, without yet using them for simulations or predictions. Instead, the analysis focuses on assessing the results directly obtained from the training process itself. This includes tracking the losses of the autoencoder-based methods, analyzing the eigenvalues of the discovered reduced dynamics, and evaluating the reconstruction accuracy of the MOR techniques.

3.2.1. Losses of Autoencoder-based Techniques

During the training process of the autoencoder-based MOR techniques (subsection 2.2.2, 2.2.3, 2.2.5, and 2.2.6), the loss functions can be tracked and plotted against the number of epochs. This allows for a visualization of how well the model is learning over time. This is done for the case with the large, non-folding, displacements (where the initial displacement of the beam is 0.2797 m), and illustrated in



(a) Decoupled autoencoder **(b)** SINDy Autoencoder **(c)** SPLIT

Figure 3.3: The training and testing losses plotted against the number of epochs for the different autoencoders, in the case of the data with the large, non-folding displacements. The training loss (blue) and testing loss on slightly smaller displacements (orange) both decrease over time, indicating improved model performance.

Figure 3.3. The training loss, represented in blue, reflects the model its performance on the training data, while the testing loss, illustrated in orange, shows the model its performance on the test data, which features the slightly smaller initial displacement (0.1412 m). The network has not encountered this test data during training. Both losses decrease over time as the autoencoder learns to better reconstruct the input data. In addition, for the joint methods from subsection 2.2.3, 2.2.5, and 2.2.6, the network also achieves a more accurate representation of the reduced dynamics.

Among the MOR techniques, the decoupled autoencoder, shown in Figure 3.3a achieves the smallest loss, demonstrating the most effective learning. It is important to keep in mind however that the loss function of this network is also the simplest. The proposed method SPLIT (and its extension SPLIT+, as they use the same autoencoder) shown in Figure 3.3c, follows with a loss that is two orders of magnitude higher. Meanwhile, the SINDy Autoencoder (Figure 3.3b) shows the highest loss among the three, suggesting it struggles more with obtaining an accurate reconstruction and dynamics compared to the other approaches. This indicates that the incorporation the linear terms, as done in SPLIT and SPLIT+, really enhances the optimization process.

3.2.2. Eigenvalues of Linear Dynamics

After training, the eigenvalues of the linear part of the discovered dynamics can be calculated. A comparison of these eigenvalues with those of the FOM provides insight into how well the ROMs capture the fundamental dynamic characteristics of the system. The eigenvalues of the first eigenmode of the FOM are $-3.9805 \pm 398.03i$. These can be calculated using the damping ratio ζ_1 [23], as

$$\left(-\zeta_1 \pm \sqrt{\zeta_1^2 - 1}\right)\omega_1, \quad \text{where } \zeta_1 = \frac{1}{2\omega_1}\left(\frac{\boldsymbol{\varphi}_1^\top \boldsymbol{C}\boldsymbol{\varphi}_1}{\boldsymbol{\varphi}_1^\top \boldsymbol{M}\boldsymbol{\varphi}_1}\right).$$

Here ω_1 is the undamped natural frequency associated with the first eigenmode $\boldsymbol{\varphi}_1$, and $\boldsymbol{C}$ and $\boldsymbol{M}$ are the damping and mass matrix of the FOM respectively. For the ROMs, the eigenvalues can be obtained by using the linear part of the reduced dynamics that where discovered during training. Specifically, the eigenvalues can be determined by examining the first two rows and columns of the coefficient matrix $\boldsymbol{R}$ (or $\boldsymbol{\Xi}$ for the SINDy Autoencoder and SPLIT).

Table 3.2: Eigenvalues of the linear part of the discovered dynamics across the six different MOR techniques and three different cases with increasing non-linearity: small displacements (0.1412 m), large displacements (0.2797 m), and displacements where the beam is in a folding configuration (0.3777 m).

	Small displacements	**Large displacements**	**Folding configuration**
Modal Projection	-3.9946 ± 397.99i	-4.1405 ± 397.53i	-4.8033 ± 396.98i
Decoupled AE	-4.1809 ± 397.71i	-4.3870 ± 396.87i	-5.1730 ± 394.13i
SINDy AE	0.19689 ± 4.6250i	-4.2943 ± 3.0028i	-3.2833 ± 4.0206i
SSMLearn	-3.9805 ± 398.03i	-3.9805 ± 398.03i	-3.9805 ± 398.03i
SPLIT	-3.9805 ± 398.03i	-3.9805 ± 398.03i	-3.9805 ± 398.03i
SPLIT+	-3.9805 ± 398.03i	-3.9805 ± 398.03i	-3.9805 ± 398.03i

The eigenvalues of the linear parts of the six ROMs are shown in Table 3.2, for the three different cases. For SSMLearn, SPLIT, and SPLIT+, the linear dynamics, and consequently their corresponding eigenvalues, are specified during training (recall subsection 2.2.4, 2.2.5, and 2.2.6). As a result, the eigenvalues of these ROMs are identical to those of the FOM in all three cases.

The eigenvalues of the modal projection and decoupled autoencoder methods show slight deviations from those of the FOM. This discrepancy increases as the non-linearity of the system increases (from small displacements to folding configurations). This indicates that, while these methods can approximate the linear dynamics well in simpler scenarios, their accuracy diminishes as the complexity of the system increases. The SINDy Autoencoder exhibits significant deviations from those of the FOM across all three cases, even producing a positive damping ratio (real part) in the case of small displacements. This indicates that it struggles to capture the linear dynamics accurately, which will lead to errors when this ROM will be used for predictions.

3.2.3. Reconstruction Accuracy

A third and final way to measure the performance of the MOR techniques, without actually conducting simulations, is by verifying that the mapping to and from the reduced space can be performed without introducing errors. This can be done using the so-called reconstruction error. It is important to note that this evaluation does not focus on the performance of the discovered dynamics. Instead, the normalized mean-trajectory-error (NMTE) is used to measure the difference between the displacement of the FOM, $\boldsymbol{q}$, and the displacement after being mapped to the reduced space and then reconstructed back to the full space, $\hat{\boldsymbol{q}}$.

For example, the reconstruction error for modal projection is given by

$$\text{NMTE} = \frac{100}{m \cdot \max_i ||\boldsymbol{q}_i||_2} \sum_{i=1}^{m} ||\boldsymbol{q}_i - \boldsymbol{\varphi}_1^\top \boldsymbol{M} \boldsymbol{\varphi}_1 \boldsymbol{q}_i||_2 ,$$

and for an autoencoder with encoder $\boldsymbol{w}$ and decoder $\boldsymbol{v}$, the NMTE is given by

$$\text{NMTE} = \frac{100}{m \cdot \max_i ||\boldsymbol{q}_i||_2} \sum_{i=1}^{m} ||\boldsymbol{q}_i - \boldsymbol{v}(\boldsymbol{w}(\boldsymbol{q}_i))||_2 .$$

Table 3.3: Normalized mean-trajectory-errors for the reconstruction of the six different MOR techniques across the three cases with increasing non-linearity: small displacements (0.0496 m), large displacements (0.1412 m), and displacements so far the beam is in a folding configuration (0.3568 m).

	Small displacements	**Large displacements**	**Folding configuration**
Modal Projection	0.93698%	2.7082%	8.2094%
Decoupled AE	0.085436%	0.16938%	0.24545%
SINDy AE	2.3038%	2.6283%	2.9808%
SSMLearn	0.021106%	0.25558%	1.1236%
SPLIT	0.086916%	0.43862%	0.94212%
SPLIT+	0.086916%	0.43862%	0.94212%

As discussed in subsection 3.1.1, the testing is done with a trajectory containing slightly smaller displacement than the one used for training.

The reconstruction errors for the six different MOR techniques, for the three different cases are shown in Table 3.3. Note that for small displacements, and thus little non-linearity, all methods seem to reconstruct the data fine, even the linear modal projection. However, when the displacements start to increase, eventually to a folding configuration, the reconstruction errors increase, especially those of the modal projection. This is in line with the expectation: A linear dimensionality reduction technique should not be able to capture complex non-linear behaviour. Also note that the reconstruction errors of SPLIT and SPLIT+ are the same, as the same autoencoder is utilized in both MOR techniques (and only the reduced dynamics are different).

3.3. Unforced Simulations

After analyzing the outcomes directly obtained from the training process, the next step is to evaluate the performance of the ROMs in making predictions. This section evaluates the performance of the ROMs on unforced predictions by utilizing the normalized mean-trajectory-error and by analyzing the backbone curves.

3.3.1. Predicting an Unforced Trajectory

To predict an unforced beam using a ROM, first an initial condition of the FOM $\boldsymbol{q}(0)$ is mapped to the reduced space. Then, the reduced dynamics are employed to make a trajectory within this reduced space using the mapped initial condition. Finally, this reduced trajectory is mapped back to the full space to obtain the prediction $\boldsymbol{q}_{ROM}(t)$ of the unforced beam in the full space. The ROMs are again tested with a trajectory containing slightly smaller displacement than the one used for training, as discussed in subsection 3.1.1.

The trajectories in the reduced space for all six ROMs are shown in Figure 3.4. These trajectories involve the large, but not yet folding displacements (0.1412 m), meaning they exhibit non-linear behaviour. In all six cases, it is evident that at the beginning of the trajectory, the true (blue) and predicted (orange) paths overlap. However, after some time, they start to diverge.

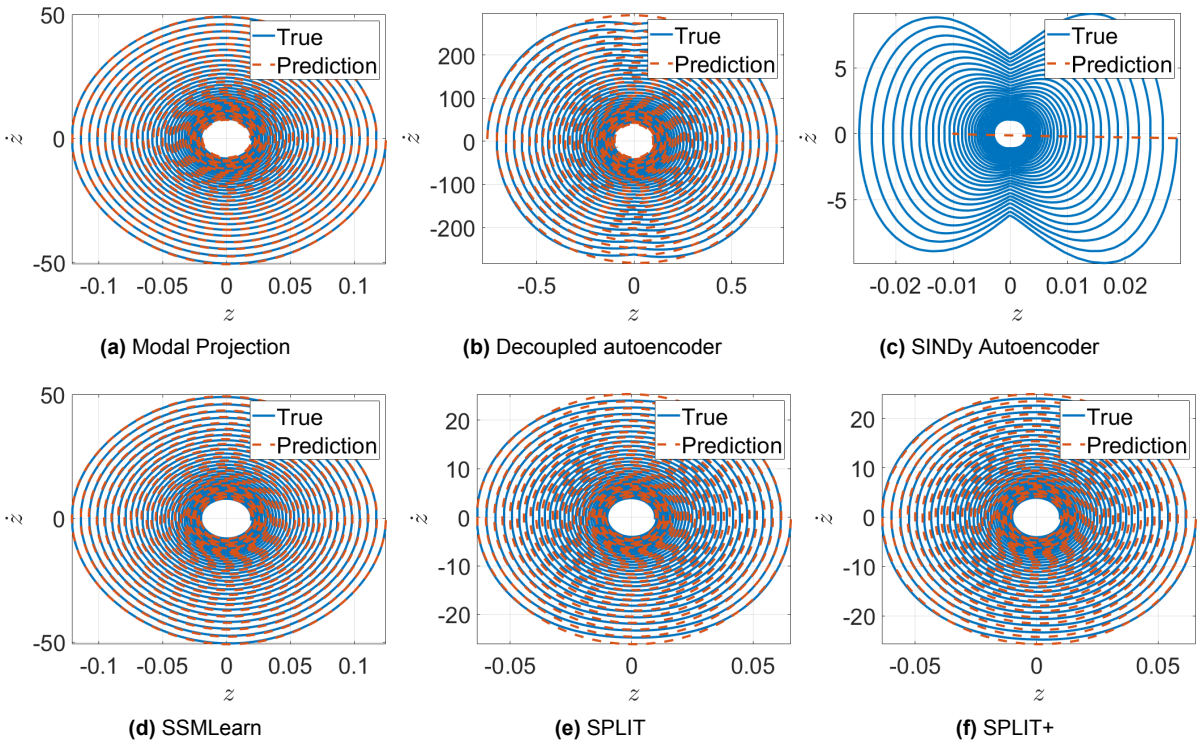
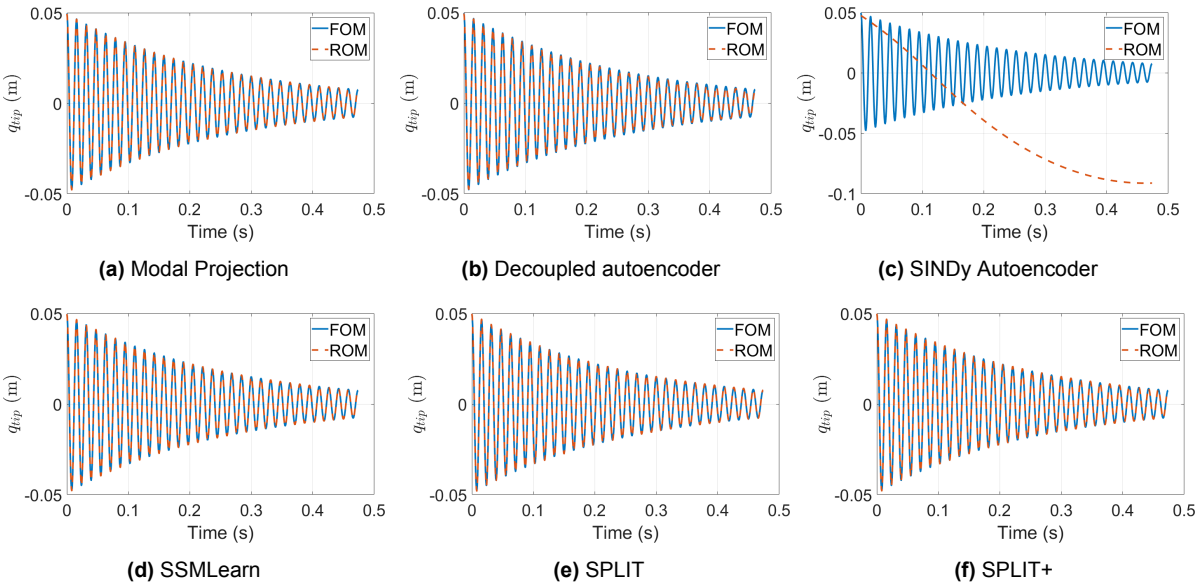


Figure 3.4: Trajectories in the reduced space for all six ROMs with large displacements (0.1412 m), exhibiting non-linear behaviour without folding. The predicted trajectories (in orange) initially align closely with the true trajectories (in blue), but begin to diverge over time. Distortions in the projection of 3.4b and 3.4c result in butterfly-like shapes rather than smooth circles.

In the cases of the decoupled autoencoder and the SINDy Autoencoder technique, shown in Figure 3.4b and 3.4c, a distortion occurs in the mapping: The (true) trajectory is being mapped to a more butterfly-like shape rather than a smooth circle. This distortion complicates the description of the dynamics in the reduced space, which causes the prediction to become worse. This issue is especially visible with the SINDy Autoencoder, where the prediction generated by the SINDy dynamics deviates entirely from the true trajectory. By using the linear terms of the reduced dynamics that are known from the FOM, which is done for SSMLearn, SPLIT and SPLIT+, this problem is overcome, as visible in Figure 3.4d, 3.4e and 3.4f.

These reduced trajectories can be mapped back to the full space to be compared to the trajectory of the FOM. The displacements of the tip of the beam over time for the different ROMs are illustrated in Figure 3.6. The same is done for the small displacements (0.0496 m) and folding configuration case (0.3568 m), shown in Figure 3.5 and 3.7.

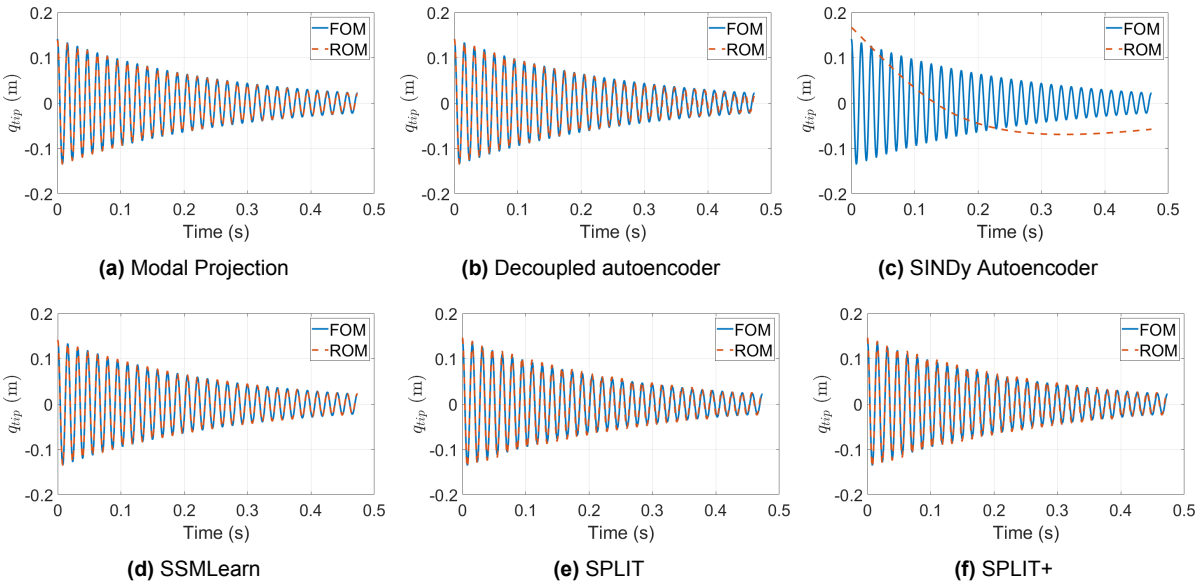
In the small displacement case, depicted in Figure 3.5, the reconstructed trajectories from most ROMs closely follow the true trajectory of the FOM, demonstrating that most methods perform well when the system exhibits mostly linear characteristics. For the large displacement case (Figure 3.6), discrepancies start to emerge between the ROMs and the FOM. This is due to the increased non-linearity in the system, making it more challenging for the reduced-order models to accurately capture the dynamics. The predicted trajectories initially align closely with the true trajectories, but begin to



(a) Modal Projection **(b)** Decoupled autoencoder **(c)** SINDy Autoencoder

(d) SSMLearn **(e)** SPLIT **(f)** SPLIT+

Figure 3.5: Trajectories of the tip of the beam for all six ROMs with small displacements (0.0496 m), exhibiting mostly linear behaviour. In this case, the predicted trajectories (in orange) align closely with the true trajectories (in blue). SINDy, in 3.5c fails to make a correct prediction.

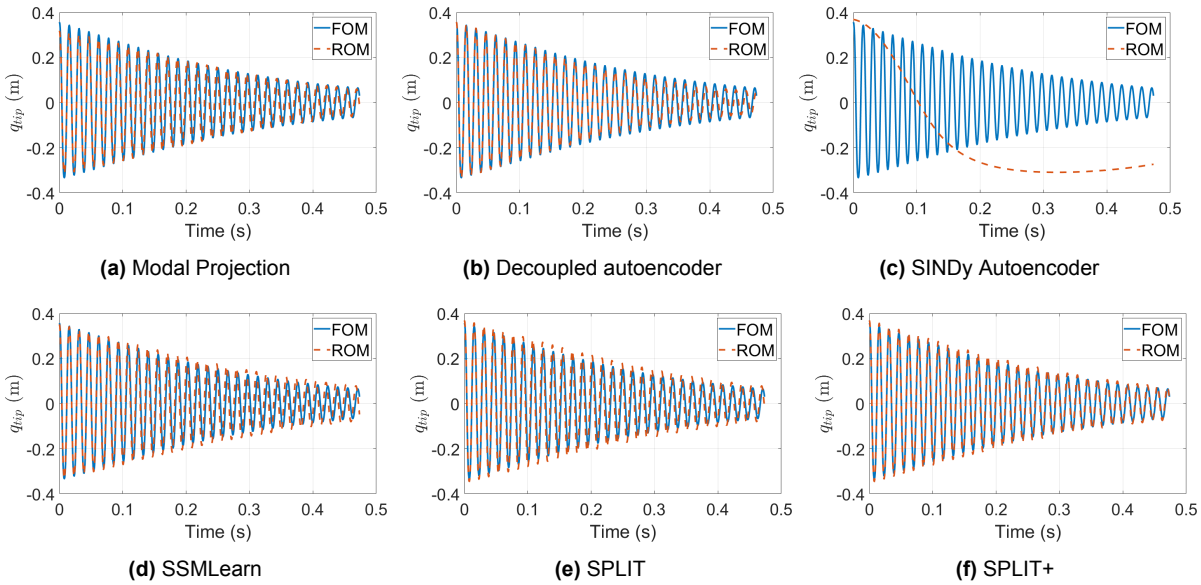


(a) Modal Projection **(b)** Decoupled autoencoder **(c)** SINDy Autoencoder

(d) SSMLearn **(e)** SPLIT **(f)** SPLIT+

Figure 3.6: Trajectories of the tip of the beam for all six ROMs with large displacements (0.1412 m), exhibiting non-linear behaviour without folding. The predicted trajectories (in orange) initially align closely with the true trajectories (in blue), but begin to diverge over time. Again, SINDy, in 3.6c fails to make a correct prediction.

diverge over time. When the displacements increase even further, to a folding configuration, illustrated in Figure 3.7, the differences between the ROMs become even more pronounced. The increased non-linearity and geometric complexity of the beam in this configuration cause most ROMs to produce a trajectory that significantly deviates from the FOM. The decoupled autoencoder and SPLIT+ are able to maintain better agreement with the FOM, with SPLIT+ showing a high level of accuracy in predicting both the amplitude and frequency of the beam, indicating its robustness in



(a) Modal Projection **(b)** Decoupled autoencoder **(c)** SINDy Autoencoder

(d) SSMLearn **(e)** SPLIT **(f)** SPLIT+

Figure 3.7: Trajectories of the tip of the beam for all six ROMs with displacements so large the beam folds over itself (0.3568 m). The increased complexity and non-linearity in this configuration cause significant deviations for some ROMs compared to the FOM. The decoupled autoencoder and SPLIT+ are able to maintain better agreement with the FOM, with SPLIT+ showing a high level of accuracy in predicting both the amplitude and frequency of the beam.

capturing the complex dynamics of the folding behaviour.

The SINDy Autoencoder seems to be having difficulties to accurately making a prediction, as is evident in Figure 3.5c, 3.6c, and 3.7c. This is somewhat anticipated, given the discrepancies in the eigenvalues of the linear part of the reduced dynamics, as seen in Table 3.2.

With the trajectories now expressed in the full space, the normalized mean-trajectory-error can be calculated to quantify the deviation between the displacement of the FOM $\boldsymbol{q}(t)$ and the displacement of the ROM $\boldsymbol{q}_{ROM}(t)$. The errors for the six different ROMs, across the three cases, are listed in Table 3.4. Similar to the reconstruction error, most methods perform well for small displacements. As the displacements increase, eventually leading to a folding configuration, the NMTE of the methods grow. This is in line with the observations from the figures. Nevertheless,

Table 3.4: Normalized mean-trajectory-errors for the unforced prediction $\boldsymbol{q}_{ROM}$ of the six different ROMs across the three cases. The decoupled autoencoder and SPLIT+ are the only two ROMs that have an error of less than 10%, even when the beam is in a folding configuration.

	Small displacements	**Large displacements**	**Folding configuration**
Modal Projection	0.99619%	3.199%	16.833%
Decoupled AE	1.3789%	4.1513%	5.8772%
SINDy AE	111.186%	50.0276%	70.8029%
SSMLearn	0.074255%	1.7065%	15.3383%
SPLIT	0.37445%	5.2151%	14.584%
SPLIT+	0.27806%	1.4485%	5.6117%

two ROMs manage to keep the error relatively small: the decoupled autoencoder and SPLIT+ methodologies maintain errors below 10%, even when the beam reaches the folding configuration. The SINDy Autoencoder seems to be failing in all three scenarios due to the butterfly-shaped patterns and its reduced dynamics, as shown in Figure 3.4c and Table 3.2.

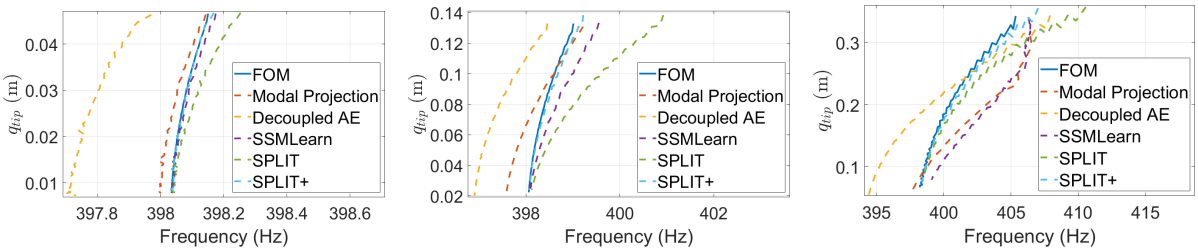
3.3.2. Backbone Curves

In addition to the NMTE of the unforced prediction, backbone curves are utilized as an evaluation metric to assess the performance of the ROMs. Backbone curves describe the relationship between the amplitude and frequency of oscillations in a non-linear system in the absence of external forcing. By comparing the backbone curves of different ROMs, their effectiveness in preserving the essential dynamical features of the original system can be assessed. For the extraction of the frequencies from the unforced time simulations, the method of Peak Finding and Fitting is employed [27].

The backbone curves for the different ROMs are analyzed across the three familiar cases: small displacements (0.0496 m), large displacements (0.1412 m), and a folding configuration (0.3568 m). The curves are shown in Figure 3.8. The backbone curve of the SINDy Autoencoder is not included in the figures, as the method of Peak Finding and Fitting could not be applied on the trajectories generated by this ROM (see Figure 3.5c, 3.6c, and 3.7c).

For the small displacements case, where the system its behaviour is nearly linear, almost all ROMs produce backbone curves that closely match the FOM (Figure 3.8a). This result is expected, as the non-linear effects are minimal, allowing even simpler models like the modal projection to perform adequately. However, the decoupled autoencoder does not perform as well in this scenario. The butterfly-like reduced dynamics may cause the backbone curve to deviate more noticeably from the FOM compared to the other ROMs.

As the displacements increase in the large and folding cases, differences between the ROMs become more apparent. For smaller amplitudes, SSMLearn and SPLIT approximate the FOM quite good. However, as the amplitude of the tip of the beam increases, even these ROMs start to show discrepancies. This suggests that while the



(a) Small displacements **(b)** Large displacements **(c)** Folding configuration

Figure 3.8: Backbone curves for the different ROMs across three cases: small displacements (0.0496 m), large displacements (0.1412 m), and a folding configuration (0.3568 m). For small displacements, most ROMs closely follow the FOM, although the decoupled autoencoder ROM struggles. As displacements increase, SPLIT+ is the method that approximate the backbone curves of the FOM the best.

trajectories of the predictions may have small errors, the underlying dynamics driving these predictions can still significantly diverge from those of the FOM. For SPLIT+ the backbone curves continue to closely approximate those of FOM, even in the folding case. This implicates that SPLIT+ has discovered dynamics that closely match those of FOM.

3.4. Forced Simulations

After evaluating the performance of the ROMs on unforced simulations, the next step is to assess their performance on forced simulations. This evaluation is motivated by the expectation that incorporating forcing into the system will not significantly disrupt the learned manifold, as spectral submanifolds are known to persist under small-amplitude forcing, thereby maintaining the ROM its predictive capabilities even under external influences [8]. The capability of making predictions of forced data, would indicate a level of generalization in the ROM, showing that it can adapt to new, unseen scenarios beyond the specific conditions on which it was trained. In this section, the ROMs will be tested on forced data to evaluate how well they can predict the system its response to external influences. This is again done using the NMTE, and by utilizing forced response curves (FRCs).

3.4.1. Predicting a Forced Trajectory

To predict a forced response using a ROM, the simulation in the reduced space performed again, but now with the inclusion of the external forcing function. The resulting reduced trajectory is mapped back to the full space to generate the prediction of the forced beam in the full space.

Recall that for the modal projection technique and SSMLearn (subsection 2.2.1 and 2.2.4), the external forcing function of the FOM could simply be mapped using the first eigenmode $\boldsymbol{\varphi}_1^\top$. For the non-linear autoencoder-based methods (subsection 2.2.2, 2.2.3, 2.2.5, and 2.2.6) external forcing could be introduced by using equation (2.16).

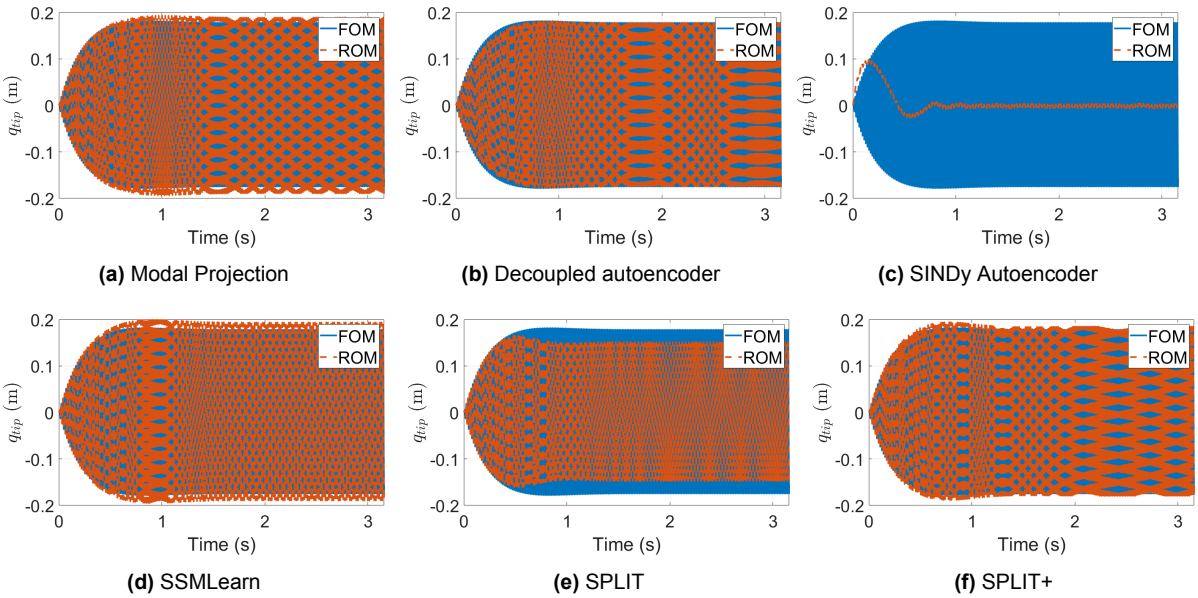
In this thesis, the external forcing function is represented by the (periodic) sine function

$$\varepsilon \boldsymbol{f}^{ext}(\Omega t) = \varepsilon \boldsymbol{M} \boldsymbol{\varphi}_1 \sin(\Omega t),$$

where ε is a parameter to control the amplitude of the forcing, and Ω the forcing frequency, which is chosen to be the (damped) eigenfrequency $\omega_1 \sqrt{1-\zeta_1^2}$. The initial displacement, velocity, and acceleration are set to zero.

Again, three different cases are considered: one with small forcing, one with large forcing, and one with forcing up to a folding configuration, corresponding to the amplitude parameters $\varepsilon = 1$, $\varepsilon = 600$, and $\varepsilon = 6000$, respectively. These forced simulations are conducted for a longer duration than the unforced simulations, to make sure fast transients die out, ensuring a steady-state response. For the non-folding cases, 200 periods T are simulated, and for the folding case, 400 periods are simulated.

The trajectory of the tip of the beam in the case of large forcing ($\varepsilon = 600$) are shown in Figure 3.9, for all six ROMs (in orange), together with the trajectory of the FOM (in blue). The plots reveal that most ROMs converge to a steady-state response



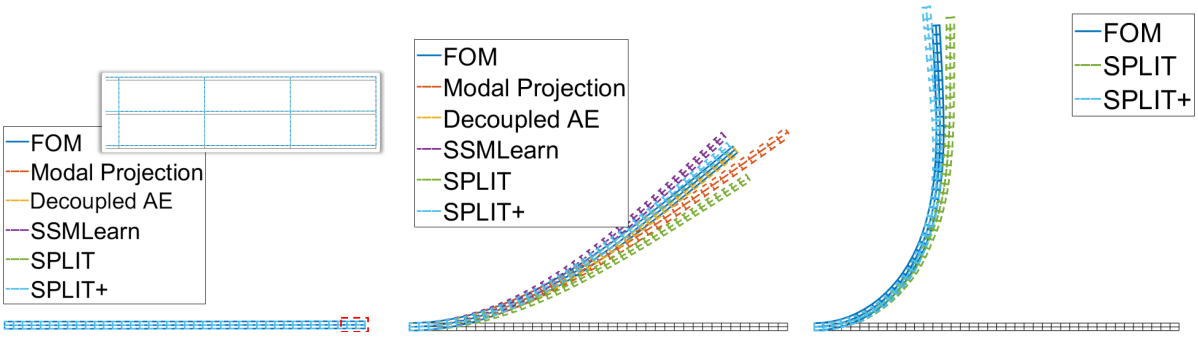
(a) Modal Projection **(b)** Decoupled autoencoder **(c)** SINDy Autoencoder

(d) SSMLearn **(e)** SPLIT **(f)** SPLIT+

Figure 3.9: Comparison of the trajectories under large forcing ($\varepsilon = 600$) for the six ROMs (orange) and the FOM (blue). Most ROMs converge to a steady-state response that is has an amplitude close to that of the FOM. The SINDy Autoencoder shows a response that is significantly inaccurate, highlighting its failure to capture the correct dynamics.

that has an amplitude close to that of the FOM. The SINDy Autoencoder however exhibits significant discrepancies in its response when compared to the FOM, indicating limitations in accurately capturing the underlying dynamics of the forced system as well.

The accuracy of the predicted trajectory of the ROM is of lesser importance, as long as the final forced response aligns with that of the FOM. Figure 3.10 contains the maximum displacement of this final response for the ROMs and the FOM. The displacement of the SINDy Autoencoder are not included, as the prediction of the forced response is significantly incorrect (recall Figure 3.9c).



(a) Small forcing **(b)** Large forcing **(c)** Folding configuration

Figure 3.10: Maximum displacement of the final response for the FOM and ROMs under varying forcing amplitudes. For small forcing ($\varepsilon = 1$), all ROMs closely match the FOM, capturing the linear dynamics effectively. As forcing increases to $\varepsilon = 600$, SSMLearn and SPLIT+ overestimate the response, while other ROMs underestimate it. Under extreme forcing ($\varepsilon = 6000$), most ROMs become unstable, with only SPLIT and SPLIT+ providing reasonable approximations.

Under small forcing ($\varepsilon = 1$), shown in Figure 3.10a, the maximum displacement is very close to the equilibrium of the beam. Upon zooming in, it becomes evident that all ROMs behave similarly and closely match the FOM, demonstrating that each method effectively captures the underlying linear dynamics. As the forcing amplitude increases to $\varepsilon = 600$ (Figure 3.10b), clear differences between the ROMs begin to appear. In this scenario, SSMLearn and SPLIT+ overestimate the response amplitude, whereas the other methods tend to underestimate it relative to the FOM. Additionally, it can be observed that the modal projection misses the compression of the beam. This is because the modal projection only considers the bending mode φ_1, neglecting other modes that contribute to the compressive behaviour of the beam.

Under extreme forcing with $\varepsilon = 6000$ (Figure 3.10c), many ROMs struggle to maintain stability. Modal projection and SSMLearn experience numerical instability, causing the reduced simulations to blow up ($z \to \infty$). As an example, the failing forced prediction made using the modal projection technique is shown in Figure 3.11. Similarly, the decoupled autoencoder method terminates prematurely due to difficulties in meeting integration tolerances without reducing the step size below acceptable limits.

It turns out that methods that seemed very promising, such as the decoupled autoencoder technique, ultimately do not work quite as hoped when conditions become too extreme. Only SPLIT and SPLIT+ provide reasonable approximations, with SPLIT+ achieving the best agreement with the FOM, albeit with some small inaccuracies.

The last period $(\tau, \tau + T]$ of the trajectories can be used in the calculation of the error of the response, using the NMTE. The results are presented in Table 3.5. In the case of small forcing, most ROMs perform well, due to the absence of very serious non-linearity's. However, the ROM with the decoupled autoencoder exhibits a slightly higher error, indicating some difficulty in accurately predicting forced responses even

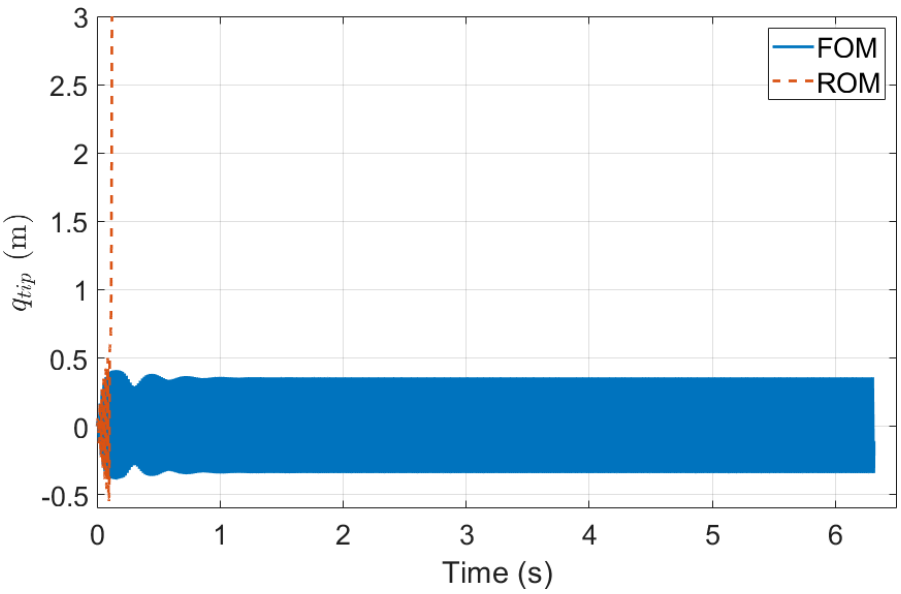


Figure 3.11: An example of numerical instability in the modal projection ROM under extreme forcing, causing the predicted displacement to diverge and the simulation to blow up.

Table 3.5: Normalized mean-trajectory-errors of the prediction of the forced response, for the six different ROMs across the various cases: small external forcing ($\varepsilon = 1$), large external forcing ($\varepsilon = 600$), and forcing so extreme, the beam reaches a folding configuration ($\varepsilon = 6000$). For the error, only the last period of the simulation is considered.

	Small forcing	**Large forcing**	**Folding configuration**
Modal Projection	0.60135%	17.3383%	Fails
Decoupled AE	5.7232%	2.3031%	Fails
SINDy AE	289.1655%	64.5305%	70.62%
SSMLearn	0.12503%	10.4902%	Fails
SLIT	0.18002%	20.9863%	4.755%
SPLIT+	0.18596%	9.3587%	6.0671%

under small external influences. The SINDy Autoencoder struggles significantly, similar to the unforced prediction.

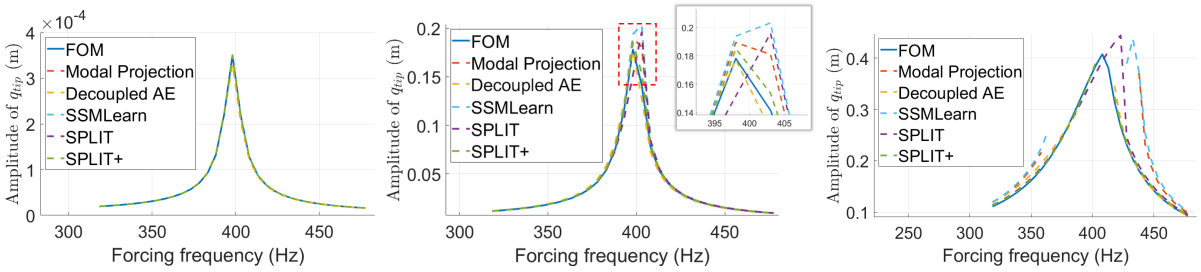
When the external forcing increases, so do the errors. In the case of forcing so large that the beam gets into a folding configuration, most ROMs fail to perform a simulation in the reduced space. The proposed method SPLIT, along with its extension SPLIT+, demonstrate excellent results, underscoring their strong capability to accurately capture and predict the complex dynamics of non-linear systems, even under challenging conditions.

3.4.2. Forced Response Curves

Finally, forced response curves (FRCs) are employed to evaluate the predictive accuracy of the ROMs. To construct the FRCs, the system is subjected to a periodic external force with varying frequencies Ω. The forced simulation is continued until fast transients dissipate, ensuring a steady-state response. The displacement of the tip of the beam $\boldsymbol{q}_{tip}$ during the last period, denoted by $[\tau, \tau + T)$, is then used to extract the amplitude and phase of the response. The FRC of the amplitude for small forcing ($\varepsilon = 1$), large forcing ($\varepsilon = 600$), and forcing until a folding configuration ($\varepsilon = 6000$) are shown in Figure 3.12. The FRCs of the phase are shown in Figure 3.13. The FRCs of the SINDy Autoencoder methodology are not included in the plots, as the predicted amplitude and phase could not be correctly extracted from their trajectory (recall Figure 3.9c).

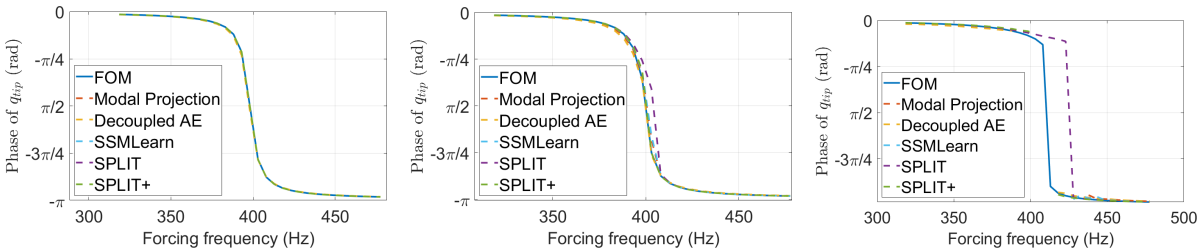
In the small forcing case, the FRCs of both amplitude (Figure 3.12a) and phase (Figure 3.13a) are generally well predicted by all five ROMs. The amplitude peaks near the system its eigenfrequency, and the phase shifts from 0 to $-\pi$ around this frequency, as expected. This demonstrates that the ROMs are able to capture the basic resonant behaviour of the system under small external perturbations.

As the forcing amplitude increases, the differences between the ROMs become more pronounced. Under the large forcing conditions, both SSMLearn and SPLIT predict the peak response at a frequency that is too high, as shown in Figure 3.12b. Additionally, these methods, along with modal projection, tend to overestimate the amplitude. SPLIT+ and the decoupled autoencoder method appear to provide the



(a) Small forcing **(b)** Large forcing **(c)** Folding configuration

Figure 3.12: Forced response curves of the amplitude of the tip of the beam under small external forcing ($\varepsilon = 1$), large external forcing ($\varepsilon = 600$), and forcing so large the response is a beam in folding configuration ($\varepsilon = 6000$). The amplitude peaks around the beam its eigenfrequency.



(a) Small forcing **(b)** Large forcing **(c)** Folding configuration

Figure 3.13: Forced response curves of the phase of the tip of the beam under small forcing ($\varepsilon = 1$), large forcing ($\varepsilon = 600$), and forcing so large the response is a beam in folding configuration ($\varepsilon = 6000$). The phase drops from 0 to $-\pi$ around the beam its frequency.

most accurate FRCs. Furthermore, Figure 3.13b shows that most ROMs can predict the phase response accurately, although SPLIT tends to overestimate it.

As the forcing frequency increases further to push the beam to a folding configuration, many simulations start to fail again. Only those simulations that could be finished are included in the plots in Figure 3.12c and 3.13c. The only ROM capable of generating a full FRC is SPLIT. However, its amplitude and phase predictions are inaccurate. While SPLIT+ is unable to produce a complete FRC, it most closely matches the FRCs of the FOM.

3.5. Evaluation of Computational Run Times

The computational performance of the ROMs is crucial, as one of the primary goals of MOR is to obtain ROMs with significantly shorter run times, compared to FOMs. The computational costs can be divided into two categories: offline and online costs. Offline costs refer to the computational resources used during the development of the ROM. It includes tasks like data collection, the training of the autoencoders, or the fitting of the reduced dynamics. The offline phase typically requires some computational time and resources, but is done only once. The results from the offline phase can then be used repeatedly during the online phase. This involves performing simulations, both unforced and forced, and the generation of the FRCs.

In this study, the autoencoders are trained using TensorFlow from Python. All remaining simulations are performed in MATLAB [45]. The computations are executed

Table 3.6: Comparison of run times of the FOM and the various ROMs, including offline costs (generation of data, mapping and reduced dynamics) and online costs (unforced simulation, forced simulation, and FRC generation).

	Offline costs			Online costs		
	Generate training data	**Obtain mapping**	**Obtain reduced dynamics**	**Unforced simulation**	**Forced simulation**	**Generate FRC**
FOM	-	-	-	20m 28.3s	136m 48.4s	4487m 15.5s
Modal Projection	20m 28.3s	0.005s	0.02s	0.2s	1.5s	45.6s
Decoupled autoencoder	20m 28.3s	96m 33.6s	0.1s	0.2s	1.5s	48.8s
SINDy AE	20m 28.3s	99m 7.2s		0.2s	1.3s	44.4s
SSMLearn	20m 28.3s	0.3s	0.06s	0.2s	1.6s	149.0s
SPLIT	20m 28.3s	98m 23.2s		0.2s	1.4s	48.0s
SPLIT+	20m 28.3s	98m 23.2s	0.1s	0.2s	1.4s	47.9s

on a machine equipped with a 2.20GHz Intel Core i7 hexa-core CPU, 16GB of RAM, with both an Intel UHD Graphics 630 and a NVIDIA Quadro P1000 GPU [53]. The run times of all online and offline processes are measured, and shown in Table 3.6.

For all MOR techniques, an unforced simulation of the FOM is required for training. The simulation consists of 30 periods T, and costs 20 minutes and 28.3 seconds. Obtaining the mapping to the reduced space and the reduced dynamics varies significantly among the different methods. The simplest technique, using modal projection, has the lowest offline costs for obtaining the mapping and reduced dynamics, taking just a fraction of a second. In contrast, methods involving autoencoders, such as the decoupled autoencoder, SINDy Autoencoder, SPLIT, and SPLIT+, have significantly higher costs due to the training of neural networks, which takes around 96 to 99 minutes. SSMLearn, while also utilizing a more sophisticated method than modal projection, still has considerably lower offline costs than the autoencoder-based methods.

In terms of online costs, which include unforced and forced simulations as well as generating FRCs, all ROMs demonstrate substantial improvements over the FOM. The unforced simulation (30 periods) for the FOM takes over 20 minutes, while all ROMs complete the same task in approximately 0.2 seconds. This dramatic reduction highlights the efficiency gains possible with ROMs. Forced simulations, which consist of 200 periods T show a similar trend, with the FOM taking over 136 minutes, whereas all ROMs complete the task in about 1.3 to 1.6 seconds. When generating FRCs, 33 forced simulations are run. The ROMs generally perform well, completing the task in under 50 seconds, except for SSMLearn, which takes 149 seconds. This is however still much faster than using the FOM, as this takes 4487 minutes.

It is important to keep in mind that while methods like modal projection offer the lowest computational costs, they are generally less effective for capturing complex dynamics (as seen in section 3.3 and 3.4). The more accurate autoencoder-based methods like SPLIT and SPLIT+ have a higher computational complexity. However, note that for example for SPLIT, performing a forced simulation, including the offline costs of data collection and training the autoencoder, is still faster than performing a

forced simulation with the FOM, as

$$\underbrace{20\text{m} + 98\text{m} + 1.4\text{s} \approx 118\text{m}}_{\substack{\text{Total time required for a} \\ \text{forced simulation using SPLIT}}} \quad < \quad \underbrace{136\text{m}}_{\substack{\text{Time required for a for-} \\ \text{ced simulation using FOM}}}.$$

In addition, the generation of an FRC shows an even more substantial improvement, with a performance gain of a factor 27, as

$$\frac{\text{Time required for FRC with FOM}}{\text{Time required for FRC with SPLIT}} = \frac{4487\text{m}}{20\text{m} + 98\text{m} + 48\text{m}} = 27.$$

This again demonstrates the effectiveness of ROMs in significantly reducing computational time. The analysis of run times illustrate that ROMs significantly outperform the FOM in terms of computational efficiency.

4

Discussion

The exploration of various model order reduction (MOR) techniques in this study has provided valuable insights into their strengths and limitations when applied to nonlinear dynamical systems such as the 2D cantilever beam. Reduced order models (ROMs) capable of accurately predicting system behaviour, while significantly reducing computational complexity, have been developed, but the results also highlighted some areas for improvement. This section reflects on some of the choices made during the study and suggest promising directions for future research.

4.1. General Points of Discussion

First, while SINDy Autoencoders have shown success in various applications [10], their performance in modeling the cantilever beam in this study was not satisfactory. The SINDy Autoencoder was unable to accurately capture the dynamics of the beam, even under moderately challenging conditions, as indicated in Table 3.2. This limitation led to incorrect predictions, as illustrated in Figure 3.5c, 3.6c, 3.7c, and 3.9c. This raises questions about the applicability of this method to certain types of systems. The outcome might be due to the particularly complex behaviour of the beam, which was too difficult to model. Alternatively, it is possible that the chosen parameters for the SINDy Autoencoder, although recommended by prior research [10], were not optimal for the specific case of this study. This suggests that while SINDy Autoencoders are robust tools, their application may require more customized parameter tuning or modifications to handle specific dynamical systems effectively.

Moreover, while advanced techniques like autoencoders were employed for data reduction, the fitting of the reduced dynamics in this study often relied on basic polynomial regression. Although autoencoders successfully identified lower-dimensional manifolds, the subsequent step of fitting dynamics onto these manifolds using simple polynomial regression might have constrained the overall accuracy of the ROMs. Enhancing this step could be a significant area for improvement. For instance, incorporating higher-order polynomial terms or leveraging advanced machine learning methods such as long short-term memory (LSTM) networks, known for their ability to handle time-series data, could yield better results, as demonstrated in other studies

[12, 18, 40, 47].

4.2. Research Outlook for Proposed Methods

The proposed techniques, SPLIT and SPLIT+, demonstrated enhanced performance in model order reduction. To build on this success, a few recommendations can be made. One consideration is the assumption of prior knowledge regarding the linear part of the governing equations. While this requirement enhances the effectiveness of SPLIT and SPLIT+, it may limit their versatility compared to purely data-driven approaches, such as SINDy Autoencoders. Exploring methods that reduce this dependency on the prior knowledge of the governing equations could broaden the applicability of these techniques.

In addition, a basic network architecture, a multilayer perceptron, was utilized for the autoencoder in this study. Ongoing research offers intriguing extensions, such as convolutional autoencoders that utilize convolutional neural networks (CNNs) to capture spatial correlations within the data. Since neighboring degrees of freedom correspond to adjacent points on the beam, CNNs could effectively capture these spatial correlations, enhancing the model order reduction tasks [18, 42]. Other innovative approaches could involve adapting the loss function of the autoencoder to automatically determine the latent space dimension [48] or ensuring that the decoder serves as the exact inverse of the encoder [38]. These advancements offer promising pathways for the development of even more precise and reliable MOR techniques.

Another promising direction for future research is to modify SPLIT and SPLIT+ to identify normal form dynamics, enabling the use of a parsimonious model to describe the reduced dynamics [28]. This enhancement could make it possible to analytically calculate forced response curves (FRCs) [9], thereby further reducing the computational cost associated with generating FRCs. Such advancements would make SPLIT and SPLIT+ even more competitive with existing methods like SSMLearn.

5

Conclusion

This study has explored the application of autoencoders in enhancing model order reduction (MOR) techniques for non-linear dynamical systems. It specifically focused on their capacity to identify manifolds with their reduced dynamics, and the effectiveness of these dynamics in accurately predicting both unforced (seen scenarios) and forced responses (unseen scenarios). The research questions posed aimed to evaluate the potential of autoencoders in MOR, assess the accuracy of the reduced dynamics, and investigate both established and novel techniques for improvement. This section will first address the sub-questions, followed by a comprehensive conclusion that answers the main research question.

What are the limitations of the established data-driven and data-assisted techniques for model order reduction?

The study has highlighted several limitations of established data-driven and data-assisted MOR techniques, specifically focusing on modal projection, SSMLearn, and SINDy Autoencoders.

Modal projection for example, is based on linear mappings (subsection 2.2.1), and therefore struggles to accurately represent the non-linear behaviour of the system. While it is effective for capturing the linear components, it fails to do so for the non-linear dynamics, as evidenced by the increased reconstruction error under non-linear conditions shown in Table 3.3. The inability to capture these non-linear interactions limits its applicability for complex systems like the cantilever beam, where non-linear phenomena come into play.

SSMLearn is able to capture non-linearity's by using a graph-style parameterization to identify non-linear manifolds (subsection 2.2.4). The method however becomes unreliable when the learned manifold folds over its graphing subspace, for example when a two-dimensional cantilever beam enters configurations involving folding. As shown in Table 3.4, SSMLearn is not able to reproduce trajectories of a folding beam, without introducing significant errors.

As discussed in section 4.1, the SINDy Autoencoder method (subsection 2.2.3) was unable to capture the dynamics of the beam. Even for moderate displacements,

SINDy exhibited significant discrepancies from the true system behaviour, as for example reflected in the eigenvalues (Table 3.2). This suggests that the effectiveness of the SINDy Autoencoders may be limited in scenarios involving highly complex or strongly non-linear dynamics.

Additionally, the established methods examined in this study (e.g. modal projection, SSMLearn, and SINDy Autoencoders) were all found to fail when extremely large external forcing was introduced, which caused the steady-state response of the beam to fold over itself. This is evident in Table 3.5. The inability to handle such high forcing amplitudes resulted in numerical instabilities and crashing simulations. This breakdown reveals that these MOR techniques struggle with scenarios involving complex non-linear behaviour, limiting their effectiveness in a broader range of practical applications.

Are autoencoders able to correctly identify the non-linear manifolds that dominate the systems its response and learn their reduced dynamics?

Autoencoders have demonstrated potential in identifying non-linear manifolds and learning their reduced dynamics, as evidenced in this thesis. However, it has been shown that simply using autoencoders, without incorporating any linear terms during training, as done in the decoupled autoencoder approach (subsection 2.2.2) and the SINDy Autoencoders (subsection 2.2.3), can lead to manifolds with a non-smooth, butterfly-like shape, as seen in Figure 3.4b and 3.4c. This distortion indicates that while the autoencoder successfully reduces dimensionality, the identified latent variables and the corresponding reduced dynamics may not accurately reflect the true system behaviour, as for example shown in Figure 3.8a, making the reduced order model (ROM) less effective.

The proposed method SPLIT (subsection 2.2.5), addresses these limitations by integrating the strengths of several established approaches. It combines the joint learning of autoencoders and reduced dynamics, inspired by the SINDy Autoencoders, while also incorporating the linear dynamics obtained from modal projection, similar to SSMLearn. This combination enables SPLIT to maintain the correct linear behaviour within the learned manifold, ensuring a meaningful representation of the system its dynamics. Further refinement can be achieved through polynomial regression in the post-processing step, as done in the extension SPLIT+ (subsection 2.2.6), which has been shown to yield even better predictions in certain scenarios by correcting small discrepancies in the learned dynamics.

Overall, while autoencoders alone may fall short in accurately identifying and learning meaningful non-linear manifolds, integrating them with more sophisticated approaches, as demonstrated by SPLIT and SPLIT+, can significantly improve the quality of the ROMs.

How well can the reduced dynamics on these manifolds make predictions for unseen data?

The ability to predict non-linear forced responses (with displacements within the range of training), despite being trained on only unforced data, arises since the MOR

techniques approximate spectral submanifolds, which persist under small-amplitude forcing [8]. For small displacements, all ROMs were indeed able to accurately predict forced simulations (Table 3.5) and forced response curves (Figure 3.12a and 3.13a), showing good agreement with the full order model (FOM). However, as the forcing amplitude increased and the beam was pushed into a highly non-linear folding configuration, many ROMs encountered numerical instabilities or convergence issues, leading to incomplete or inaccurate simulations. SPLIT and its extension SPLIT+, however, have demonstrated strong predictive capabilities for unseen data, even under these challenging conditions.

In the numerical experiments conducted in this thesis, SPLIT was the only method able to generate a complete forced response curve (FRC) for the folding configuration, although its amplitude and phase predictions deviated from those of the FOM, as shown in Figure 3.12b and 3.13b. While SPLIT+ could not produce a complete FRC under these extreme conditions, it exhibited the best overall agreement with the FOM, closely matching its amplitude and phase behaviour where it succeeded. This suggests that the enhanced optimization process and additional regression step in SPLIT+ provide improved predictive power and stability, making it a more reliable method for the MOR of complex non-linear systems, even when predicting unseen data.

How can autoencoders enhance non-intrusive model order reduction for non-linear dynamical systems described by high-dimensional finite element models?

In conclusion, autoencoders have demonstrated significant potential for enhancing MOR in non-linear dynamical systems by learning lower-dimensional representations using data. They are capable of identifying manifolds that capture the essential dynamics of these systems.

This study has shown that while autoencoders can successfully perform dimensionality reduction, their performance in terms of accurately capturing and predicting system dynamics can vary, depending on the complexity of the behaviour of the system, and the specific parameters chosen for the ROM. This inconsistency is for example highlighted through methods like SINDy Autoencoders (subsection 2.2.3), which struggled to model the underlying dynamics accurately.

The development of the proposed methods SPLIT and SPLIT+ (subsection 2.2.5 and 2.2.6), represents a significant advancement by integrating the strengths of existing techniques and carefully optimizing key parameters. SPLIT successfully combined the autoencoders its capability to learn meaningful manifolds with the aid of linear dynamics, resulting in a robust method for reducing model complexity while preserving dynamic accuracy. By jointly learning the manifold and its reduced dynamics, SPLIT offered a reliable ROM that accurately captured both linear and non-linear behaviour. Its extension, SPLIT+, further improved performance by refining the learned dynamics through polynomial regression, leading to even better predictions in complex scenarios.

In summary, while autoencoders offer valuable tools for dimensionality reduction, their application in MOR requires careful integration with methods that ensure accu-

rate modeling of the dynamics. The success of SPLIT and SPLIT+ demonstrates the potential for autoencoder-based MOR techniques by combining state-of-the-art dimensionality reduction with strategies that enforce accurate dynamic representations. This integrated approach forms the basis for further improvements, enabling even more effective and robust MOR techniques for nonlinear dynamical systems. Some recommendations for further improvements were discussed in section 4.2.

References

[1] Pierre Baldi and Kurt Hornik. “Neural networks and principal component analysis: Learning from examples without local minima”. In: *Neural Networks* 2.1 (Jan. 1989), pp. 53–58. ISSN: 0893-6080. DOI: 10.1016/0893-6080(89)90014-2.

[2] G Berkooz, P Holmes, and J L Lumley. “The Proper Orthogonal Decomposition in the Analysis of Turbulent Flows”. In: *Annual Review of Fluid Mechanics* 25.1 (Jan. 1993), pp. 539–575. ISSN: 1545-4479. DOI: 10.1146/annurev.fl.25.010193.002543.

[3] Hans Georg Bock, ed. *Model Order Reduction: Theory, Research Aspects and Applications*. SpringerLink. Includes bibliographical references and index. Berlin: Springer, 2008. ISBN: 978-3540788416.

[4] Jamie Bowers, Eli Durant, and Reetesh Ranjan. “Application of Intrusive and Non-Intrusive Reduced Order Modeling Techniques for Simulation of Turbulent Premixed Flames”. In: *AIAA Propulsion and Energy 2021 Forum*. American Institute of Aeronautics and Astronautics, July 2021. DOI: 10.2514/6.2021-3634.

[5] Steven L. Brunton. *Data-driven science and engineering. Machine learning, dynamical systems, and control*. Ed. by J. Nathan Kutz. 2nd revised edition. Literaturverzeichnis: Seite 552 - 587. Cambridge, United Kingdom: Cambridge University Press, 2022. 1614 pp. ISBN: 978-1009089517.

[6] A. Cammarano et al. “Bifurcations of backbone curves for systems of coupled nonlinear two mass oscillator”. In: *Nonlinear Dynamics* 77.1–2 (Feb. 2014), pp. 311–320. ISSN: 1573-269X. DOI: 10.1007/s11071-014-1295-3.

[7] M. Cenedese et al. “Data-driven nonlinear model reduction to spectral submanifolds in mechanical systems”. In: *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences* 380.2229 (June 2022). ISSN: 1471-2962. DOI: 10.1098/rsta.2021.0194.

[8] Mattia Cenedese et al. *Data-Assisted Non-Intrusive Model Reduction for Forced Nonlinear Finite Elements Models*. 2023. DOI: 10.48550/ARXIV.2311.17865.

[9] Mattia Cenedese et al. “Data-driven modeling and prediction of non-linearizable dynamics via spectral submanifolds”. In: *Nature Communications* 13.1 (Feb. 2022). ISSN: 2041-1723. DOI: 10.1038/s41467-022-28518-y.

[10] Kathleen Champion et al. “Data-driven discovery of coordinates and governing equations”. In: *Proceedings of the National Academy of Sciences* 116.45 (Oct. 2019), pp. 22445–22451. ISSN: 1091-6490. DOI: 10.1073/pnas.1906995116.

[11] Jay L. Devore. *Probability and statistics for engineering and the sciences*. 4. ed., [6.] print. Pacific Grove [u.a.]: Duxbury Press, 1997. 743 pp. ISBN: 0-534-24264-2.

[12] Hamidreza Eivazi et al. “Deep neural networks for nonlinear model order reduction of unsteady flows”. In: *Physics of Fluids* 32.10 (Oct. 2020). ISSN: 1089-7666. DOI: 10.1063/5.0020526.

[13] Michel Geradin. *Mechanical Vibrations. Theory and Application to Structural Dynamics*. 1st ed. New York Academy of Sciences Series. Description based on publisher supplied metadata and other sources. Newark: John Wiley & Sons, Incorporated, 2015. 1617 pp. ISBN: 978-1118900185.

[14] J.D Gergonne. “The application of the method of least squares to the interpolation of sequences”. In: *Historia Mathematica* 1.4 (Nov. 1974), pp. 439–447. ISSN: 0315-0860. DOI: 10.1016/0315-0860(74)90034-2.

[15] Amir Hossein Ghasemikaram et al. “Flutter suppression of an aircraft wing with a flexibly mounted mass using magneto-rheological damper”. In: *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering* 234.3 (Nov. 2019), pp. 827–839. ISSN: 2041-3025. DOI: 10.1177/0954410019887039.

[16] Xavier Glorot and Yoshua Bengio. “Understanding the difficulty of training deep feedforward neural networks”. In: *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*. Ed. by Yee Whye Teh and Mike Titterington. Vol. 9. Proceedings of Machine Learning Research. Chia Laguna Resort, Sardinia, Italy: PMLR, May 2010, pp. 249–256. URL: https://proceedings.mlr.press/v9/glorot10a.html.

[17] Arthur Stanley Goldberger. *Econometric theory*. Faks.-repr. d. Ausg Wiley, 1964. Ann Arbor, Mich.: UMI, 1996. 399 pp. ISBN: 0471311014.

[18] Francisco J. Gonzalez and Maciej Balajewicz. *Deep convolutional recurrent autoencoders for learning low-dimensional feature dynamics of fluid systems*. 2018. DOI: 10.48550/ARXIV.1808.01346.

[19] Jonathan H. Tu et al. “On dynamic mode decomposition: Theory and applications”. In: *Journal of Computational Dynamics* 1.2 (2014), pp. 391–421. ISSN: 2158-2505. DOI: 10.3934/jcd.2014.1.391.

[20] George Haller and Sten Ponsioen. “Nonlinear normal modes and spectral submanifolds: existence, uniqueness and use in model reduction”. In: *Nonlinear Dynamics* 86.3 (Aug. 2016), pp. 1493–1534. ISSN: 1573-269X. DOI: 10.1007/s11071-016-2974-z.

[21] Haller, George, Jain, Shobhit, and Cenedese, Mattia. “Dynamics-based machine learning for nonlinearizable phenomena. Data-driven reduced models on spectral submanifolds”. en. In: *SIAM News* 55(5) (June 2022), pp. 1–4. DOI: 10.3929/ETHZ-B-000587451.

[22] David Hartman and Lalit K. Mestha. “A deep learning framework for model reduction of dynamical systems”. In: *2017 IEEE Conference on Control Technology and Applications (CCTA)*. IEEE, Aug. 2017. DOI: 10.1109/ccta.2017.8062736.

[23] Shobhit Jain, Thomas Breunung, and George Haller. “Fast computation of steady-state response for high-degree-of-freedom nonlinear systems”. In: *Nonlinear Dynamics* 97.1 (May 2019), pp. 313–341. ISSN: 1573-269X. DOI: 10.1007/s11071-019-04971-1.

[24] Shobhit Jain and George Haller. “How to compute invariant manifolds and their reduced dynamics in high-dimensional finite element models”. In: *Nonlinear Dynamics* 107.2 (Oct. 2021), pp. 1417–1450. ISSN: 1573-269X. DOI: 10.1007/s11071-021-06957-4.

[25] Shobhit Jain, Jacopo Marconi, and Paolo Tiso. *YetAnotherFEcode*. 2022. DOI: 10.5281/ZENODO.4011281.

[26] Shobhit Jain et al. *SSMTool 2.5: Computation of invariant manifolds in high-dimensional mechanics problems*. 2023. DOI: 10.5281/ZENODO.4614201.

[27] Mengshi Jin et al. “Identification of Instantaneous Frequency and Damping From Transient Decay Data”. In: *Journal of Vibration and Acoustics* 142.5 (June 2020). ISSN: 1528-8927. DOI: 10.1115/1.4047416.

[28] Manu Kalia et al. *Learning normal form autoencoders for data-driven discovery of universal,parameter-dependent governing equations*. 2021. DOI: 10.48550/ARXIV.2106.05102.

[29] Gaetan Kerschen et al. “The Method of Proper Orthogonal Decomposition for Dynamical Characterization and Order Reduction of Mechanical Systems: An Overview”. In: *Nonlinear Dynamics* 41.1–3 (Aug. 2005), pp. 147–169. ISSN: 1573-269X. DOI: 10.1007/s11071-005-2803-2.

[30] Kwangkeun Kim et al. “Nonlinear Reduced Order Modeling of Flat Cantilevered Structures”. In: *50th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*. American Institute of Aeronautics and Astronautics, May 2009. DOI: 10.2514/6.2009-2492.

[31] Diederik P. Kingma and Jimmy Ba. *Adam: A Method for Stochastic Optimization*. 2014. DOI: 10.48550/ARXIV.1412.6980.

[32] Toni Lassila et al. “Model Order Reduction in Fluid Dynamics: Challenges and Perspectives”. In: *Reduced Order Methods for Modeling and Computational Reduction*. Springer International Publishing, 2014, pp. 235–273. ISBN: 978-3319020907. DOI: 10.1007/978-3-319-02090-7_9.

[33] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. “Deep learning”. In: *Nature* 521.7553 (May 2015), pp. 436–444. ISSN: 1476-4687. DOI: 10.1038/nature14539.

[34] John A. Lee. *Nonlinear dimensionality reduction*. Ed. by Michel Verleysen. Information Science and Statistics Ser. Includes bibliographical references (p. 283-295) and index. New York: Springer, 2007. 308 pp. ISBN: 978-0387393513.

[35] Mingwu Li, Shobhit Jain, and George Haller. “Nonlinear analysis of forced mechanical systemswith internal resonance using spectral submanifolds, Part I: Periodic response and forced response curve”. In: *Nonlinear Dynamics* 110.2 (Aug. 2022), pp. 1005–1043. ISSN: 1573-269X. DOI: 10.1007/s11071-022-07714-x.

[36] M. Merriman. *A List of Writings Relating to the Method of Least Squares: With Historical and Critical Notes*. Transactions of the Connecticut Academy of Arts and Sciences. Academy, 1877. ISBN: 978-1166426859. URL: https://books.google.nl/books?id=8tIGAAAAYAAJ.

[37] Mario Ohlberger and Stephan Rave. “Reduced Basis Methods: Success, Limitations and Future Challenges”. In: *Proceedings of ALGORITMY 2016* (2016), pp. 1–12. DOI: 10.48550/ARXIV.1511.02021.

[38] Samuel E. Otto, Gregory R. Macchio, and Clarence W. Rowley. “Learning nonlinear projections for reduced-order modeling of dynamical systems using constrained autoencoders”. In: *Chaos: An Interdisciplinary Journal of Nonlinear Science* 33.11 (Nov. 2023). ISSN: 1089-7682. DOI: 10.1063/5.0169688.

[39] Shafic S. Oueini, Char-Ming Chin, and Ali H. Nayfeh. “Dynamics of a Cubic Nonlinear Vibration Absorber”. In: *Nonlinear Dynamics* 20.3 (1999), pp. 283–295. ISSN: 0924-090X. DOI: 10.1023/a:1008358825502.

[40] Jose L. Paniagua and Jesus A. Lopez. “Dimensionality Reduction Applied to Time Response of Linear Systems Using Autoencoders”. In: *2019 IEEE Colombian Conference on Applications in Computational Intelligence (ColCACI)*. IEEE, June 2019. DOI: 10.1109/colcaci.2019.8781797.

[41] S. Pawar et al. “A deep learning enabler for nonintrusive reduced order modeling of fluid flows”. In: *Physics of Fluids* 31.8 (Aug. 2019). ISSN: 1089-7666. DOI: 10.1063/1.5113494.

[42] Alberto Racca, Nguyen Anh Khoa Doan, and Luca Magri. “Predicting turbulent dynamics with the convolutional autoencoder echo state network”. In: *Journal of Fluid Mechanics* 975 (Nov. 2023). ISSN: 1469-7645. DOI: 10.1017/jfm.2023.716.

[43] G. Rozza, D. B. P. Huynh, and A. T. Patera. “Reduced Basis Approximation and a Posteriori Error Estimation for Affinely Parametrized Elliptic Coercive Partial Differential Equations: Application to Transport and Continuum Mechanics”. In: *Archives of Computational Methods in Engineering* 15.3 (May 2008), pp. 229–275. ISSN: 1886-1784. DOI: 10.1007/s11831-008-9019-9.

[44] Peter J. Schmid. “Dynamic mode decomposition of numerical and experimental data”. In: *Journal of Fluid Mechanics* 656 (July 2010), pp. 5–28. ISSN: 1469-7645. DOI: 10.1017/s0022112010001217.

[45] Aron Schouten. *Simultaneous Projection and Linear Informed Training (SPLIT)*. Version 1.0.0. Oct. 2024. URL: https://github.com/AronSchouten/SPLIT.

[46] Yichang Shen et al. “Comparison of Reduction Methods for Finite Element Geometrically Nonlinear Beam Structures”. In: *Vibration* 4.1 (Mar. 2021), pp. 175–204. ISSN: 2571-631X. DOI: 10.3390/vibration4010014.

[47] Thomas Simpson, Nikolaos Dervilis, and Eleni Chatzi. “Machine Learning Approach to Model Order Reduction of Nonlinear Systems via Autoencoder and LSTM Networks”. In: *Journal of Engineering Mechanics* 147.10 (Oct. 2021). ISSN: 1943-7889. DOI: 10.1061/(asce)em.1943-7889.0001971.

[48] David Sondak and Pavlos Protopapas. “Learning a reduced basis of dynamical systems using an autoencoder”. In: *Physical Review E* 104.3 (Sept. 2021), p. 034202. ISSN: 2470-0053. DOI: 10.1103/physreve.104.034202.

[49] Paolo Tiso and Morteza Karamooz Mahdiabadi. “4 Modal methods for reduced order modeling”. In: *System- and Data-Driven Methods and Algorithms*. De Gruyter, Oct. 2021, pp. 97–138. ISBN: 978-3110498967. DOI: 10.1515/9783110498967-004.

[50] Cyril Touzé and Olivier Thomas. “Reduced-order modeling for a cantilever beam subjected to harmonic forcing”. In: *EUROMECH 457, Nonlinear modes of vibrating systems*. Fréjus, France, June 2004. URL: https://ensta-paris.hal.science/hal-01154710.

[51] J. H. Verner. “Numerically optimal Runge–Kutta pairs with interpolants”. In: *Numerical Algorithms* 53.2–3 (Apr. 2009), pp. 383–396. ISSN: 1572-9265. DOI: 10.1007/s11075-009-9290-3.

[52] Alessandra Vizzaccaro et al. “High order direct parametrisation of invariant manifolds for model order reduction of finite element structures: application to large amplitude vibrations and uncovering of a folding point”. In: *Nonlinear Dynamics* 110.1 (July 2022), pp. 525–571. ISSN: 1573-269X. DOI: 10.1007/s11071-022-07651-9.

[53] WikiChip. *Core i7-8750H - Intel - WikiChip*. https://en.wikichip.org/wiki/intel/core_i7/i7-8750h. Retrieved January 14, 2023. Oct. 2022.

[54] Süleyman Yıldız, Murat Uzunca, and Bülent Karasözen. “Intrusive and non-intrusive reduced order modeling of the rotating thermal shallow water equation”. In: (2021). DOI: 10.48550/ARXIV.2104.00213.