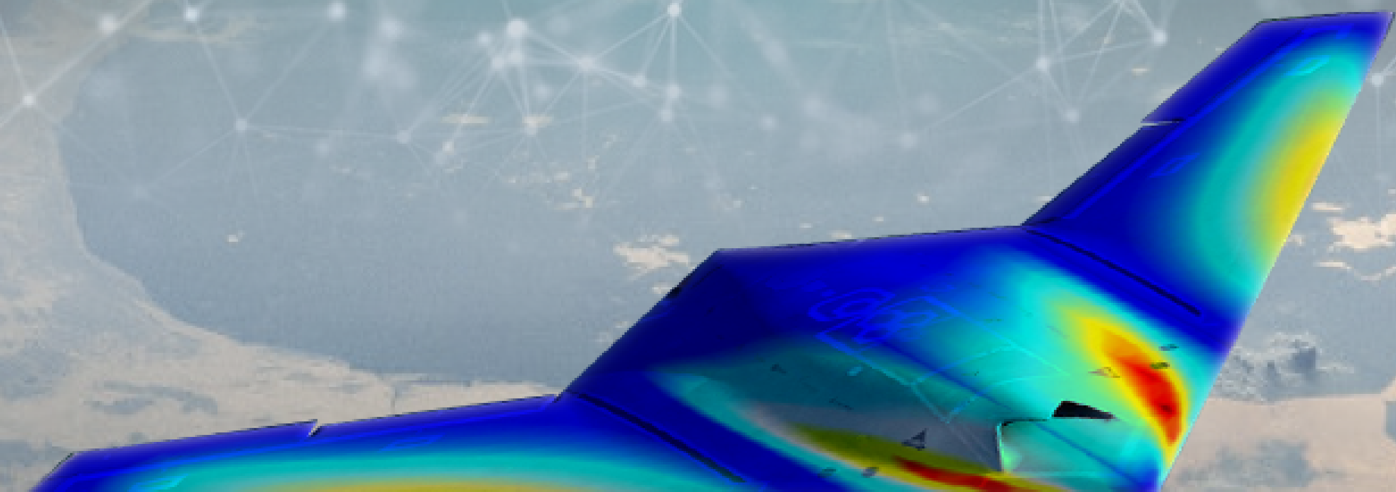
Development of a CFD data-driven surrogate model using the neural network approach for prediction of aircraft performance characteristics

S. J. L. B. Bourier

Technische Universiteit Delft



Development of a CFD data-driven surrogate model using the neural network approach for prediction of aircraft performance characteristics

by

S. J. L. B. Bourier

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on Friday November 26, 2021 at 14:00.

Student number:	4292502	
Project duration:	Juli, 2020 – November, 2021	
Supervisors:	Dr. S. J. Hulshoff	Delft University of Technology (TU Delft)
	Dr. ir. M. P. C. van Rooij	Royal Netherlands Aerospace Centre (NLR)
Thesis committee:	Dr. ir. G. la Rocca,	Delft University of Technology (TU Delft)
	Dr. S. J. Hulshoff,	Delft University of Technology (TU Delft)
	Dr. ir. C. C. de Visser,	Delft University of Technology (TU Delft)
	Dr. ir. M. P. C. van Rooij	Royal Netherlands Aerospace Centre (NLR)

Preface

With this thesis work my time as a student at the faculty of Aerospace Engineering at Delft University of Technology comes to an end. I am grateful for all the years that I have spent at this beloved faculty with wonderful experiences, ranging from meeting new friends to going abroad to work for one of the biggest engine companies, and I am excited to continue this journey as an engineer in the aerospace industry, full with challenges that needs to be tackled to make the aerospace industry more sustainable and even safer than it is nowadays.

As a more rapid and fast-paced industry is necessary to develop, improve but also to maintain current technologies within the aviation sector, I hope that this work will contribute and inspire others to make this a reality.

This thesis has been the greatest challenge I have experienced so far and I would like to express my gratitude towards my supervisors Michel van Rooij and Steven Hulshoff. I am grateful for the trust, support and all the knowledge you provided me throughout this journey and without you this work would not have been possible. Furthermore, I would like to thank the Royal Netherlands Aerospace Centre (NLR) for giving me the opportunity to work at a research company and giving me insights in current developments within the aerospace industry. It has been a wonderful time and I hope I can once return as employee or client to work with you.

The journey of finishing this work has been hard and certain circumstances definitely did not improve this environment, whereas the corona pandemic is just one example. However, I was always able to rely close on friends and housemates and for this I would like to express my gratitude. You have been of great support throughout my time as a student and I am looking forward to the times that will come. Furthermore, I would like to thank my family and in particular, my parents and my sister. You always have been there for me and your love and support is the accomplishment of many of my achievements. Finally, I would like to thank my girlfriend. In the last couple of months you have always been of great support and your motivation inspired me to continue on working and make me perform at its best. You make me happy and I am very grateful for every day we spend together. With this closure, I am looking forward to the many challenges that will cross my path and I am certain that with the gratitude of my friends and family I will always be able to strive for the best.

S. J. L. B. Bourier
Delft, November 2021

Abstract

Evaluation of aircraft performance for design, certification, and maintenance purposes requires aerodynamic knowledge for the entire flight envelope of an aircraft. Simplified models that relate geometric properties and flight conditions of an aircraft to its aerodynamic properties are simply not sufficient anymore, as non-linear aerodynamic phenomena, e.g., vortex development and flow separation, can drastically influence the performance of an aircraft. Therefore, high-fidelity simulations and wind tunnel experiments are necessary to assess the performance of an aircraft sufficiently. Although the development of new technologies in the last centuries allowed the computational time of the high-fidelity simulations to be decreased significantly, the computational expense still remains large. New techniques that are used to simulate the load cases of an aircraft and evaluate its aerodynamic properties are therefore aimed to provide an accurate representation of the high-fidelity simulations while the numerical complexity and hence the computational cost of the model is reduced. These techniques can be referred to as Reduced-Order Models (ROM).

Previous studies, conducted at the Royal Netherlands Aerospace Centre (NLR) as part of a collaborative research task group within the Science and Technology Organization (STO) of the North Atlantic Treaty Organisation (NATO), investigated the development of different ROMs that should be able to accurately predict the surface pressure distribution of the MULti-DIsciplinary CONfiguration (MULDICON) Unmanned Combat Air Vehicle (UCAV). In particular, a ROM based on the neural network approach shows promising results but suffers from drawbacks such as inaccurate prediction of the surface pressure near the wing tip region and the exclusion of time-history effects. This thesis report will serve as a continuation of previous research done on the reduced-order modeling approach for the prediction of the surface pressure coefficients of the UCAV configuration and will provide a baseline for future research. The research objective for this thesis report is therefore to develop a CFD data-driven Reduced-Order Model (ROM) based on the neural network approach that is able to predict the surface pressure and integral aerodynamic load coefficients of the UCAV MULDICON design and is able to capture the transient effects of the flow field for different flight maneuvers with the capability to reach the same level of accuracy as high-fidelity tools.

In this thesis, a ROM is developed that makes use of a reduced-order basis that is constructed using Proper Orthogonal Decomposition (POD). The POD method decomposes the high-fidelity samples used to train the ROM into spatial POD modes that will be used for the reduced-order basis, that are ranked in order of contribution towards the total kinetic energy (TKE). A reduced-order basis with only 10 spatial POD modes already captures 99 % of the TKE, which includes the most dominant flow properties and can be used to project the full-order solution with an accuracy that is in the same orders of magnitude while the data that is used is minimized. A Recurrent Neural Network (RNN) architecture is used as a surrogate model to determine the input-output relationship of the model. For this thesis, a Long Short-Term Memory (LSTM) architecture is used, which is widely used for the prediction of time-sequential data. The ROM that is constructed in this thesis is a combination between the POD method and the LSTM architecture, which can be referred to as the POD-LSTM model.

The performance of the POD-LSTM model has been evaluated and is split up into two different phases: the offline and online stages. The offline stage of the ROM is the stage where the reduced-order basis is constructed and the high-fidelity samples that are used for training are evaluated. From the offline stage, it can be concluded that the high-fidelity samples provide good coverage of the regressor space and most of the test samples have the same projection error as the high-fidelity training samples that are used for the construction of the reduced-order basis. The POD-LSTM model has been trained for varying model parameters and it was shown that the number of LSTM units has the largest influence on the computational training time. In the online stage, the performance of the prediction of the surface pressure coefficients using test samples that are gathered from steady and unsteady harmonic pitch and plunge oscillations are evaluated. The results for the steady simulations show that the POD-LSTM model is able to accurately predict the axial and normal force coefficient, while inaccuracy is shown for the pitching moment coefficient. The main reason for the inaccurate prediction of the pitching moment coefficient is the inaccurate prediction of the surface pressure coefficient near the wing tip region. Both the harmonic pitch and plunge oscillations show similar results, whereas the normal and axial force coefficients are predicted with good accuracy but the pitching moment coefficient is inaccurate due to an error in the surface pressure prediction near the wing tip region. The performance of the POD-LSTM model in terms of computational cost has been evaluated and was proven to be faster than previous studies, whereas the number of LSTM layers and LSTM units has the largest influence on the computational performance.

Overall, the POD-LSTM model has provided solutions to improve on previous studies for the surface pressure coefficient prediction of the UCAV configuration. However, it must be noted that this thesis report serves as a baseline for future research and the full capabilities of the POD-LSTM model still need to be exploited to provide a fair comparison with previous studies. Different training maneuvers, increase in training time and different model architectures could influence the performance of the POD-LSTM model and therefore should be incorporated in future studies to provide improvements to the proposed ROM.

Contents

List of Symbols

Roman Symbols

$\mathbf{a}$	Time coefficients	[-]
$\boldsymbol{a_0}$	Nominal angle of attack value	[deg]
$\mathbf{A}$	Angle of attack amplitude	[deg]
	Time coefficient matrix	[-]
$\mathbf{b}$	Bias	[-]
	Wing span	[m]
$\mathbf{C}$	Cost function	[-]
	Covariance matrix	[-]
$\boldsymbol{c_A}$	3-D axial force coefficient	[-]
$\boldsymbol{c_M}$	3-D pitching moment coefficient	[-]
$\boldsymbol{c_N}$	3-D normal force coefficient	[-]
$\boldsymbol{c_P}$	3-D surface pressure coefficient	[-]
$\boldsymbol{f}$	Forget gate	[-]
$\boldsymbol{f}$	Frequency	[Hz]
$\mathbf{h}$	Hidden state	[-]
$\mathbf{i}$	Input gate	[-]
$\boldsymbol{l_{ref}}$	Wing reference length	[m]
$\mathbf{M}$	Moment reference point	[m]
$\boldsymbol{M_\infty}$	Mach number	[-]
$\mathbf{o}$	Output gate	[-]
$\boldsymbol{P_\infty}$	Ambient pressure	[Pa]
$\boldsymbol{q}$	Pitch rate	[deg/s]
$\mathbf{S}$	Wing surface area	[m^2]
$\mathbf{t}$	Time	[s]
$\mathbf{T}$	Period	[s]
$\boldsymbol{T_\infty}$	Ambient temperature	[K]
$\mathbf{u}$	Velocity in x-direction	[m/s]
$\mathbf{U}$	Velocity vector field matrix	[-]
$\boldsymbol{V_\infty}$	Flight velocity	[m/s]
$\mathbf{w}$	Weights	[-]
$\boldsymbol{\bar{x}}$	Normalized coordinates in x-direction	[-]
$\boldsymbol{\bar{y}}$	Normalized coordinates in y-direction	[-]
$\boldsymbol{z}$	Neuron input	[-]

Greek Symbols

$\boldsymbol{\alpha}$	Angle of attack	[deg]
$\boldsymbol{\delta}$	Neuron error	[-]
$\boldsymbol{\epsilon}$	Error in terms of MSE	[-]
$\boldsymbol{\eta}$	Learning rate	[-]
$\boldsymbol{\theta}$	Pitch angle	[deg]
$\boldsymbol{\lambda}$	Regularization parameter	[-]
$\boldsymbol{\rho_\infty}$	Ambient density	[kg/m^3]
$\boldsymbol{\sigma}$	Sigmoid function	[-]
$\boldsymbol{\phi}$	Phase shift	[deg]
	Spatial POD mode	[-]

List of Figures

List of Tables

Acronyms

ANN Artificial Neural Network. xi, 10–19

BPTT Backpropagation Through Time. 18, 19

BWB Blended-Wing-Body. 23

CFD Computational Fluid Dynamics. 3, 5, 9, 27, 28, 33, 64, 66, 71, 75, 81, 86, 99, 107

CNN Convolutional Neural Network. 2, 3, 11, 17, 64

DMD Dynamic Mode Decomposition. 6

DoE Design of Experiments. 9, 27, 28

FFNN Feed-Forward Neural Network. 11, 17, 18

GRU Gated Recurrent Unit. 21

LSTM Long Short-Term Memory. v–vii, xi, 3, 4, 19–21, 23, 26, 39–43, 47, 56–58, 63–66, 71, 72, 75, 78, 80, 81, 83, 92, 93, 97–101, 103, 104

MCP McCulloch-Pitts. 10, 11

MLP Multi-Layer Perceptron. 11, 12, 15, 18

MOR Model Order Reduction. 25, 34, 43

MULDICON MULti-DIsciplinary CONfiguration. v, xi, xv, 23–29, 32, 33, 35, 39, 42, 100, 101

NATO North Atlantic Treaty Organisation. 1, 23

NLR Royal Netherlands Aerospace Centre. xi, 1, 2, 23, 25, 33, 57, 64

NMSE Normalized Mean Squared Error. 70, 71, 78, 81, 89, 110

POD Proper Orthogonal Decomposition. v–vii, ix, 3, 4, 6–8, 21, 23, 25, 26, 34–36, 39–44, 47, 50, 53–56, 63–66, 71, 72, 75, 78–81, 83, 89, 92, 97–101, 103, 104, 110

RANS Reynolds-Averaged Navier–Stokes. xi, 5, 33, 34

RNN Recurrent Neural Network. xi, 3, 11, 17–21, 39, 99

ROM Reduced-Order Model. v, vi, xv, 1, 3, 5, 6, 23–28, 32, 34, 35, 39, 41–43, 47, 79, 90, 111

RPF Relative Peak Factor. 29

SLP Single-Layer Perceptron. 11

STO Science and Technology Organisation. 1, 23

TBPTT Truncated Backpropagation Through Time. 19

TKE Total Kinetic Energy. xi, 3, 6, 8, 34–36, 47, 78

UCAV Unmanned Combat Air Vehicle. v, vi, 1–5, 17, 23, 28, 98–101

1 Introduction

The process of aircraft design, certification, and load predictions for maintenance purposes require a lot of aerodynamic knowledge that usually is obtained from data that is gathered for the entire flight envelope. This aerodynamic knowledge that is required to evaluate the loads, the performance, and the handling qualities of an aircraft can vary from integral loads of the aircraft up to distributed data such as pressure and shear stress distributions. Currently, this aerodynamic analysis of aircraft is performed using high-fidelity tools such as Computational Fluid Dynamics (CFD) and windtunnel experiments. Although these approaches are very promising, they are still considered to be computationally expensive and hence very time-consuming. New techniques that are developed to simulate the load cases of an aircraft are therefore aimed to provide an accurate representation of the high-fidelity tools while the numerical complexity and the computational training time are reduced. A typical approach to do this is to simplify the physics of the model, e.g., the use of potential flow equations for the determination of the load cases [22]. However, the issue with these simplified models is that significant effects, e.g., transonic flow effects and stall, are neglected and therefore not all features of the flow are incorporated. Another technique that can be used to reduce the numerical complexity is referred to as reduced-order modeling, which aims to provide an accuracy that is comparable to the already existing high-fidelity tools while remaining its complex features. Implementations of such techniques as the Reduced Order Model (ROM) can be used next to high-fidelity tools to minimize the amount of high-fidelity data that is necessary and hence to decrease the computational training time and prediction time during aerodynamic analysis of an aircraft. This thesis report will go into the research details of the construction of a ROM with the aim to accurately predict the surface pressure distribution of an unconventional aircraft design. This chapter serves as an introduction to the research studies that have been performed in collaboration with the Royal Netherlands Aerospace Centre (NLR). First, the project motivation will be discussed. Second, the problem description of the thesis will be described. Third, the research objective and research questions that will be handled during this report will be discussed. Finally, the report outline of this thesis will be presented.

1.1. Project motivation

In 2018, a research task group was formed within the Science and Technology Organisation (STO) was formed by the North Atlantic Treaty Organisation (NATO). The purpose of this task group AVT-251 on multi-disciplinary design and performance assessment of effective, agile NATO air vehicles, was the application of advanced numerical tools and collaborative approaches to the design of an Unmanned Combat Air Vehicle (UCAV) for a specific bomber mission [4]. The UCAV design can be categorized as an unconventional aircraft type. Usually, high-fidelity tools or experiments are necessary in order to evaluate the performance of such designs as empirical models based on existing aircraft are not applicable or do not cover the design space well. Therefore, the implementation of ROMs could enhance the early stages of the design process, as they will reduce the number of high-fidelity simulations necessary. Furthermore, ROMs can be used for load prediction for different states of flight. Additionally, they can be coupled to flight dynamics models and structural models to determine aircraft performance and establish a basis for predictive maintenance. As part of the STO task group, the Royal Netherlands Aerospace Centre (NLR) has been contributing to the development of ROMs for aerodynamic load prediction for the UCAV design, with a particular interest in the indicial response approach and the neural network approach ([28],[18]). The latter shows potential but suffers from an inaccurate prediction of the surface pressure near the wingtips, causing inaccurate results for the pitching moment coefficient. This research study, in collaboration with the NLR, is done as a continuation of the research of ROMs for aerodynamic load prediction on the UCAV design and will provide a further investigation into the neural network approach.

1.2. Problem description

Previous studies have shown that the neural network approach can be used as a good surrogate model in the construction of Reduced-Order Models. In particular, the neural network approach that has been introduced by Papp [18] and was based on the Convolutional Neural Network (CNN) architecture has shown good promise for the aerodynamic load prediction of the UCAV design using state variables only, e.g., the angle of attack α and the pitch rate q. However, a few drawbacks using this approach have been identified:

- The CNN architecture makes use of an input tensor that is based on the geometrical coordinates of the UCAV design. Although the number of data points in the domain is reduced, the size of the input tensor still causes the CNN architecture to be very complex, which can lead to an increase in computational training time and computational prediction time. A reduction of the input tensor size for the CNN architecture could lead to a decrease in the computational training and computational prediction time of the model.
- Currently, the CNN architecture is only able to predict the surface pressure distribution using state variables of one instantaneous time step, i.e., the CNN architecture is not able to make use of time-history effects that can influence the flow for the prediction of its consecutive time step. The incorporation of multiple time steps in the CNN architecture could lead to an increase of the model prediction accuracy of the surrogate model due to coverage of transient effects within the flow. The inclusion of time-history effects is particularly important for unsteady simulations.
- The CNN architecture has no imposed boundary conditions for the prediction of the surface pressure coefficients. As the CNN architecture directly relates the geometric coordinates of the UCAV design and current state variables with the surface pressure prediction, unrealistic results could be produced. The introduction of boundary conditions within the CNN architecture or the input tensor could lead to an enhancement of the model prediction accuracy.
- The prediction of the surface pressure distribution near the wing tip region is inaccurate for higher angles of attack. It has been shown that only a small error in the prediction of the surface pressure near the wing tip can lead to large errors in the prediction of the pitching moment coefficient. The introduction of a weighted relationship towards the tip region could put more emphasis on the accurate prediction of this region, leading to an increase in the model prediction accuracy.
- The CNN architecture that is introduced is based on state-of-the-art applications that were shown in literature and previous studies. Evaluation of model parameter sensitivity is not included, while model parameters could have a large influence on the model performance in terms of accuracy and cost. A sensitivity analysis of the model parameters can provide the user insights into the model performance and hence determine which model parameters are suitable for the application of the model.

To summarize, the CNN architecture, as discussed in the work of Papp [18], is a complex surrogate model that makes use of an input tensor that is based on geometric coordinates and thus is computationally expensive. Although the model is able to predict the surface pressure distribution of most cases well, the model is only capable of predicting the surface pressure distribution using one instantaneous time step. Additionally, the inaccurate prediction of the surface pressure near the wing tip region of the UCAV design leads to large errors in the prediction of the pitching moment coefficient.

A potential solution to improve the previous model of Papp and to overcome its drawbacks is twofold:

1. The introduction of boundary conditions could enhance the model prediction accuracy. However, this must be done without increasing the complexity of the model.

2. The use of multiple time steps for prediction, i.e., the inclusion of time-history effects, could improve the model prediction accuracy.

A literature study has been performed to identify possible reduced-order modeling techniques to incorporate the suggested solutions as stated above. A more elaborate discussion of this literature can be found in Chapter 2 and Chapter 3. First of all, in order to impose boundary conditions in the model and to make sure that the predictions of the surrogate model are realistic, it was found that one can make use of a modal decomposition method that is referred to as Proper Orthogonal Decomposition (POD). The POD method uses high-fidelity snapshots that are typically instantaneous full-order solutions obtained from CFD data. This method will decompose the full-order solution into spatial modes with its corresponding time coefficients that are ranked in relevance towards the contribution of the Total Kinetic Energy (TKE) of the flow, i.e., the first spatial mode that is found has a higher TKE than the second spatial mode. From the obtained spatial modes, a reduced-order basis can be constructed on which the full-order model is projected via multiplication with its time coefficients. Usually, only a few spatial modes are necessary in order to obtain 99 % of the TKE [29], i.e., a reduced-order basis that is constructed from only a few spatial modes can cover the most dominant flow properties. One of the benefits of using this approach is that it will reduce the amount of data significantly. Furthermore, as the spatial modes only need to be constructed once, they are not used in the prediction of the surrogate model. This means that with the use of the POD method, one only is interested in the prediction of the time coefficients of the reduced-order basis. As the spatial modes cover the most dominant flow properties, boundary conditions are imposed automatically if the time coefficients are predicted well, hence this will reduce the inaccuracy of the surface pressure prediction of the model. As only the prediction of time coefficients of the reduced-order basis is necessary to approximate the full-order solution, complex neural network architectures, e.g., the CNN, become redundant. As the inclusion of multiple time steps in the prediction of the model to incorporate time-history effects is of importance, the Recurrent Neural Network (RNN) architecture is of interest. The RNN architecture, as the name already suggests, is able to handle time-sequential data. Literature has shown that within the RNN architectures, the Long Short-Term Memory (LSTM) model performs best for predicting time-sequential data and is used for a wide range of applications. One of the benefits of the LSTM model is that it will only memorize relevant information and will disregard all the information that is irrelevant for the prediction of the time-sequential data. This working principle makes the LSTM model a proper surrogate model that is able to capture general trends within the data.

The reduced-order modeling approach that is proposed in this thesis work will be a combination between the Proper Orthogonal Decomposition method and the LSTM architecture and will be referred to as the POD-LSTM model. It must be noted that the purpose of this thesis work is to identify the strengths and weaknesses of the new proposed model and therefore will only serve as a baseline for the construction of the POD-LSTM model. If the work is deemed to be successful, a more elaborate investigation is necessary in order to provide a good comparison between the current and previous studies. Furthermore, as the high-fidelity CFD data that is used for this work has been obtained from previous studies, the generation of the high-fidelity CFD data is outside of the scope of this project.

1.3. Research objective and questions

The proposed reduced-order modeling approach that will be discussed in this thesis aims to provide solutions for the drawbacks that were found in previous studies. As this work serves as a baseline for further research, this thesis work aims to fulfill the following formulated research objective:

"To develop a CFD data-driven Reduced-Order Model (ROM) based on the neural network approach that is able to predict the surface pressure and integral aerodynamic load coefficients of the UCAV MULDICON design and is able to capture the transient effects of the flow field for different flight maneuvers with the capability to reach the same level of accuracy as high-fidelity tools."

To achieve this research objective, the following research question is proposed:

Is the reduced-order modeling approach that is based on the POD-LSTM model a suitable approach to predict the surface pressure and integral aerodynamic load coefficients of the UCAV MULDICON design, while the accuracy of the model is in the same orders of magnitude as the high-fidelity simulations and

computational training time and computational prediction time of the model are low?

In order to provide answers to the main research question, additional research questions are formulated that will provide more insights into the proposed research for this thesis work:

1. What are the limitations of using a reduced-order basis? How many spatial POD modes are necessary in order to construct a reduced-order basis for a given accuracy?
2. What is the effect of varying the spatial POD modes within the reduced-order basis?
3. Do the high-fidelity samples cover the regressor space of the proposed POD-LSTM model sufficiently to predict samples that are within the bounds of coverage?
4. How can the errors of the POD-LSTM model predictions be evaluated in a systematic way? How can the differences between the errors of the reduced-order basis and the surrogate model be identified?
5. What is the effect of varying the model parameters on the computational expense of the model?

1.4. Report outline

This thesis report will investigate possibilities to improve the previously proposed surrogate model that is based on the neural network approach by Papp [18] and will provide the answers to the research questions as stated in this introduction. In Chapter 2 and Chapter 3, the theoretical background for this thesis work is discussed, which serves as fundamental knowledge for the proposed POD-LSTM model. Then, Chapter 4 will describe the methodologies that are used to construct the proposed POD-LSTM model. In Chapter 5, the results of the simulations for the prediction of the surface pressure coefficients of the UCAV design are shown. Finally, in Chapter 6, the conclusions of this thesis work are discussed and recommendations for further research are proposed.

2

Literature Review: An introduction to reduced-order modeling techniques

Prior to this thesis work, a literature study has been performed in order to identify potential solutions that can be used to improve previous studies on the prediction of the surface pressure distribution of the UCAV design. As the neural network surrogate modeling approach shows potential, the scope of the literature review has been limited to state-of-the-art applications that are using the neural network approach. This chapter will provide the reader a basic understanding of the neural network approach and will serve as an introduction towards reduced-order modeling.

2.1. Introduction to Reduced-Order Modeling

Currently, highly expensive simulations such as CFD simulations and wind tunnel experiments are necessary in order to evaluate the performance of aircraft for design, certification, and maintenance purposes. Although current developments in RANS CFD simulations and the rapidly evolving developments in hardware are decreasing the computational cost, they are still considered to be too expensive for real-time evaluation. Hence, new techniques that are developed to simulate the load cases of an aircraft are therefore aimed to provide an accurate representation of the high-fidelity tools while the numerical complexity and the computational cost of it are reduced. Models that are able to capture the main features of the flow, while numerical complexity is reduced and high accuracy with respect to high fidelity models is achieved, are referred to as Reduced-Order Models (ROM).

Two different reduced-order modeling approaches exist: the intrusive and the non-intrusive approach. One of the major drawbacks of the intrusive reduced-order modeling approach is that it requires access to the full model and solvers, which are typically unavailable when working with fluid flows, where the details of governing equations are not known. The non-intrusive reduced-order modeling approach is based on snapshots from high-fidelity tools, e.g., numerical approximations obtained from Computational Fluid Dynamics (CFD) or measurements via wind tunnel experiments. The benefit of non-intrusive reduced-order models is that it does not require any knowledge about the underlying equations but is still able to approximate the dominant properties of the flow field. As this research will be a continuation of the previous proposed research, one is only interested in the non-intrusive reduced-order modeling approach. The non-intrusive approach works in a black-box manner, i.e., the approach is entirely data-driven and thus enables great flexibility for practical problems [31]. Reduced-order models work with an offline-online strategy, where the offline stage is used to determine a reduced-order basis and construct the surrogate model and the online stage is used to predict new solutions for a given input with the given surrogate model. An example of the offline-online strategy is shown in Figure 2.1.

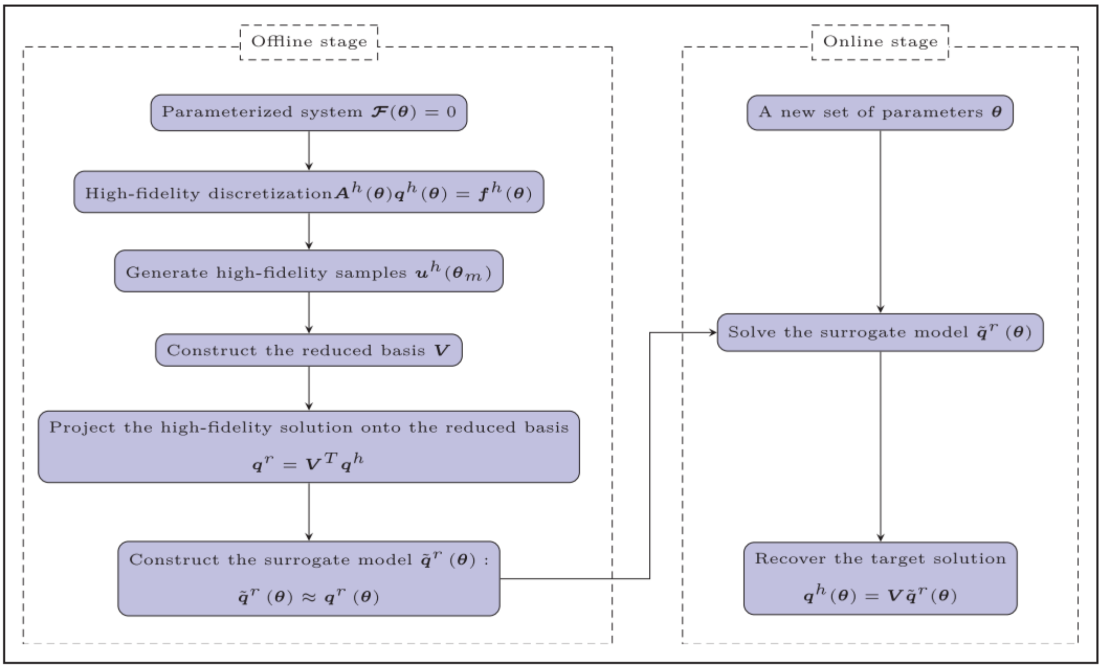


Figure 2.1: A flow chart of the non-intrusive ROM [31].

2.1.1. Dimensionality Reduction for non-intrusive Reduced-Order Models: The Proper Orthogonal Decomposition

One of the benefits of the (non-intrusive) reduced-order modeling approach is that only a limited amount of data is used to approximate the full-order solution by means of dimensionality reduction methods. Dimensionality reduction can be done in various ways, however, the modal decomposition method is the most used in current literature. Modal decomposition can be used to approximate the full-order solution by projecting the full-order solution on a reduced-order basis. Two main modal decomposition methods exist: The Proper Orthogonal Decomposition (POD) method and the Dynamic Mode Decomposition (DMD) method. Although both methods have their own benefits, the POD method is deemed to be optimal [13] and shows the most potential for the research objective of this work. The POD method and its fundamentals will be discussed in this section.

Proper Orthogonal Decomposition

One of the techniques that are often applied for dimensionality reduction in reduced-order modeling approaches is called the Proper Orthogonal Decomposition (POD) method. The POD method is known under various names, e.g., the Principal Component Analysis (PCA) or the Karhunen-Loève Transform, depending on the field of application. The basic concept of the POD method in the field of fluid dynamics is to decompose a given vector field and determine its orthogonal basis, known as spatial functions, that each capture a portion of the Total Kinetic Energy (TKE) of the flow. It must be noted that the deterministic spatial functions found with POD are ordered by relevant significance, i.e., the first spatial mode found has a higher TKE than the second spatial mode. Dimensionality reduction using the POD is done by approximating the entire flow field with a limited number of spatial modes that will represent the most dominant behavior of the flow while maintaining accuracy. As one is mainly interested in approaches that can be used for non-intrusive ROM, the section below will only elaborate on POD methods that can be used with data obtained from high-fidelity simulations.

Direct Method

The POD method was introduced for the very first time by Lumley in 1967 in the field of fluid dynamics [13]. This POD method is more commonly known as the POD direct method. The direct method as described in this section is based on the given example by Weiss [29], considering a n-dimensional flow with the velocity vector $U = (u, v, w)$ and the spatial coordinates $\bar{x} = (x, y, z)$ for a given time t. It must be noted that POD also is applicable for different variables, e.g., pressure and vorticity, and is not just limited to the velocity vector as described in this section.

As one is mainly interested in the flow dynamics, one can reconstruct the velocity field by subtracting the time-averaged velocity field from each individual vector field at a given instant t to obtain the velocity fluctuation at a given point $\bar{x}$. If one then re-orders each time instant fluctuating velocity field and adds all of them into one matrix, one obtains the matrix U, known as the snapshot matrix. For simplicity, only the longitudinal velocity u was considered in this example. If one is dealing with more velocity components, e.g., v, w, one can simply extend the columns of snapshot matrix U with the given velocity components. It must be noted that the re-ordering of matrix U can be done in several ways, as long as the same procedure is used for the reversed process. An example for the snapshot matrix U with m given time instants for one component is given as:

$$U = \begin{pmatrix} u_{11} & ... & u_{1n} \\ u_{21} & ... & u_{2n} \\ . & . & . \\ . & . & . \\ u_{n1} & ... & u_{mn} \end{pmatrix} = \begin{pmatrix} u'(x_1, y_1, t_1) & ... & u'(x_n, y_n, t_1) \\ u'(x_1, y_1, t_2) & ... & u'(x_n, y_n, t_2) \\ . & . & . \\ . & . & . \\ u'(x_1, y_1, t_m) & ... & u'(x_n, y_n, t_m) \end{pmatrix} \tag{2.1}$$

The idea behind the POD direct method is to decompose the given velocity fields such that the vector field can be written as a set of deterministic spatial functions, or POD modes, ϕ_k, which are multiplied by a time coefficient $a_k(t)$ for a given time instant t. This is shown in Equation (2.2).

$$u'(x, t) = \sum_{k=1}^{\infty} a_k(t)\phi_k(x) \tag{2.2}$$

To determine if the vector field is correlated, thus to find deterministic spatial functions ϕ_k, it is necessary to determine the covariance matrix of the vector field U with m given time instants:

$$C = \frac{1}{m-1} U^T U \tag{2.3}$$

It must be noted that the determination of the covariance matrix, C, is resulting in a square matrix of (NxN) dimensions according to N-given data points in the grid. By evaluating the eigenvalues and eigenvectors of the covariance matrix C and ordering the eigenvalues from the largest eigenvalue to the smallest with their corresponding eigenvectors, one can obtain the proper orthogonal modes ϕ from the re-arranged eigenvector matrix.

$$\phi = \begin{pmatrix} \phi_{11} & ... & \phi_{1n} \\ \phi_{21} & ... & \phi_{2n} \\ . & . & . \\ . & . & . \\ \phi_{n1} & ... & \phi_{nm} \end{pmatrix} \tag{2.4}$$

Finally, one can obtain the variance of the vector field U on the orthogonal basis by projecting the vector field U onto each individual eigenvector of the proper orthogonal modes ϕ:

$$A = U\phi = \begin{pmatrix} a_{11} & ... & a_{1n} \\ a_{21} & ... & a_{2n} \\ . & . & . \\ . & . & . \\ a_{n1} & ... & a_{nm} \end{pmatrix} = \begin{pmatrix} u_{11} & ... & u_{1n} \\ u_{21} & ... & u_{2n} \\ . & . & . \\ . & . & . \\ u_{n1} & ... & u_{mn} \end{pmatrix} \begin{pmatrix} \phi_{11} & ... & \phi_{1n} \\ \phi_{21} & ... & \phi_{2n} \\ . & . & . \\ . & . & . \\ \phi_{n1} & ... & \phi_{nm} \end{pmatrix} \tag{2.5}$$

The columns of the obtained matrix **A** are known as the time coefficients for each of the corresponding modes for the given time instants. To visualize the modes as presented above, one can follow the exact inverse procedure to obtain a scalar field, referred to as the spatial modes of the flow field. An example of these spatial modes with the corresponding time coefficients can be seen in Figure 2.2.

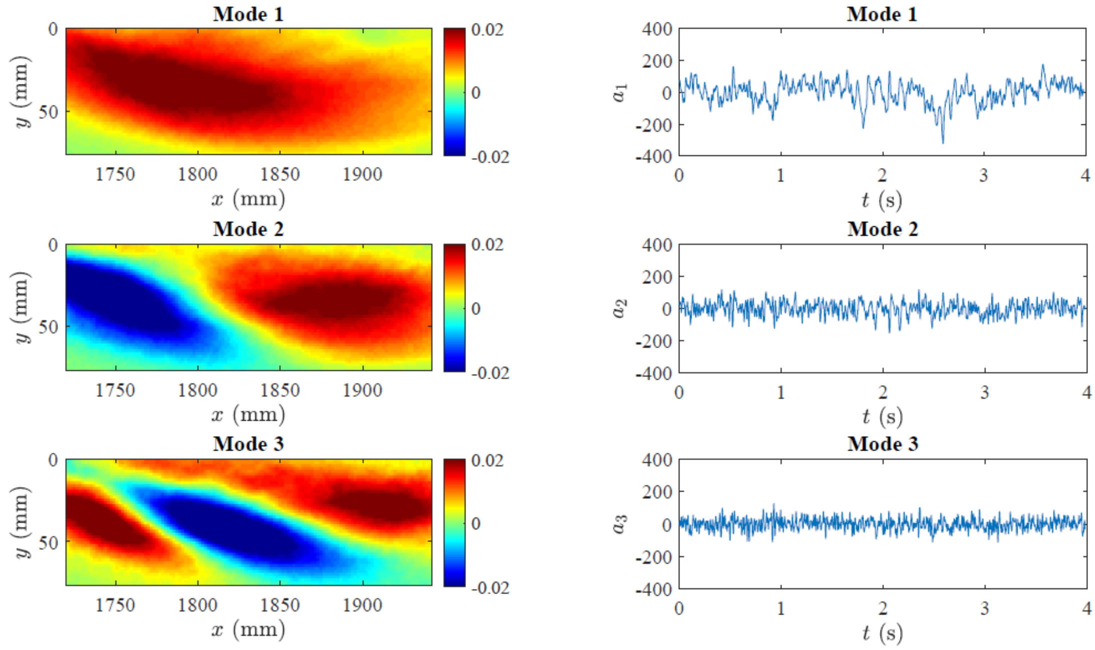


Figure 2.2: POD spatial mode representation for first three modes of turbulent separation-bubble flow [29].

The benefit of using the POD method is that the spatial modes are ranked in the relevance of the TKE of the flow, i.e., the first spatial mode has a higher contribution to the TKE than the second spatial mode. Moreover, the energy that is covered in each mode is expected to decay rapidly with increasing ranks. Therefore, using only the first few modes, i.e., the reduced basis, captures a large portion of the overall energy and in most of the time is sufficient to serve as a good approximation to the original high-fidelity simulations [31]. One of the properties of the POD method is that the dominant flow properties of the flow can be extracted from the full-order solution with only a limited amount of spatial POD modes necessary. Therefore, the POD method not only serves as a dimensionality reduction method, which decreases the computational cost of the model but also bounds the approximation of the full-order solution, i.e.., the reduced-order basis will only provide values that are deemed realistic. An example of the full-order solution approximation using a reduced-order basis constructed using the POD method is shown in Figure 2.3, where only two spatial POD modes are used to approximate the instantaneous full-order solution.

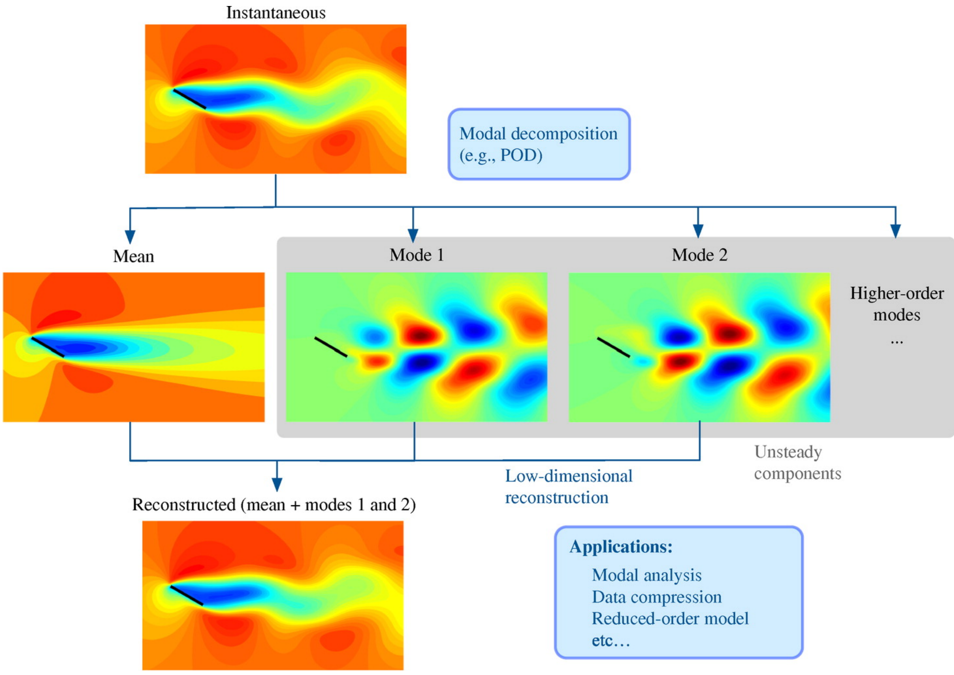


Figure 2.3: Example of dimensionality reduction with POD [26].

2.2. Surrogate modeling techniques for non-intrusive Reduced-Order Models

One of the benefits of a non-intrusive reduced-order model is that the model is data-driven, hence understanding the underlying fundamentals is not necessary. However, as there are no governing equations available an approximation model is necessary that is able to map the relationship between the input and output of the full-order solution. Such approximation models are referred to as surrogate models. As the exact working between the input-output relationship is unknown, it is said that these surrogate models work in a black-box manner. Usually, the model is constructed based on modeling the response to a limited number of high-fidelity simulations. An example of the construction of a surrogate model is shown in Figure 2.4.

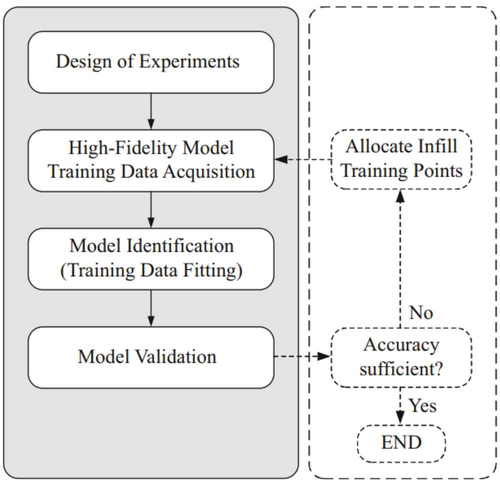


Figure 2.4: A surrogate model construction flowchart [11]

As shown from the figure, the surrogate model is constructed using high-fidelity data, e.g., CFD simulations, that are obtained from a number of preselected inputs. These preselected inputs usually are determined using Design of Experiments (DoE), which tries to minimize the number of data points necessary while the full regressor space is covered.

There are various data-driven surrogate models available such as Response Surface Modeling, Kriging, the use of Radial Basis Functions, or the neural network approach. As one is mainly interested in the improvement of previous studies that are based on a neural network surrogate model, the scope of this work is limited to the neural network approach only.

2.2.1. Introduction to machine learning: the neural network approach

This section will provide a basic understanding of the state-of-the-art ann applications that are currently used in the field of fluid dynamics. The fundamentals of ANNs will be discussed to give the reader sufficient information behind the theory of the black-box manner behavior of the neural network approach.

Basic principles of ANNs

The modeling of ANNs is inspired by how the biological brain processes information. The first understanding of how the biological brain does this has been investigated by Warren McCulloch and Walter Pitts [20] with their concept of a simplified brain cell, also referred to as neuron, described with the McCulloch-Pitts (MCP) neuron model. The human brain contains more than 86 billion of these neurons [2] that are all involved in the process of gathering and transmitting electrical signals and hence processing information. A schematic overview of a single neuron, as described by McCulloch and Pitts [20], is shown in Figure 2.5. As seen from the figure, the neuron receives a signal via their dendrites, processes all the input signals in the cell body, and transmits its output via the axons, which are connected to other neurons. It was described that the cell body of the MCP neuron model only transmits an output signal if the accumulated input signal exceeds a certain threshold [20].

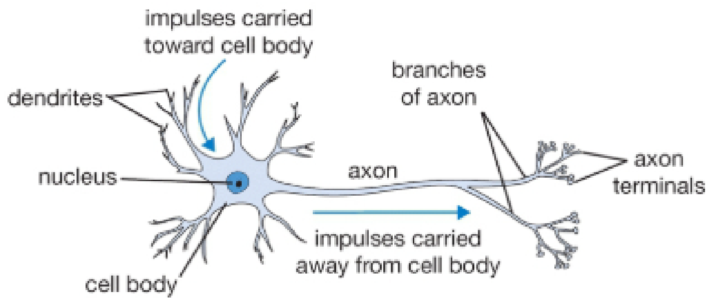


Figure 2.5: Schematic overview of the working of a single neuron [1].

The computational model that is accompanied with this MCP neuron model relates the synapses, the connection between different neurons, as weighted signals. The neuron only transmits a signal if the accumulated weighted input signals exceed a certain threshold according to its activation function, which will be elaborated on later in this section. A schematic overview of this computational model is shown in Figure 2.6.

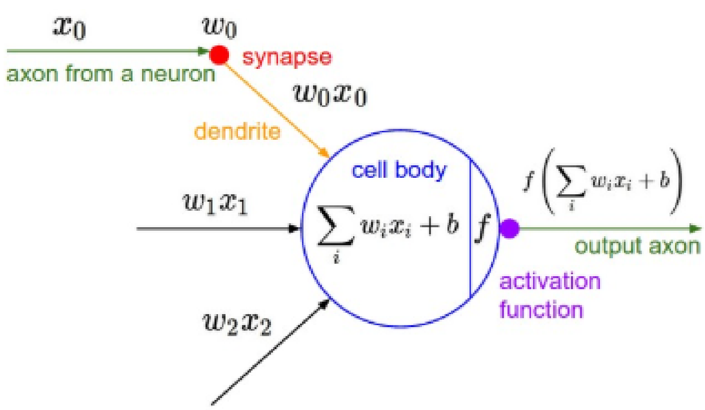


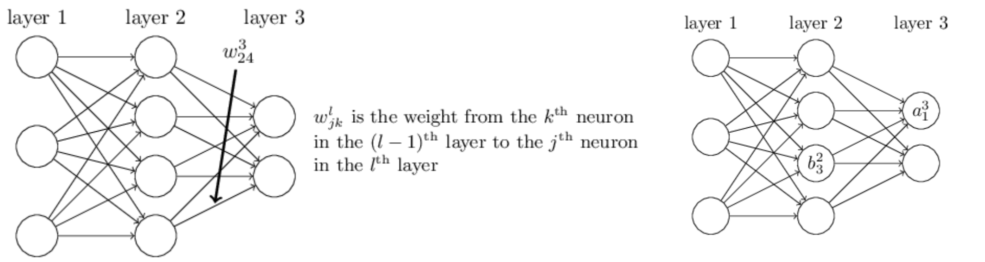
Figure 2.6: Schematic overview of the computational model of a single neuron [1].

As seen from Figure 2.6, the operation that is carried out by a single neuron can be derived as the output value, y, that is obtained from the activation function, f, evaluated at the summation of the weighted input signal, defined as the input, x, multiplied by its corresponding weight, w, plus its bias, b:

$$y_j = f(z_j) = f\Big(\sum_i w_i x_i + b\Big) \tag{2.6}$$

Based on the basic principles of such a MCP neuron model, Rosenblatt published the first concept of the perceptron learning rule [23]. This rule would predict whether or not a neuron would transmit a signal for a given input, by automatically learning the optimal weight coefficients from previously given input data. Both the MCP neuron model and the perceptron learning rule are the basis of how neural networks are currently modeled.

The basic architecture of an ANN consists out of multiple neurons stacked in a layer. The simplest form of an ANN exists as a Single-Layer Perceptron (SLP) network. This SLP network directly relates the input to the output. However, for more complex feature extraction, one can use Multi-Layer Perceptron (MLP) networks. These networks consist, as the name suggests, out of multiple layers. All the layers that exist between the input and the output of the ANN are called hidden layers. If the ANN contains more than one hidden layer, the ANN is defined as a deep artificial neural network [21]. However, it must be noted that the input layer of the neural network architecture does not count as a layer. The annotations that will be used for the weights, biases, and activations of the ANNs in this report are according to Figure 2.7. This figure shows that the weights that are used, w^l_{jk}, represent the weight of the connection of the k th neuron in the $(l-1)$ th layer to j th neuron in the l th layer. Furthermore, the bias and the activation, b^l_k and a^l_k respectively, represent the bias and the activation of the k th neuron in the l th layer.



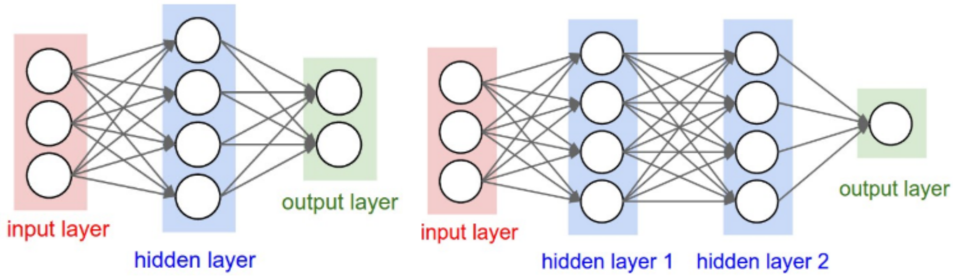
(a) Annotation for the weights of an ANN.

(b) Annotation for the activation and bias of an ANN.

Figure 2.7: Annotations for a given ANN [17].

Although the connection between the neurons in consecutive layers is linear, one can solve complex functions with the use of the activation function, f. The activation function is a function that translates the input of a given neuron to a given value according to its pre-defined function. Different functions exist, e.g., sigmoid or hyperbolic tangent functions. The result of the activation function will influence the behavior and ultimately the training of the ANN. The higher the output of the neuron, the more important that neuron is in the process of representing the behavior that is given. Furthermore, the bias is used in order to make sure that all the neurons are trained in an efficient manner, e.g., if the neuron has an output of zero, the activation function, dependent on the function given, will also output a value of zero, hence the neuron will not be trained.

The most common connection between layers is the fully-connected layer. In a fully-connected layer, the neurons between two adjacent layers are fully pairwise connected, but the neurons share no connections within each layer. An example of this architecture can be seen in Figure 2.8. Different kinds of connections exist within the ANN architecture, which on its own define the type of architecture. The most common architectures that are found in neural network modeling are the Feed-Forward Neural Network (FFNN), the Convolutional Neural Network (CNN) and the Recurrent Neural Network (RNN).



(a) A 2-layer MLP network. The input layer is not counted as a layer in the network.

(b) A 3-layer MLP network. The input layer is not counted as a layer in the network.

Figure 2.8: Schematic overview of fully-connected neural network architectures [1].

In order to assure that the ANN is capable of predicting the outcomes for new input data, the ANN needs to adjust the weights of the predefined neurons accordingly, as described by the perceptron learning rule [23]. The process at which ANN parameters are changed in order to increase the performance of the model is called training. By training the model, the ANN basically extracts features from given data and tries to alter the ANN parameters to obtain a better fit for the model. If the training is deemed to be successful, the ANN will be able to predict the outcome of new data, which is similar to the data that is used for the training of the ANN, with small errors.

Essentially, the training of an ANN is an optimization process where the error between the predicted and the desired outcome of the ANN is minimized using a defined cost or loss function. As one of many examples, such a cost function can be defined as the sum of squared errors, dependent on the given objective [21]:

$$C(w) = \frac{1}{2}\sum_i \left(y_i - f(z)_i\right)^2 \tag{2.7}$$

As one is interested in minimizing the cost function of the ANN, it is necessary to determine the effect of a variation of the ANN parameters and hence find the optimal solution of all the parameters to obtain the minimum value of the cost function. In other words, it is necessary to determine the gradient of the cost function with respect to all the parameters of the ANN in order to determine the most influencing parameters and enhance the minimization of the cost function by altering these parameters. One of the algorithms that are used to determine the minimum of the cost function and thus to train ANNs efficiently is the gradient descent [17].

Gradient descent

The gradient descent algorithm is an iterative method that determines the desired direction to reach the minimum of the cost function in the most efficient manner by evaluating the gradient of the cost function, ∇C. If one then chooses a small variation in parameters, i.e., the weights and biases for a given ANN, Δv, one can obtain the following expression [17]:

$$\Delta C \approx \nabla C \cdot \Delta v \tag{2.8}$$

One can define these changes of parameters using a predefined (positive) learning rate, η, to define the change vector of parameters as [17]:

$$\Delta v = -\eta \nabla C \tag{2.9}$$

Using both expressions, one can obtain a new relation for the difference in the cost function for the varying parameters:

$$\Delta C \approx -\eta \nabla C \cdot \nabla C = -\eta ||\nabla C||^2 \tag{2.10}$$

It is known that $||\nabla C||^2 \geq 0$, hence the variation of the cost function by altering the parameters is always negative or equal to zero. This means that the given cost function will always decrease or remain constant for a variation of the cost function parameters. Knowing this, one can use this relationship to determine the updated value of the cost function parameters:

$$v \rightarrow v' = v - \eta \nabla C \tag{2.11}$$

One will repeat this process until convergence is reached and the minimum of the cost function or the maximum amount of iterations, also known as epochs, is found. In essence, for ANNs the gradient descent algorithm is used to find the weights, w_k, and the biases, b_k, which will minimize the defined cost function to obtain accurate results for the given problem. For ANNs Equation (2.11) can therefore be rewritten as follows:

$$\begin{aligned} w^l_{jk} \rightarrow (w^l_{jk})' = w^l_{jk} - \eta \frac{\partial C}{\partial w^l_{jk}} \\ b^l_j \rightarrow (b^l_j)' = b^l_j - \eta \frac{\partial C}{\partial b^l_j} \end{aligned} \tag{2.12}$$

One of the main challenges that is introduced when dealing with ANNs is that the cost function, mainly due to the activation function and the many layers that are associated with them, is non-convex. Therefore, a proper selection of the learning rate, η, is necessary in order to make sure that the one reaches the global minimum of the cost function as efficient as possible such that the number of iterations can be minimized and hence the computational cost of the ANN can be decreased. Essentially, a large learning rate allows the model to learn faster as the change of the cost function is large as well, with the risk to overshoot the global or local minimum of the cost function, eventually resulting in a larger training error. A small learning rate allows the model to determine the global or local minimum with very high accuracy, but a small learning rate will result in a slower convergence and hence increase the computational cost. Parameters that are pre-defined by the user are also referred to as hyperparameters. The learning rate, as well as the choice of activation function of the model, are common hyperparameters, however, it must be noted that the number of hyperparameters depends on the ANN architecture that is chosen. The hyperparameters of an ANN can influence the performance of the model significantly and are hence necessary to be optimized using trial and error.

Several different approaches for the gradient descent exist. The method that is described above is often referred to as the batch gradient descent, as each update is performed after the summation of all of the data for each iteration. Especially with large data sets, which is very common in the use of ANNs, this can result in an increase in the computational cost of the ANN. Instead of using all the data from the data set, one can lower the number of data that is used for training the ANN model and use a randomized subset of the given data, referred to as mini-batch gradient descent. The advantage of this method is that the amount of data that is used is reduced and hence the computational cost of the model is reduced. However, due to the randomization of the given subset, noise is introduced at each iteration, and convergence of the model is therefore considered to be hard. An alternative to both methods is the stochastic gradient descent that, instead of the batch gradient descent, performs the parameter updates on each example after randomization of the data set. The stochastic gradient descent is usually better with respect to generalization, due to the addition of more noise due to the randomization. The disadvantage of this approach is that due to the increase of noise, the convergence of the model becomes even harder than the mini-batch gradient descent, resulting in a longer computational time of the model. A representation of the mentioned approaches is shown in Figure 2.9

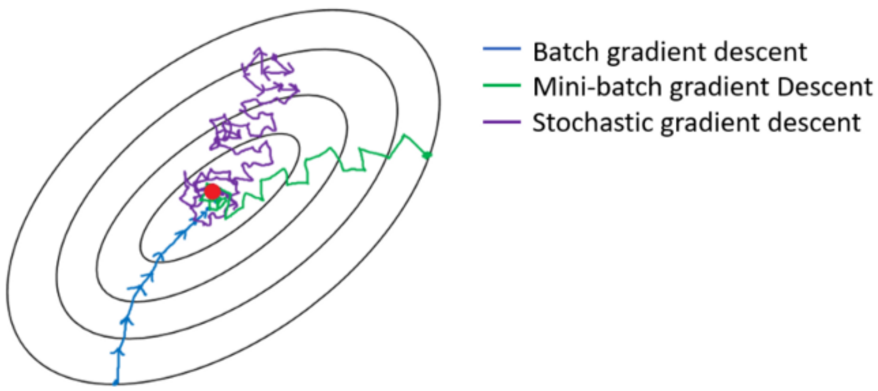


Figure 2.9: Representation of different gradient descent approaches. [5]

Backpropagation
In order to use the gradient descent algorithm, it is necessary to determine the gradient of the cost function with respect to the parameters of the ANN. One of the conventional ways to do this is to introduce a small variation in one of the parameters and obtain the difference between the original value of the cost function and the new value of the cost function. The problem with this conventional way of determining the gradient vector is that this method is slow for a multi-dimensional problem with many parameters, as the same cost function needs to be evaluated for all the parameters individually. Therefore, in order to increase the efficiency of the gradient descent algorithm, it is necessary to use different methods. One of the methods that currently is widely used to determine the gradients of the cost function is called backpropagation [21]. Although the technique of backpropagation already was discovered, the popularity for the use of this technique comes from the study by Rumelhart [24], which showed the efficiency of this method for ANNs above others.

The basic idea of backpropagation becomes clear from the expression of the change in the cost function for a given weighted input. If one considers a small change to the weighted input to one of the neurons in a network, Δz_j^l, such that the activation function of the neuron now is defined as $f(z_j^l + \Delta z_j^l)$, one can determine the change of the cost function with respect to this small change of the weighted input as follows:

$$\Delta C = \frac{\partial C}{\partial z_j^l} \Delta z_j^l \tag{2.13}$$

Using this relation it can be seen that if the gradient of the cost function with respect to the weighted input, $\frac{\partial C}{\partial z_j^l}$, is a high absolute value, one could reduce the cost function drastically by selecting a small change to the weighted input, Δz_j^l, that has an opposite sign to the given gradient. However, if the gradient of the cost function with respect to the weighted input is small, the neuron is already near the optimal value for it to minimize the cost function. Note that this only holds if a small change of the weighted input is considered. From this given example, it can be said that the gradient of the cost function is a measure of the error in the neuron, δ_j^l. Therefore, the error of the $j-th$ neuron in the $l-th$ layer is defined as follows:

$$\delta_j^l \equiv \frac{\partial C}{\partial z_j^l} \tag{2.14}$$

In backpropagation the error is propagated from the output layer to the input layer of the ANN, hence the error and its dependencies is derived backwards as the name already suggests. One can derive the procedure of backpropagation by starting with the error of the output layer, defined as the $L-th$ layer, using the chain rule:

$$\delta_j^L = \frac{\partial C}{\partial z_j^L} = \sum_i \frac{\partial C}{\partial a_i^L} \frac{\partial a_i^L}{\partial z_j^L} = \frac{\partial C}{\partial a_j^L} f'(z_j^L) \tag{2.15}$$

It must be noted that the activation a_i^L only returns a value if $i = j$, returning a_j^L.

Now suppose that the error of the $l+1^{th}$ layer, δ_j^{l+1}, is known. In backpropagation, one would like to express the error of the $l-th$ layer as an expression of the error of the $l+1^{th}$ layer, in order to define the movement of the error backward in the neural network. If one would apply the chain rule, one can obtain the following relationship:

$$\delta_j^l = \frac{\partial C}{\partial z_j^l} = \sum_k \frac{\partial C}{\partial z_k^{l+1}} \frac{\partial z_k^{l+1}}{\partial z_j^l} = \sum_k \frac{\partial z_k^{l+1}}{\partial z_j^l} \delta_k^{l+1} \tag{2.16}$$

From the MLP, the weighted input of the $l+1^{th}$ can be described as:

$$z_k^{l+1} = \sum_j w_{kj}^{l+1} a_j^l + b_k^{l+1} = \sum_j w_{kj}^{l+1} f(z_j^l) + b_k^{l+1} \tag{2.17}$$

Now by combination of Equation (2.16) and Equation (2.17), one will obtain the following relation for the error in the $l-th$ layer, δ_j^l:

$$\delta_j^l = \sum_k w_{jk}^{l+1} \delta_k^{l+1} f'(z_j^l) \tag{2.18}$$

As now the error of the output layer, δ_j^L, is defined and the error of any layer can be expressed as the error of the next layer, as seen in Equation (2.16), one can combine apply both relations to determine the error for any layer in the network. As one mainly is interested in determining weights such that the cost function of the ANN is minimized, one needs to express the errors that are found throughout the layers to the given weights. This is done using the chain rule and the gradient of the cost function with respect to the weights is defined with the following expression:

$$\frac{\partial C}{\partial w_{jk}^L} = \sum_i \frac{\partial C}{\partial z_i^L} \frac{\partial z_i^L}{\partial w_{jk}^L} = \sum_i \delta_i^L \frac{\partial \sum_k w_{ik}^l a_k^{l-1} + b_i^l}{\partial w_{jk}^L} = \delta_j^L a_k^{L-1} \tag{2.19}$$

The same idea holds for the gradient of the cost function with respect to the biases, b_j^L:

$$\frac{\partial C}{\partial w_{jk}^L} = \sum_i \frac{\partial C}{\partial z_i^L} \frac{\partial z_i^L}{\partial b_j^L} = \sum_i \delta_i^L \frac{\partial \sum_k w_{ik}^l a_k^{l-1} + b_i^l}{\partial b_j^L} = \delta_j^L \tag{2.20}$$

The advantage, as can be seen by the derivation of the backpropagation procedure above, is that it simultaneously computes all the partial derivatives, using just one forward and backward pass through the network. This in comparison with the conventional technique, in which only one weight parameter is adjusted for each iteration, shows a tremendous reduction in computational cost, especially for large ANNs with a large number of weights associated. The training of an ANN using the backpropagation method is therefore very efficient.

Generalization

The main feature of an ANN is that it should be able to predict the results of new unexplored data. Training data is used to minimize the error between the predicted and expected outcome, also known as the training error, and hence to determine the values of the weights and biases of the network such that the error of the predicted outcome is minimal. If the training of an ANN is successful, the model is able to predict the outcome of unexplored data well,i.e., with small test or generalization error, the model is said to be generalized. Generalization is a very important aspect in the design of ANNs and should be handled with care in order to prevent under- or overfitting. The training of an ANN mainly exists out of two procedures: First, to reduce the training error by determining the weights and biases that will minimize the cost function for a given data set. Secondly, to determine the generalization error by predicting the outcomes of unexplored data, e.g., test data set, and to minimize the difference between the training error and the generalization error. If the ANN fails to predict the outcome of the training data set and hence the training error is large, the model is said to be underfitting, i.e., the model has a high bias. However, if the ANN is successfully able to predict the outcome of the training data set, but the model is not able to predict the data of unexplored data, e.g., test data set, the model is said to overfit, i.e., the model has a high variance) [21]. A representation of the problem of under- and overfitting is shown in Figure 2.10. It can be concluded from the figure that the under-or overfit compares to the complexity of the model, where underfit is related to a model that is not complex enough and overfit is related to a model that is too complex for the given data. A few common solutions that are used to reduce the generalization error are the collection of more training data, introducing a penalty term at the cost function using regularization, by determining a simpler model with fewer parameters or by reducing the dimensionality of the data such that fewer parameters are necessary to be fitted to the data [21].

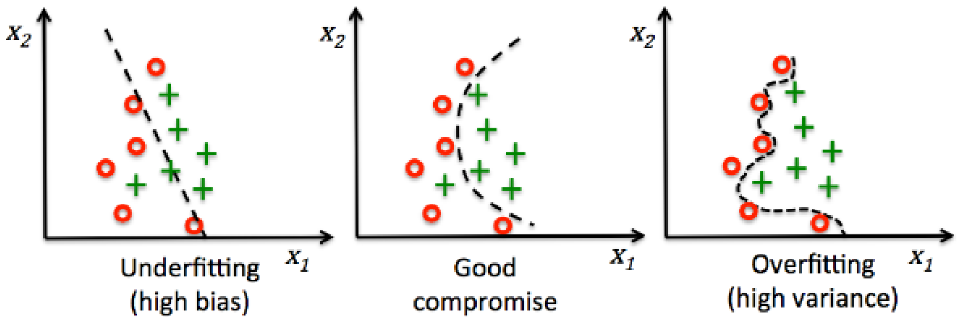


Figure 2.10: Representation of underfitting and overfitting. [21]

Regularization techniques are mainly used to prevent overfitting by tuning the complexity of the models [21]. There are two different well-known regularization techniques: L1 regularization (Lasso Regression) and L2 regularization (Ridge Regression) [15]. Both regularization techniques are an adjustment to the original cost function of the ANN. However, the difference between the two regularization techniques is the penalty term. The L1 regularization (Lasso Regression) adds the absolute value of the coefficient as a penalty term to the cost function:

$$L1_{penalty} = \lambda \sum_k |w_k| \tag{2.21}$$

The L2 regularization adds the squared magnitude as penalty term to the cost function:

$$L2_{penalty} = \lambda \sum_k w_k^2 \tag{2.22}$$

The regularization parameter, λ, controls how well the training data will fit while obtaining small weights. If the regularization parameter is chosen to be very small or zero, one will obtain the original cost function. An increase of the regularization parameter will increase the regularization strength, hence the introduction of more information to penalize extreme parameter values [21]. These regularization methods are specially used for large data sets containing a large set of features [15].

3

State-of-the-art: Surrogate modeling for non-intrusive Reduced-Order Models

Previous research that was introduced by Papp to predict the surface pressure coefficients of an UCAV configuration has shown that the neural network approach is a suitable approach to be used as surrogate model [18]. However, one of the drawbacks of the proposed Convolutional Neural Network (CNN) architecture is that this neural network architecture is only able to handle one time step at a time, i.e., previous time steps are neglected and therefore time-history effects are excluded in the prediction of the surface pressure distribution. This chapter will explore neural network architectures that are able to handle time sequential data and will discuss the proposed surrogate model used for this research.

3.1. Recurrent neural networks (RNNs) and its limitations

ANNs that have been discussed so far, e.g., FFNNs and CNNs, are not capable of handling the order of input samples. For these models the input is passed through the model using feedforward and backpropagation steps and the weights are updated independently on the order of samples. It is said that these models do not have memory of the past seen samples [21]. Up until now, it therefore has been assumed that the input data for machine learning algorithms is independent. However, this assumption is not valid when one deals with sequential data. In order to handle sequential data, or sequences, the Recurrent Neural Network (RNN) architecture was developed that is able to remember past given input data. RNNs currently are widely used for voice recognition and translation applications. Sequential data modeling can come as a vector of ordered inputs, a vector of ordered outputs or in several cases both. Dependent on the type of sequential modeling tasks, the architecture of the ANN will differ. An example of different types of sequential modeling is shown in Figure 3.1.

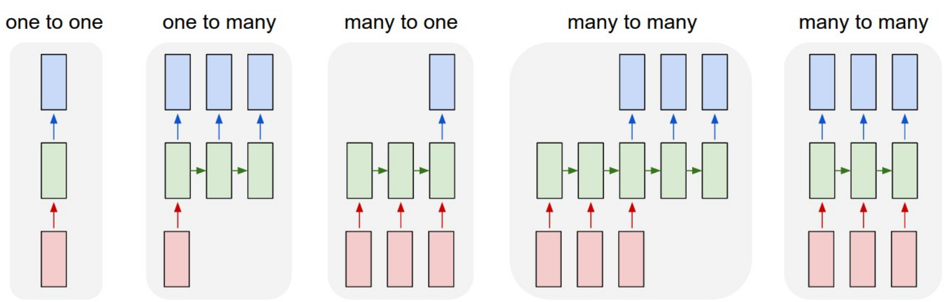


Figure 3.1: Different types of sequence modeling tasks [9].

Contrary to a standard FFNN, the hidden layers in a RNN architecture not only receive information from the input layers but the hidden layers receive information from both the input layer and the hidden layer from the previous time step. This procedure usually is visualised as a loop in the RNN architecture, also known as a recurrent edge [21]. If one would unfold the loop in time, one would obtain a deep FFNN with shared weights as shown in Figure 3.2 [3].

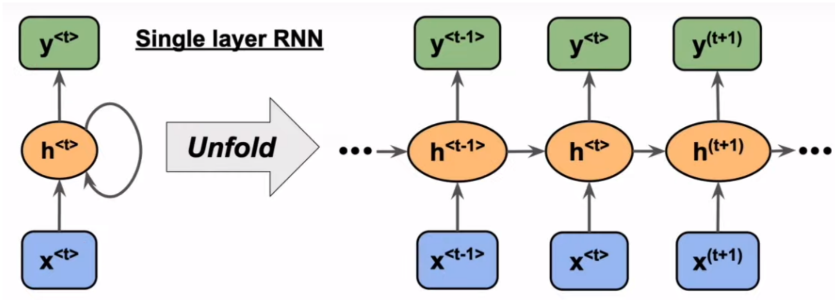


Figure 3.2: The architecture of an unfolded single layer RNN [21].

As already mentioned before, in the RNN architecture the connections between the different hidden states, the recurrent edges of the network, are associated with a weight vector. It must be noted that these weights do not depend on time and hence are shared with all the states in the RNN model. Therefore, the calculation of the activation functions of the hidden layers is very similar to the approach of calculating the activation functions of MLPs. In Figure 3.3 a representation of a single layer RNN can be seen with the associated weights.

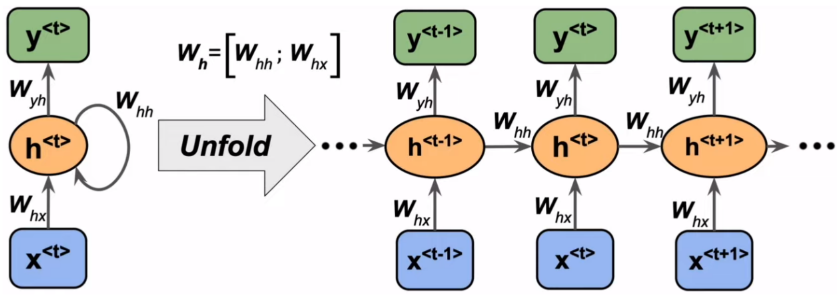


Figure 3.3: The representation of a single layer RNN with associated weights [21].

From Figure 3.3 one can obtain the relation for activation of the hidden layer at the time step t:

$$h^t = f_h(W_{hx}x^t + W_{hh}h^{t-1} + b_h) \tag{3.1}$$

Once the activation of the hidden layers is calculated, one can calculate the output at time step t using the following relationship:

$$y^t = f_y(W_{yh}h^t + b_y) \tag{3.2}$$

In order to train the ANN, one can use Backpropagation Through Time (BPTT). BPTT is the use of backpropagation for multiple time steps and was introduced by Werbos [30]. The basic idea behind BPTT is that the overall cost C is the sum of all the cost functions at the adjacent given timesteps:

$$C = \sum_{t=1}^{T} C^t \tag{3.3}$$

As the cost function is dependent on the hidden layers of the previous all the previous time steps, one can calculate the gradient of the cost function according to the chain rule using the following relationship:

$$\frac{\partial C^t}{\partial W_{hh}} = \frac{\partial L^t}{\partial y^t}\frac{\partial y^t}{\partial h^t}\left(\sum_{k=1}^{t}\frac{\partial h^t}{\partial h^k}\frac{\partial h^k}{\partial W_{hh}}\right) \tag{3.4}$$

The gradient of the hidden units in the layers with respect to previous time steps, $\frac{\partial h^t}{\partial h^k}$, can be computed as a multiplication of adjacent time steps:

$$\frac{\partial h^t}{\partial h^k} = \prod_{i=k+1}^{t} \frac{\partial h^i}{\partial h^{i-1}} \tag{3.5}$$

Although BPTT is an efficient way to determine the gradient of the cost function with respect to the weights given, it also introduces a new problem: vanishing or exploding gradients. The problem of vanishing or exploding gradients occurs in very deep networks, where the amount of layers are large, but the phenomenon is dominant in RNN networks. The problem arises due to the backpropagation of the error into the network due to the gradient of the hidden units with respect to previous time steps, $\frac{\partial h^t}{\partial h^k}$, which has $t-k$ multiplications. As the weights are shared with all the states in the RNN network, this means that the weights of the recurrent edge will be multiplied with a factor of $t-k$ resulting in a factor of w_{hh}^{t-k}. Hence, if the weight vector $|w_{hh}| < 1$, the resulting weight vector becomes very small when $t-k$ is large. The same holds for the weight vector $|w_{hh}| > 1$, where the resulting weight vector becomes very large when $t-k$ is large. A desirable solution would therefore be that the weights of the solution are approximately equal to 1, $|w_{hh}| = 1$, such that the relative change for each time step is minimal. As already mentioned, the same holds for deep ANN architectures, where the amount of layers is relatively large, causing the backpropagation of the error to explode or vanish the gradient with respect to the cost function. The problem that is presented here does not mean that the training of a RNN is impossible, however it means that the gradient descent becomes increasingly inefficient when the temporal span of the dependencies increases, hence the convergence of the training will take a long time or will not succeed at all [3]. Several solutions to overcome the vanishing and exploding gradient problem exist:

- **Gradient clipping**
  This technique is a partial solution that can be used to avoid exploding gradients. Gradient clipping sets a maximum value to the gradients if they grow too large, to stop the gradient becoming larger.

- **Complexity reduction**
  One of the most simple ways to avoid exploding or vanishing gradients is by reducing the complexity of the ANN by limiting the amount of layers that can be used in an ANN architecture. By limiting the amount of layers, the multiplication of the weights is also limited. It must be noted that when one reduces the number of layers, the complexity of the model reduces and hence mapping different features can be difficult.

- **Truncated Backpropagation Through Time (TBPTT)**
  This technique limits the number of time steps that the signal will backpropagate through the model. In this way the temporal span of the dependencies is decreased and hence exploding or vanishing gradients can be avoided.

- **Weight initialization**
  A proper weight initialization can be a partial solution for vanishing or exploding gradients. By selecting a proper method for the weight initialization, one can avoid that the output of the activation functions already tend to be large at the start of training the ANN.

- **Introduction of the Long Short-Term Memory (LSTM)**
  One that has been more successful, especially for RNN modeling, is the introduction of the LSTM. The LSTM architecture uses a memory cell to model long-range sequences to avoid the vanishing gradient problem.

3.1.1. Candidate network: the Long Short-Term Memory (LSTM) network

The LSTM architecture was introduced by Hochreiter and Schmidhuber [7] to overcome the vanishing gradient problem. Although a lot of research was already performed on increasing the efficiency of RNNs, the LSTM architecture was one of the first that lead to many more successful runs, and learns much faster than previous solutions, e.g., weight initialization and truncated backpropagation through time [7].

The basic idea of the LSTM is that the hidden layer of a RNN is replaced by a memory cell. The unfolded LSTM memory cell can be seen in Figure 3.4. Each memory cell contains desirable weights, $|w| = 1$, to overcome vanishing gradient problems and each cell has a cell state, C^t, associated that is dependent on the previous time step. The LSTM memory cell is able to remember information over time in the RNN architecture via so-called gates that add or remove information of the cell state. These gates learn and extract which information is relevant to remember or forget, which is trained during the trainig process.

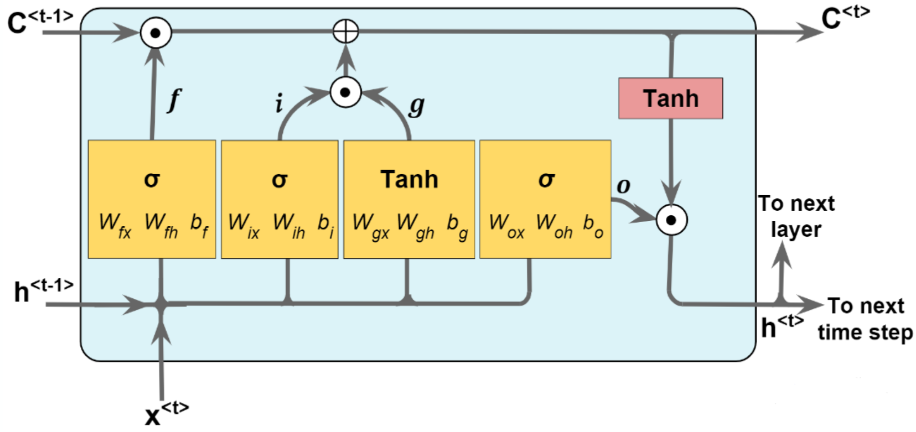


Figure 3.4: LSTM memory cell [21].

In the LSTM memory cell, three different types of gates are used: the forget gate, the input gate and the output gate. All the gates contain sigmoid or hyperbolic tangent activation functions, which return a linear combination by performing matrix-vector multiplications on their input. The different types of gates are explained below.

- **The forget gate** f_t
  This gate determines what information should be remembered or removed before it is passed on to calculate the new cell state, C^t. The values that are returned are between 0 and 1 and are relevant for passing the information to the calculation of the new cell state, i.e., a value of 0 means that the information should be suppressed and a value of 1 means that the information is passed through. The forget gate is computed as follows:

$$f_t = \sigma\left(W_{xf} x^t + W_{hf} h^{t-1} + b_f\right) \tag{3.6}$$

- **The input gate** i_t
  This gate is used for calculation of the new cell state, C^t, based on the information that has been passed through by the forget gate and the cell state of the previous time step, C^{t-1}. Both the previous hidden state as the input are used in a sigmoid and hyperbolic tangent function, where the sigmoid determines how relevant the new given input is and hence determines how much information from the new input will be added to the network. Both the input gate, i_t, as the input node, g_t, are calculated using the following relations:

$$i_t = \sigma\left(W_{xi} x^t + W_{hi} h^{t-1} + b_i\right) \tag{3.7}$$

$$g_t = \tanh\left(W_{xg} x^t + W_{hg} h^{t-1} + b_g\right) \tag{3.8}$$

  As now both the addition of new information as the information that should be passed through the network are known, one can determine the current cell state, C^t, using the following relation:

$$C^t = \left(C^{t-1} \odot f_t\right) \oplus \left(i_t \odot g_t\right) \tag{3.9}$$

- **The output gate** o_t
  This gate determines how to update the value of the current hidden state, h^t:

$$o_t = \sigma\left(W_{xo}x^t + W_{ho}h^{t-1} + b_o\right) \tag{3.10}$$

From this, the hidden state can be calculated using the following:

$$h^t = o_t \odot \tanh\left(C^t\right) \tag{3.11}$$

Both the current cell state, C^t as the current hidden state, h^t, are then passed through the next time step.

The advantage of the LSTM model is that the network architecture remembers the relevant information that is passed through in the time sequence, irrelevant of the length of the time sequence given, with respect to original RNN architectures where the problem of vanishing or exploding gradients leads to the ineffectiveness of the model due to the relevant information of the sequences that is forgotten by the network.

Variations on the LSTM network architecture exist that are used to decrease the computational cost of the network, e.g., the Gated Recurrent Unit (GRU). The GRU is similar to the LSTM model but does not use a cell-state in its architecture. The GRU contains two different gates, a reset and update gate. As the GRU has less gates and hence less operations to calculate, the GRU is deemed to be faster than the LSTM architecture. However, no significant difference between the performance of the two model architectures is presented and usually both models are performed to determine which one is more suitable for the given data [19].

3.2. The POD-LSTM Long-Short Term Memory (LSTM) network

The previous section introduced the LSTM architecture as one the many RNN architectures. However, the LSTM architecture currently is one of the most successful RNN architectures that is used in the prediction of time sequential data and is used in many applications such as prediction of financial trends. As discussed in Chapter 2, the POD method is one of the most efficient methods for dimensionality reduction and the usage of this method leaves the user with a constructed reduced-order basis, that is obtained from a pre-selected number of spatial POD modes and its corresponding time coefficients. As the spatial POD modes are only constructed once from the high-fidelity training samples, the only relationship that needs to be approximated with the use of a surrogate model is the relationship between the input of the model, e.g., the angle of attack α and the pitch rate q, and the time coefficients that are necessary to project the full-order solution on the reduced-order basis. This leaves the user with limited sequential data that is suitable for the LSTM architecture. Knowing this, the proposed reduced-order model can be constructed with the use of the Proper Orthogonal Decomposition (POD) method in combination with the surrogate model that is based on the LSTM architecture. In short, this combination can be referred to as the POD-LSTM model. The POD-LSTM model shows potential as the only data that is used to train the model are the time coefficients of the corresponding spatial POD modes and the input state variables of the high-fidelity samples, reducing the complexity of the neural network architecture tremendously with respect to previous studies.

4

Methodologies: Construction of the POD-LSTM model

This chapter will provide the methods that are used for the construction of the proposed POD-LSTM model. First, the case study that will be used for the current research will be introduced. Finally, the methodologies behind the proposed non-intrusive reduced-order model will be described in detail.

4.1. The MULDICON Unmanned Combat Aerial Vehicle

The case study that will be used for the assessment of the proposed non-intrusive ROM approach for this thesis report is the Unmanned Combat Air Vehicle (UCAV) configuration of the STO research task group AVT-251, called the MULti-DIsciplinary CONfiguration (MULDICON). The purpose of the MULDICON design was to assess the performance of advanced numerical tools of the complex interaction of over-wing vortices, leading to non-linear aerodynamic characteristics, and collaborative approaches to an UCAV design for military missions [4],[16]. The Royal Netherlands Aerospace Centre (NLR) has been contributing to the development of ROMs for aerodynamic load prediction for the UCAV design and this thesis report will elaborate on the further development of this research in collaboration with NLR. The MULDICON was chosen as a case study for this thesis as it allows to assess the performance of the proposed non-intrusive ROM model with previously generated data and to reflect the outcomes on previous approaches that were developed within the 251st NATO task group.

The MULDICON is a tailless Blended-Wing-Body (BWB) that is designed to fulfill the requirements of a long-range transport mission with a radius of 1500 km and no aerial refueling, which in this case is closely related to the low-level penetration bomber classification [12]. The design mission for the MULDICON is shown in Figure 4.1. The main parameters of the MULDICON are given in Table 4.1.

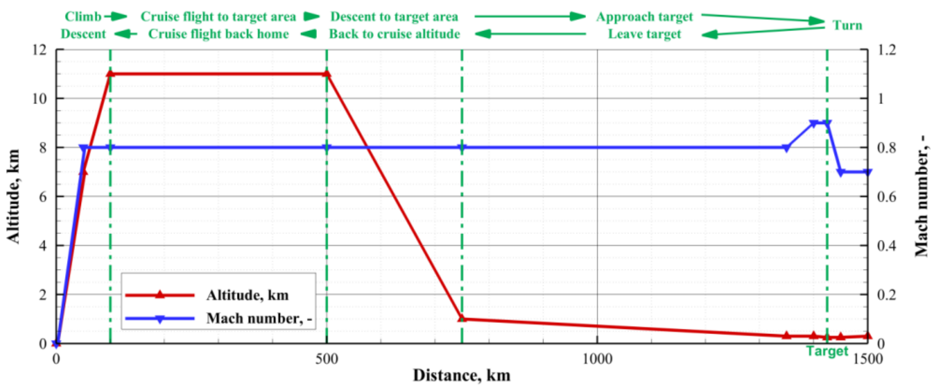


Figure 4.1: MULDICON design mission profile [12].

Table 4.1: MULDICON main properties [18],[12].

Parameter	Description
Propulsion	1 Turbofan engine w/o afterburner
Engine integration	Buried
Payload storage	Internal
Payload mass	2 x 1000 kg
Design range	3,000 km w/o aerial refueling
Fuel reserve	≈ 45 min
Cruise altitude	11 km
Cruise Mach Number	0.8
Stability margin	0 - 3 % MAC

Table 4.2: MULDICON wing geometry properties [12].

Parameter	Symbol	Unit	Value
Wing reference area	S_{ref}	m^2	77.8
Wing span	b_{ref}	m	15.38
Reference length	l_{ref}	m	6.0
Moment reference point (x,y,z)	$M_{ref}(x,y,z)$	m	(6,0,0)

The design of the MULDICON has a sharp leading edge of 53 degrees and a trailing edge of 30 degrees with a span of 16 meters. The wing planform of the MULDICON is shown in Figure 4.2 and its relevant geometrical properties are shown in Table 4.2.

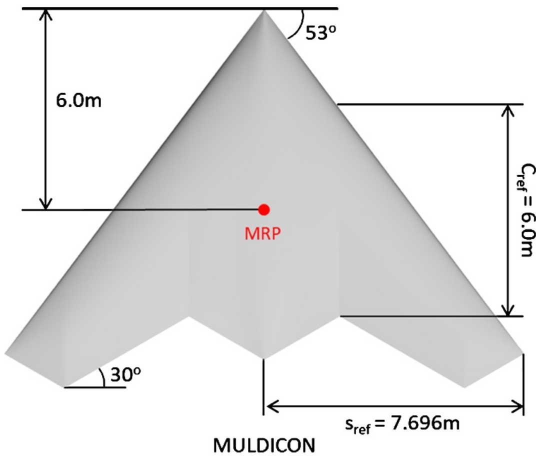


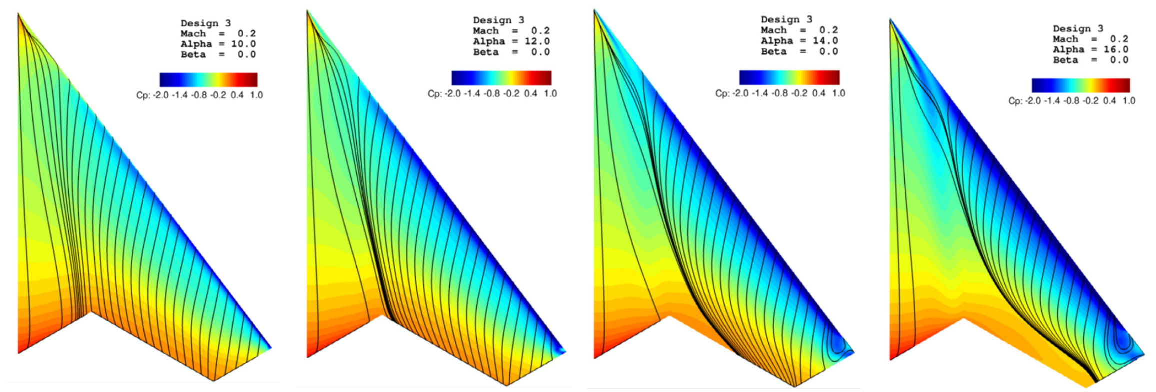
Figure 4.2: Planform of the MULDICON UCAV [28].

As the purpose of this thesis is to develop a non-intrusive ROM that is capable of capturing both linear and non-linear flow phenomena, the flight conditions that are considered for this thesis are derived from the MULDICON design mission, as depicted in Figure 4.1, where both phenomena occur. The take-off configuration of the design mission will be used as a design point for the high-fidelity simulations, as it both includes the largest excitation of the angle of attack, which is 20 degrees, and the highest requirement of the pitch rate, which is 20 degrees per second [12]. For simplification, only symmetric motions are considered in the approach for the development of the non-intrusive ROM at this design point. An overview of the flight conditions that will be used for the high-fidelity simulations is shown in Table 4.3.

Table 4.3: Flight conditions for the take-off configuration [12].

Flight Conditions	Symbol	Unit	Value
Mach Number	M_∞	-	0.2
Velocity	V_∞	m/s	68.06
Ambient pressure	P_∞	Pa	101325
Ambient temperature	T_∞	K	288.15
Ambient density	ρ_∞	kg/m^3	1.225
Angle of attack range	α	deg	[-2,20]
Maximum pitch rate	q_{max}	deg/s	20

The design point of the take-off configuration is considered to be the optimal design point for the development of the non-intrusive ROM, as it allows for the investigation of both linear and non-linear phenomena of the flow field. An example of the flow visualization for varying angles of attack is shown in Figure 4.3. It can be seen that at an angle of attack of 10 degrees and above, two leading-edge vortices are formed over the inboard section of the upper surface. Furthermore, at the wingtips, a small vortex is present which moves inboard and grows with increasing angle of attack. As it grows, the strength of the vortex is reduced which results in a reduction of local lift. The combination of both flow separation and vortices make the MULDICON aerodynamics non-linear and unsteady at angles of attack of 12 degrees and higher and therefore the MULDICON is considered as an interesting case for the assessment of ROM approaches [16].



(a) Flow visualization for α = 10 deg.

(b) Flow visualization for α = 12 deg.

(c) Flow visualization for α = 14 deg.

(d) Flow visualization for α = 16 deg.

Figure 4.3: NLR ENSOLV flow development of the MULDICON Design3 with angle of attack α=10, 12, 14, 16 deg., for low-speed M=0.2 [16].

4.2. Proposed non-intrusive Reduced-Order Model: The POD-LSTM network

This section will describe the proposed non-intrusive ROM that will be used for the prediction of the surface pressure coefficients of the MULDICON. The proposed ROM is based on the non-intrusive ROM architecture as described in Chapter 2 and the flow diagram of the proposed ROM is shown in Figure 4.4. As can be seen in the figure, the non-intrusive ROM has two different stages:

- **Offline stage**
  The offline stage is the stage where the ROM is constructed. The offline stage can be split up into three different sections: First, the number of sampling points of the model is determined and constructed. The sampling strategy (Design of Experiments) that is used to collect the sampling points determines the minimum amount of high-fidelity samples that is necessary to train the surrogate model of the ROM in an efficient and sufficient matter. The sampling strategy that is applied for this ROM is known as the Schroeder Sweep [8] and will be elaborated on in the next sections of this chapter. Second, a Model Order Reduction (MOR) technique is applied to the high-fidelity sampling points by using a modal reduction method. For this particular ROM, the POD technique as described in Chapter 2 will be used to reconstruct a reduced-order basis that is based on a limited number of spatial POD modes.

The reduced-order basis that is constructed from the limited number of spatial POD modes together with its time coefficients are normalized and split up into small sets of data. Finally, a surrogate model is constructed that can approximate a relationship between a given input and the desired output using a data-driven approach. As described in Chapter 2 several surrogate modeling techniques exist that can approximate a linear or non-linear relationship between the given input and output data. For this ROM the neural network approach will be used to construct a surrogate model. In particular, the Long Short-Term Memory (LSTM) model will be used that is capable of handling sequential data. The given input of the surrogate model is a series of state variables of the MULDICON for a given range of time steps and the desired output is the surface pressure coefficient distribution at the last time step in the series. Once the LSTM model is constructed, the model is trained using the data that is constructed from the sampling strategy. The trained LSTM model can predict the normalized time coefficients of the spatial POD modes for a given set of state variables.

- **Online stage**
  The online stage is the stage where the trained surrogate model is used to predict new sampling points that were not covered by the sampling strategy of the offline stage. For this thesis, both steady and unsteady simulations are used to evaluate the performance of the proposed ROM to assess the prediction of both linear and non-linear aerodynamic phenomena. The prediction of the surrogate model for a new set of state variables gives the normalized time coefficients for the spatial POD modes that were constructed in the offline stage. The flow field of the MULDICON can be reconstructed with the reduced-order basis from the offline stage and the de-normalized predicted time coefficients of the surrogate model.

This section will discuss all the essentials that are necessary to construct the non-intrusive ROM as depicted in Figure 4.4. This section will first discuss the sampling strategy that is used for the ROM. Second, this section will discuss the collection of the high-fidelity sample that is used to train the surrogate model. Third, the modal decomposition method that is used for the ROM is explained. Finally, the surrogate model that is used for the ROM and the evaluation methods of the non-intrusive ROM are explained.

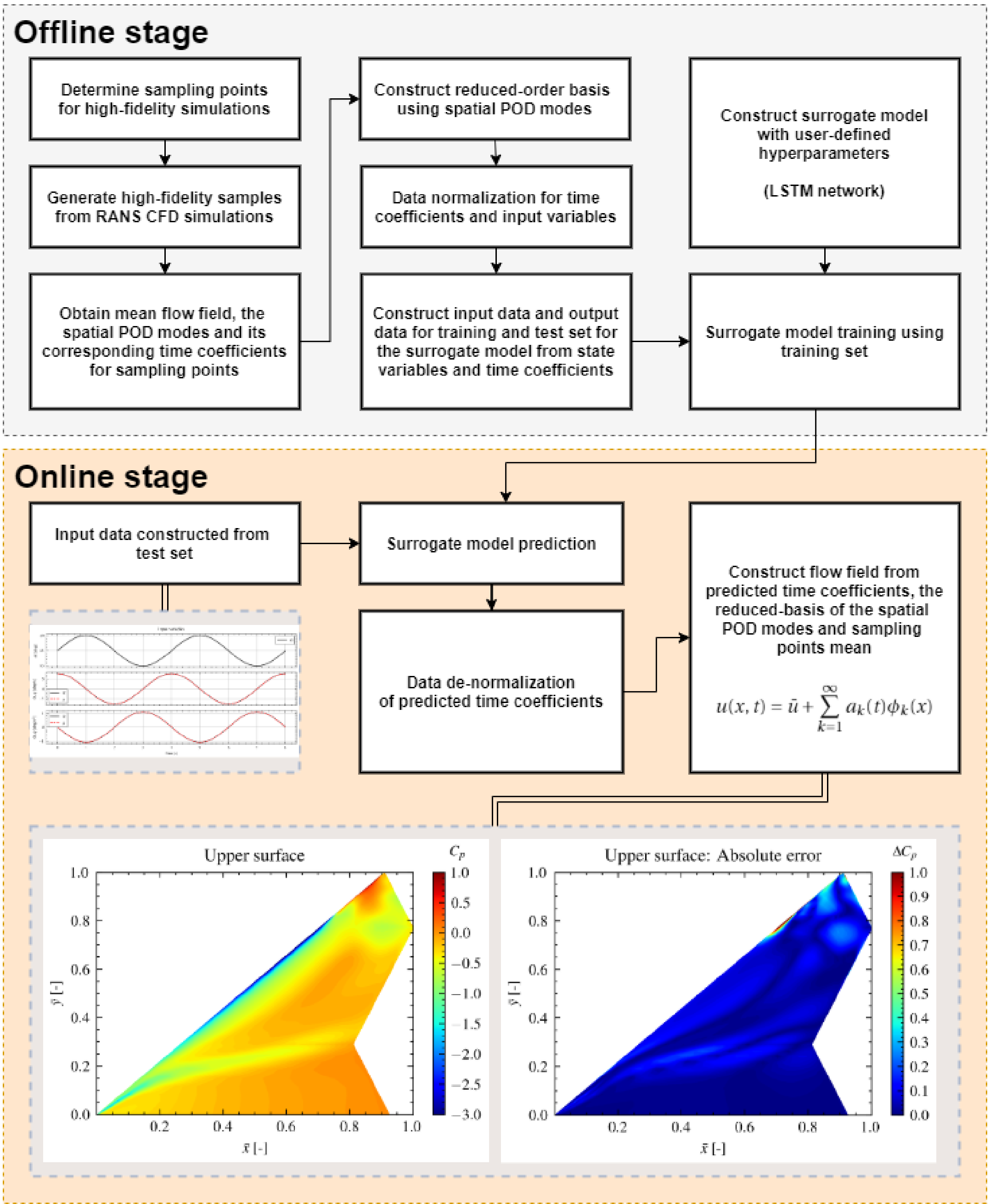


Figure 4.4: Proposed POD-LSTM non-intrusive Reduced-Order Model architecture.

4.2.1. Sampling Strategy: The Schroeder Sweep

The selection of the number of sampling points that are constructed from high-fidelity tools, e.g., CFD simulations, is of great importance for the construction of a non-intrusive ROM. For example, the number of high-fidelity samples should be sufficient enough in order to cover the desired regressor space, but the number of samples should not be too large in order to avoid the computational expense and therefore making the use of ROMs redundant. Thus, to extract maximum information about the underlying input-output relationship of the high-fidelity simulations, the samples that are chosen should fill the regressor space in an optimal sense. The design of such a strategy is also referred to as Design of Experiments (DoE) [10]. Several DoE techniques exist in order to determine the required amount of high-fidelity samples necessary, as well as the regressor range for which these samples are covered. This section will discuss the DoE technique that is based on optimality criteria and is used to obtain the high-fidelity samples from CFD simulations that are used to train the surrogate model of the ROM. First, the reference frame of the MULDICON dynamics model will be discussed. Second, the motion variables and the dynamics model that will cover the design space are discussed. Finally, the DoE technique is discussed for the generation of the training data.

Flight dynamics model

In order to generate the high-fidelity samples from CFD simulation software, it is necessary to determine the motion variables of the aircraft at a given time instant. The motion variables of the MULDICON are described by a simple aircraft dynamics model, where only the attitudes, linear and angular velocities, and the accelerations of the aircraft are of interest. Using an earth frame as a reference, the attitude of the MULDICON is commanded by the angle of attack α and the pitch angle θ. As the motion variables are prescribed by previous research from Papp ([18]) and only is used to gather the high-fidelity samples, further investigation of the flight dynamics model is out of the scope of this thesis work. The motion variables that are used in this work are the same as previous studies and the reference frame for these variables is shown in Figure 4.5.

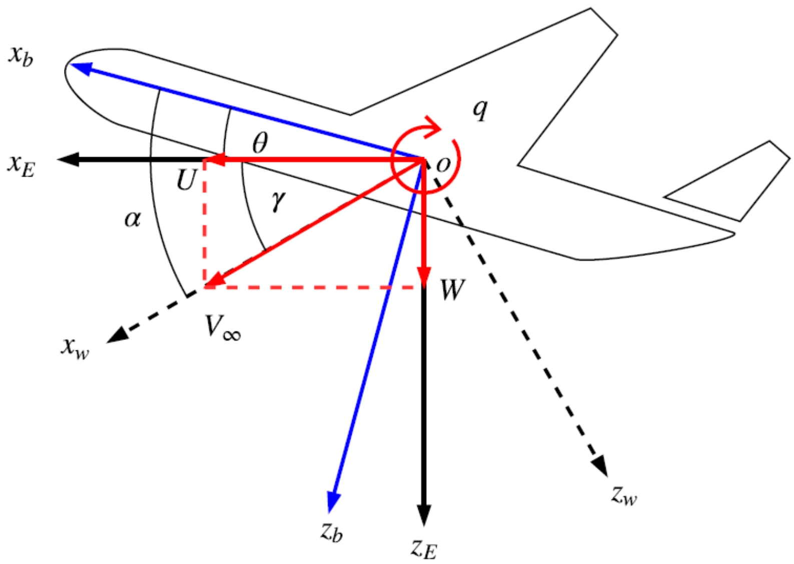


Figure 4.5: Motion variables in-flight dynamic reference frame [18].

Design of Experiment (DoE) technique: The multi sine input design

The use case of this thesis, the MULDICON UCAV, allows the identification of both linear and nonlinear aerodynamic flow phenomena. As the generation of data using high-fidelity simulation tools, e.g., CFD, is computationally expensive, it is necessary to limit the number of samples to outweigh the benefits of the use of the ROM approach. Capturing all the relevant flow properties within a limited amount of high-fidelity samples is challenging and therefore a proper DoE technique is necessary. Extensive research has been performed to investigate different *a priori* DoE techniques for system identification methods. One of the techniques mentioned in the paper by Jirasek ([8]) that is also used for this thesis is based on the use of optimality criteria. This thesis will make use of previously generated data as a benchmark for the assessment of the new proposed ROM technique. As described previously by Papp [18], generally the motion of an aircraft is described by the excitation of the control surfaces. However, due to the simplified model of the MULDICON and the lack of control surfaces, the DoE technique will be applied to the state variables α(t) and θ(t) directly. For the scope of this project, only symmetrical excitations will be considered.

The approach that was used is an extension of the Schroeder sweep input design method [25] to construct multiple orthogonal input signals with optimized peak factors for real-time parameter estimation as described by Morelli [14]. Each input to the aircraft motion is described as a phase-shifted set of sinusoids and takes the following form:

$$u_j = \sum_k A_k cos\Big(\frac{2\pi k t}{T} + \phi_k\Big) \tag{4.1}$$

Table 4.4: Summary of the constructed multisine signals [18]

Parameter	Symbol	Unit	Value
Signal duration	T	s	25
Minimum frequency	f_{min}	Hz	0.04
Maximum frequency	f_{max}	Hz	1
Number of frequencies	n_f	-	25
Initial value for angle of attack	α_0	deg	12.5
Amplitude for angle of attack	A_α	deg	10
Initial value for pitch angle	θ_0	deg	0
Amplitude for pitch angle	A_θ	deg	10

The main benefit of this technique is that the only *a priori* information that is necessary to use this technique is an estimation of the frequency band for the system dynamics. The idea behind this technique is that multiple multi sine input signals can be constructed that will excite the state variables simultaneously due to its orthogonality but also are designed in an optimal sense for a given input due to the optimization of the Relative Peak Factor (RPF) of the signals. The RPF, also known as the crest factor, is a measurement of the efficiency of the signal in terms of the amplitude range of the signal. A signal with a RPF of 1 is considered to be optimal, hence low RPFs are desired for the construction of multi sine input signals. The signals are optimized for their RPF by altering the individual phase shift of each sinusoid. For more information about the multi sine input design, one is referred to the paper by Morelli in which the entire approach is extensively discussed [14].

The multi sine input design requirements are obtained from the requirements of the chosen design point of the MULDICON [12]. For the take-off configuration a maximum angle of attack of 20 degrees was found and a maximum pitch rate requirement of 20 deg/s. These requirements can be translated into inputs for the construction of the input signals. The angle of attack range that will be considered for the construction of the input signal is between 0 and 20 degrees. To make sure that the angle of attack range is covered well, the nominal value of the input signal for the angle of attack is set to 12.5 degrees with an amplitude of 10 degrees. Furthermore, as angular rates for both the angle of attack and pitch are identical, the same amplitudes are defined for the pitch angle, however, the nominal value used for the pitch angle was set to 10 degrees. The frequency of the input signals was set according to the calculation of the reduced frequency of the signal, whereas a reduced frequency of 0.01 was set as minimum as this range of reduced frequencies can still be regarded quasi-steady, resulting in a minimum frequency of 0.04 Hz. The maximum frequency of the signal was set to 1 Hz, which yields more than double the pitch rate requirement (50 deg/s) to extend and obtain better coverage of the regressor space. The frequency resolution is based on the lowest frequency of the signal, hence the total signal is built from 25 sinusoids. A summary of the constructed signals is shown in Table 4.4.

The optimized multisine signals for the angle of attack and pitch angle are shown in Figure 4.6 and have relative peak factors of $\mathbf{RPF}(u_\alpha) \approx 1.05$ and $\mathbf{RPF}(u_\theta) \approx 1.02$ respectively.

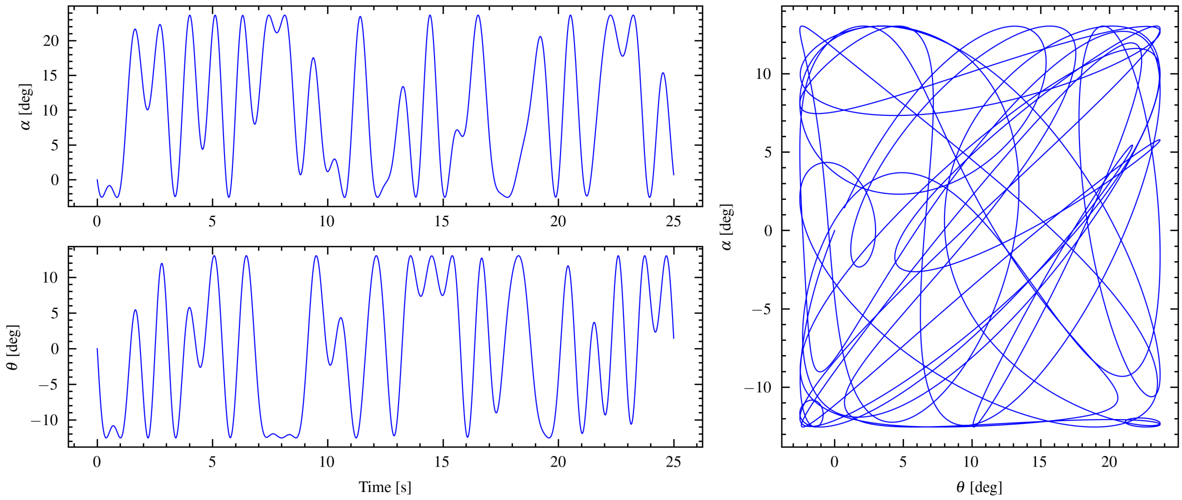


Figure 4.6: Optimized multisine signal of the motion variables with relative peak factors of $\mathbf{RPF}(u_\alpha) \approx 1.05$ and $\mathbf{RPF}(u_\theta) \approx 1.02$ [18].

The regressor space coverage for the most relevant state variables, namely the angle of attack α, the pitch rate q and the angular rate of the angle of attack $\dot{\alpha}$, is shown in Figure 4.7. It can be seen that the generated signals cover the regressor space well within the requirements of the given design point. However, it must be noted that a proportional part of the input signals is concentrated near the boundary of the regressor space, which is a typical property for the Schroeder sweep. One could use different input signals, e.g., the (piecewise) spiral maneuver, in order to cover the lower frequencies of the regressor space if the Schroeder sweep seems to provide insufficient coverage [8]. The full regressor space coverage for the projected state variables is shown in Figure 4.8.

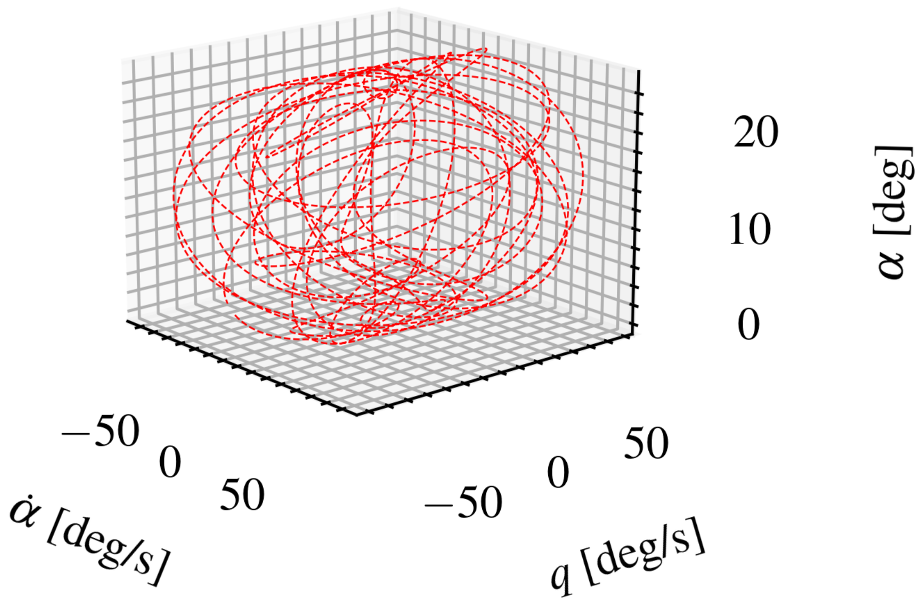


Figure 4.7: Regressor coverage of the training maneuver for the state variables α, $\dot{\alpha}$ and q.

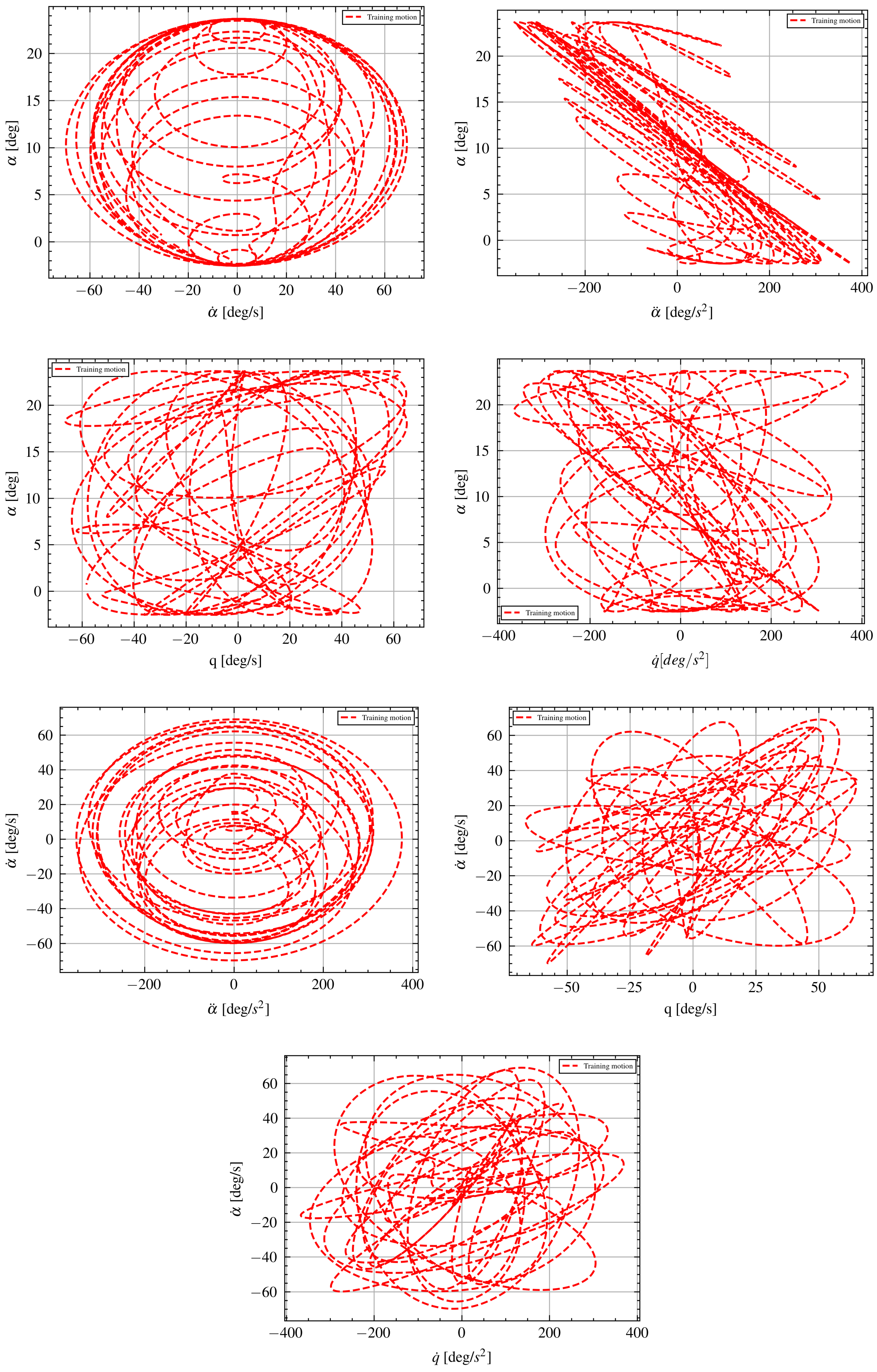


Figure 4.8: Projected regressor coverage of the training maneuver for different state variables.

Table 4.5: Pitch oscillation motion variables resolved in the inertial earth reference frame [18].

Variable	Symbol	Dimension	Values											
Nominal AoA	A_0	[deg]		5		10		10		15				
Amplitude	A	[deg]		5			5			10			5	
Frequency	f	[Hz]	0.25	0.5	1.0	0.25	0.5	1.0	0.125	0.25	0.5	0.25	0.5	1.0
Duration	T	[s]	4	2	1	4	2	1	8	4	2	4	2	1
Horizontal distance	x	[m]	270.5	135.1	67.4	267.5	133.6	66.6	531.9	265.9	123.8	262.3	131.0	65.4
Vertical distance	z	[m]	23.7	11.8	5.9	47.2	23.6	11.8	47.2	23.6	11.8	70.4	35.2	17.6
Horizontal velocity	U_0	[m/s]		67.8			67			67			65.7	
Vertical velocity	W_0	[m/s]		5.9			11.8			11.8			17.6	

Table 4.6: Plunge oscillation motion variables resolved in the inertial earth reference frame [18].

Variable	Symbol	Dimension	Values											
Nominal AoA	A_0	[deg]		5		10		10		15				
Amplitude	A	[deg]		5			5			10			5	
Frequency	f	[Hz]	0.25	0.5	1.0	0.25	0.5	1.0	0.125	0.25	0.5	0.25	0.5	1.0
Duration	T	[s]	4	2	1	4	2	1	8	4	2	4	2	1
Horizontal distance	x	[m]	270.5	135.1	67.4	267.5	133.6	66.6	531.9	265.9	123.8	262.3	131.0	65.4
Vertical distance	z	[m]	23.7	11.8	5.9	47.2	23.6	11.8	93.8	46.9	23.4	70.3	35.1	17.5
Horizontal velocity	U_0	[m/s]		67.8			67			67			65.7	
Horizontal velocity min.	U_{min}	[m/s]		67.3			65.7			64.0			64.0	
Horizontal velocity max.	U_{max}	[m/s]		68.1			67.8			68.1			67.0	
Vertical velocity	W_0	[m/s]		5.9			11.8			11.8			17.6	
Vertical velocity min.	W_{min}	[m/s]		0.0			5.9			0.0			11.8	
Vertical velocity max.	W_{max}	[m/s]		11.8			17.6			23.3			23.3	

Evaluation of the regressor space: Test cases

In order to assess the performance of the proposed ROM, high-fidelity samples are gathered that are well within the limits of the regressor space but are not used to train the surrogate model. Two different test cases are considered in order to evaluate the performance of the model: steady and unsteady simulations.

The steady simulations are obtained from singular high-fidelity samples and cover the range of the angle of attack from 0 to 20 degrees with zero rates and accelerations. For the unsteady simulations, two different types of harmonic motions are considered: pitch and plunge oscillations. During a pure sinusoidal pitching maneuver, the pitch angle is changed while the absolute velocity vector of the aircraft is kept unchanged. In contrast, during a pure sinusoidal plunging motion, the pitch angle is preserved and the normal and longitudinal components of the absolute velocity vector differ causing the instantaneous angle of attack to change. As for both motions, only the angle of attack is used in the high-fidelity simulations, the input of the angle of attack is considered the same for both motions. The other motion variables are derived from the boundary conditions. An illustration of the pure sinusoidal pitch and plunge oscillations are shown in Figure 4.9.

Figure 4.9: Definition of pitch and plunge oscillations [6].

The high-fidelity simulations for the MULDICON will be performed within the boundaries of the regressor coverage space as defined by the requirements of the design point. The high-fidelity simulations will be constructed from various sets of pure pitch and plunge sinusoidal oscillations with angle of attack amplitudes of 5 and 10 degrees around different sets of nominal angle of attack values of 5, 10, and 15 degrees. The frequency of the signals will be within the bounds of the regressor space and are set to 0.25, 0.5, and 1 Hz respectively. A full overview of the different pitch and plunge motions is shown in Table 4.5 and Table 4.6, respectively.

4.2.2. Full-order model for computational fluid dynamics

All the high-fidelity sample points are obtained from CFD simulations using the in-house solver ENSOLV of the Royal Netherlands Aerospace Centre (NLR). ENSOLV is an advanced solver capable of solving three-dimensional, time-dependent flows around complex configurations using the Reynolds-Averaged Navier–Stokes (RANS) equations that makes use of a finite volume method. The sampling points from the CFD simulations were gathered for previous research, hence the turbulence model selection, validation, and verification or the generation of the mesh was outside the scope of this project. For more details about the high-fidelity CFD simulations, one is referred to the report of Papp [18].

The mesh that was used for the CFD simulations covers half of the span of the MULDICON as the performed motions are considered to be symmetrical only. The grid of the MULDICON model that is used for collection of the high-fidelity samples of the is shown in Figure 4.11. In total, the grid contains over 12 million cells of which the mesh around the wing covers 112 cells along the chord and 128 along the span. For the scope of this project, one is only interested in flow-wise wing surface pressure distributions, hence for the generation of the high-fidelity samples, the domain around the wing is fully neglected. For each time step, the wing pressure distribution is saved for each vertex in terms of non-dimensional coefficients, e.g., the pressure coefficient and the integral load coefficients.

The data that is gathered from the CFD simulations are stored into a single matrix to simplify computations making it able to obtain the entire flow field into one domain. The upper half of the matrix represents the lower surface of the wing, whereas the lower half of the matrix represents the upper surface of the wing. A representation of the matrix format is shown in Figure 4.10. It must be noted that the domain of the entire wing was halved, resulting in a final grid of 113 x 65 nodes (7345 elements).

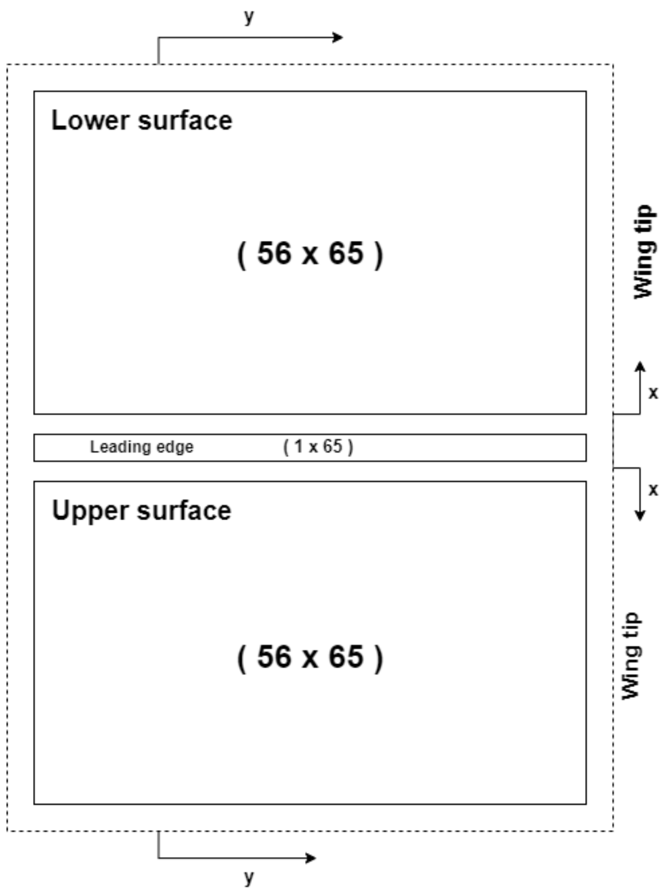
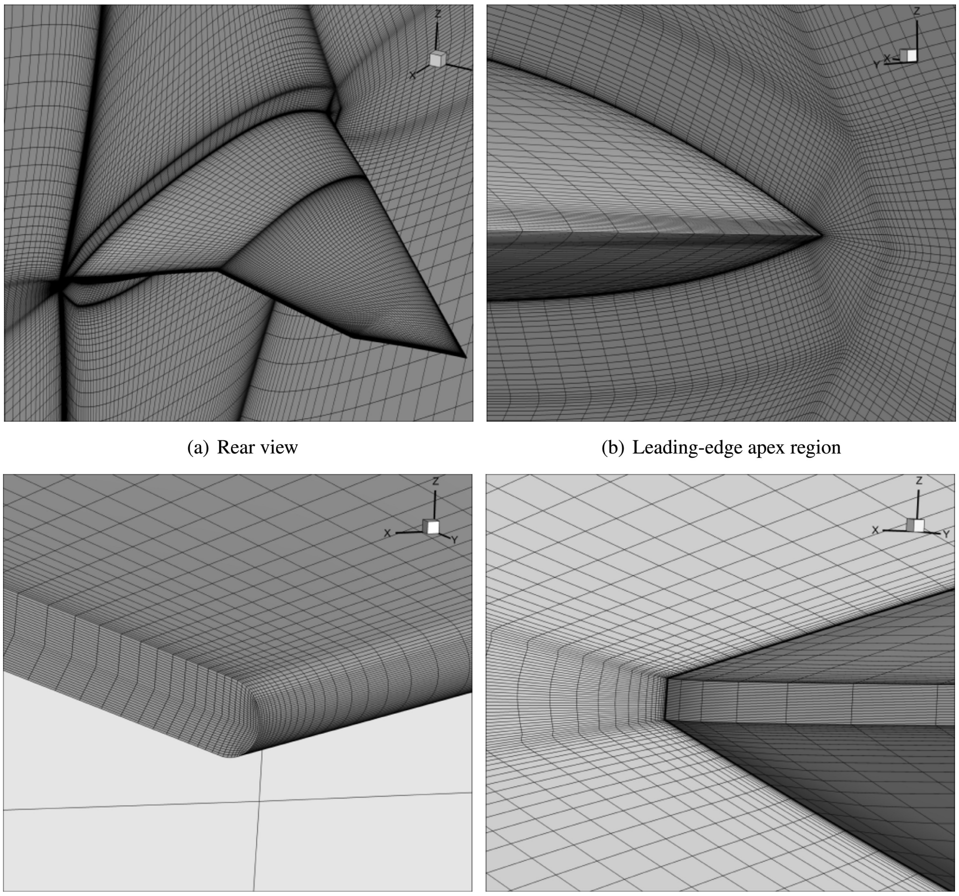


Figure 4.10: Representation of the high-fidelity data in matrix format.



(a) Rear view (b) Leading-edge apex region

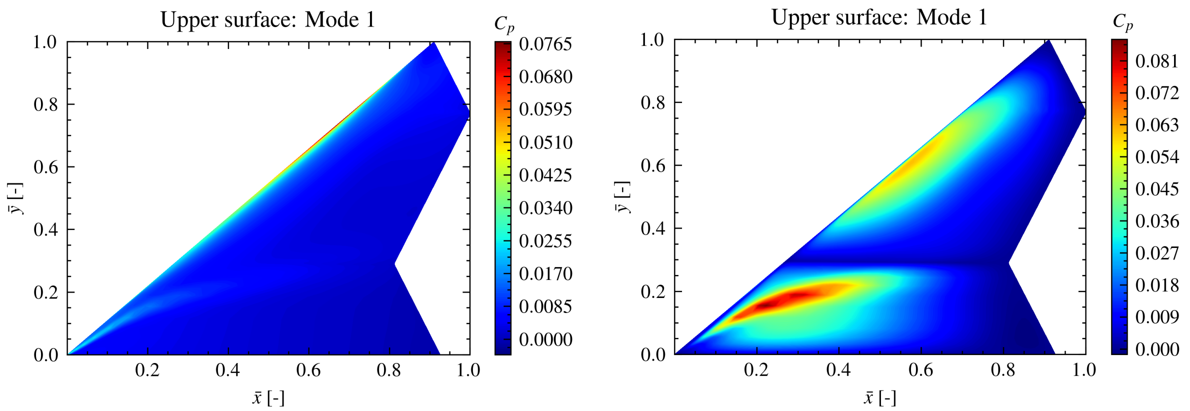
(c) Wing tip leading edge (d) Trailing-edge centerline region

Figure 4.11: The constructed grid used for the RANS simulations for the MULDICON UCAV [16].

4.2.3. Modal Decomposition Method: Proper Orthogonal Decomposition

Model Order Reduction (MOR) approaches are one of the most popular techniques used in the construction of a ROM. These MOR techniques allow constructing a reduced-order basis onto which the full-order simulations are projected, while still maintaining sufficient accuracy. One of the most widely used approaches in MOR is the Proper Orthogonal Decomposition (POD) method, as described in Chapter 2. The POD method is a modal decomposition method that allows reconstructing the full-order model into a subset of orthogonal spatial modes with corresponding time coefficients. One of the benefits of this method is that all the spatial modes are orthogonal, hence unique. Furthermore, the spatial modes are ranked in order of relevance for the contribution of the Total Kinetic Energy (TKE), i.e., the first spatial mode has a higher contribution to the TKE than the second, etc. Therefore, the most dominant characteristics of the flow field can be modeled using only a few spatial modes, allowing to reduce the amount of data necessary tremendously.

For the proposed non-intrusive ROM, the high-fidelity training samples are stored in matrix format as depicted in Figure 4.10. Using this matrix format, the high-fidelity samples are decomposed according to the Proper Orthogonal Decomposition (POD) method as described in Chapter 2. First, the wing surface pressure fluctuation is determined by subtracting the mean of all the high-fidelity samples for each given time step. Second, the spatial POD modes and their corresponding time coefficients are constructed from the training samples. The number of spatial modes that are constructed is dependent on the number of data points in the grid that is used. Therefore, from the high-fidelity training samples a total of 7,345 spatial POD modes are constructed. Finally, the constructed spatial POD modes from training samples are used to determine the time coefficients of the test samples. It must be noted that as the concentration of the number of vertices, due to the structure of the grid, at both the leading edge and the trailing edge is higher than half chord, all the data points of the high-fidelity samples are first multiplied with its surrounding cell areas. Therefore, each data point is multiplied by 4 different cell areas except for the boundaries of the domain. This is done to reconstruct spatial POD modes that will be able to capture the dominant flow phenomena well. A comparison of the spatial POD modes for data with and without cell area multiplication is shown in Figure 4.12. It can be seen that the spatial POD modes that are constructed from data with cell area multiplication are able to cover the leading edge vortices, which were also seen in the flow visualization of the introduction of the MULDICON use case, well. It must be noted that in order to reconstruct the flow field with the reduced-order basis, the spatial POD modes first need to be transformed back to the original data set, i.e., all the spatial POD modes are divided by their cell area.



(a) POD mode without cell area multiplication

(b) POD mode with cell area multiplication

Figure 4.12: Difference between POD mode reconstruction without and with cell area multiplication.

As previously mentioned, one of the benefits of the POD method is that the spatial POD modes are orthogonal, hence unique. According to this property, the spatial POD modes can be ranked in order to the contribution of the TKE. The energy containment is related to the eigenvalue for each mode and can be determined using the following relationship:

$$\%\,TKE = \frac{\lambda_k}{\sum_{k=1}^{\infty} \lambda_k} \tag{4.2}$$

One of the benefits of this property is that the user can set a requirement for the number of modes that are necessary to be used on a reduced-order basis. Generally, the criteria to determine the minimum number of modes is that the reduced-order basis should cover at least 99% of the TKE. Although the TKE is a good measurement to determine the minimum amount of spatial POD modes necessary to obtain the most dominant properties, it is not necessarily a good approach to obtain phenomena that are less frequent, such as non-linear phenomena, e.g., flow separation and turbulent flow. To capture these phenomena, a proper selection of the minimum number of spatial POD needs to be incorporated as criteria for an accurate ROM.To give the reader a general idea of the constructed spatial POD modes from the high-fidelity training samples, the energy containment for each mode up until the 10th mode is shown in Figure 4.13. It can be seen that the only the first 10 spatial POD modes are sufficient to capture more than 98 % of the TKE.

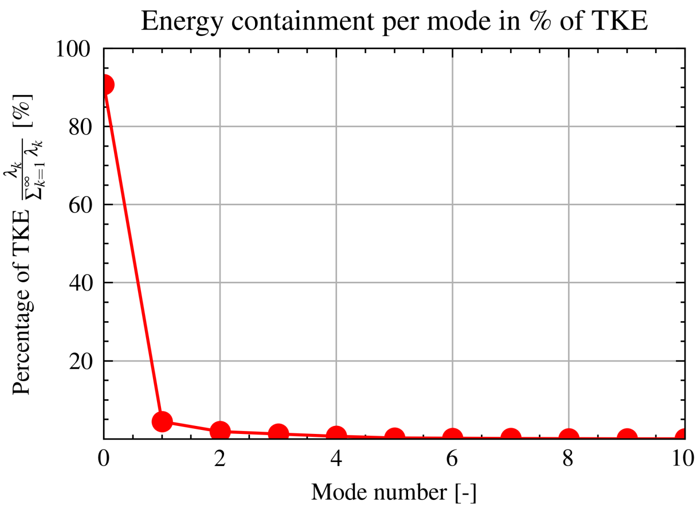


Figure 4.13: Energy containment for each mode in percentage of the TKE.

Next to the contribution of the TKE for each mode, also the amplitudes of the time coefficient for each spatial POD mode are a good measurement to determine how dominant each mode is with respect to the flow field projection. Moreover, as the POD method is a linear method, the higher the amplitude, the higher the contribution of that given spatial POD mode is on the reduced-order basis. In Figure 4.14 the maximum amplitudes of the time coefficient for each mode are shown up until the 50th mode. Also in this figure, it can be seen that the first 10 to 20 modes have large amplitudes and are therefore considered to have a large contribution to the projected flow, whereas after the 20th mode the values of the amplitude decrease significantly.

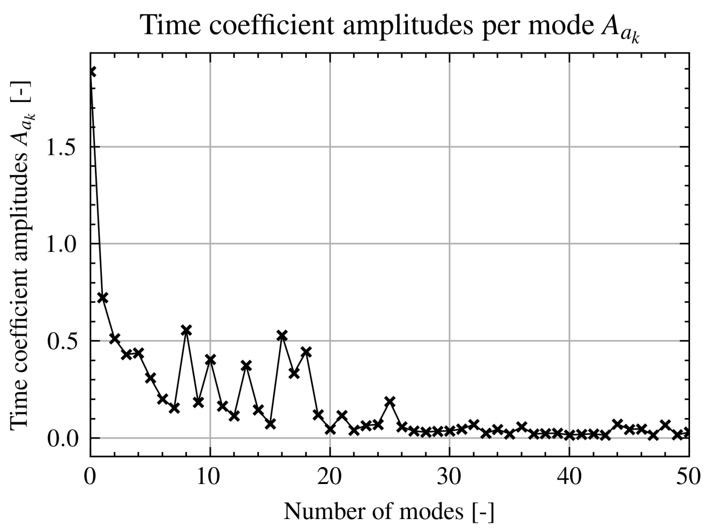


Figure 4.14: Maximum time coefficients amplitudes per mode for the training maneuver.

An example of the constructed spatial POD modes with their corresponding time coefficients for the high-fidelity training samples is shown in Figure 4.15. It must be noted that the time coefficients for both upper and lower surfaces are identical as the spatial POD modes are constructed using the entire flow domain. The spatial POD modes are only split up into the upper and lower surface domain for illustration purposes. A more elaborate example of the spatial POD modes and the corresponding mean of the high-fidelity training samples is shown in Figure 4.16. From the figure, it can be seen that the mean of the flow includes the generation of a small leading-edge vortex, which is due to the nominal value of the input signal which is set to an angle of attack of 12.5 degrees. Furthermore, the figure shows that the first two modes of the upper surface contribute to a large portion of the leading edge vortex development, whereas for the lower surface the spanwise leading edge flow development is highlighted.

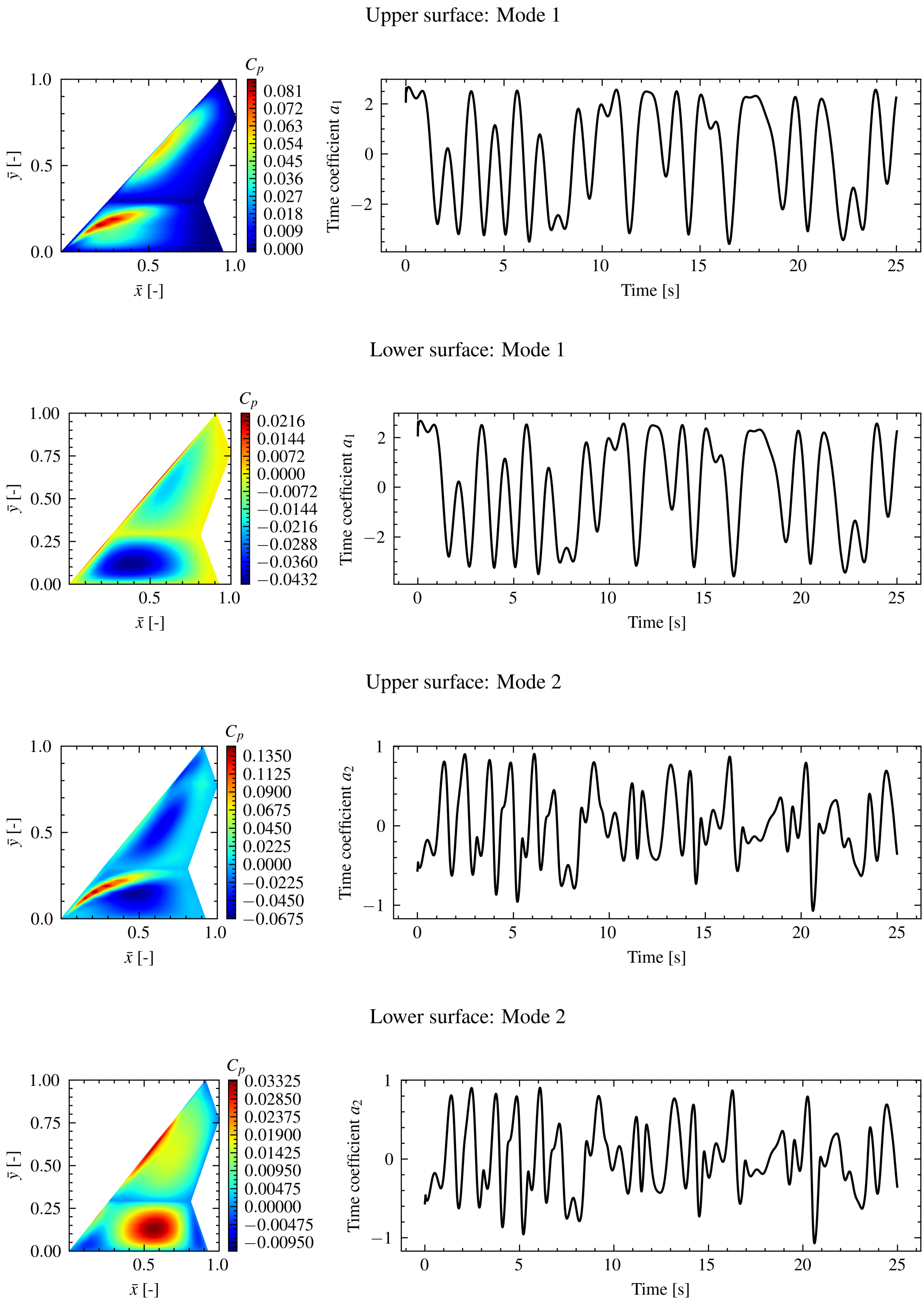


Figure 4.15: Example of the first two constructed POD spatial modes of the MULDICON with its corresponding time coefficients for the given training maneuver.

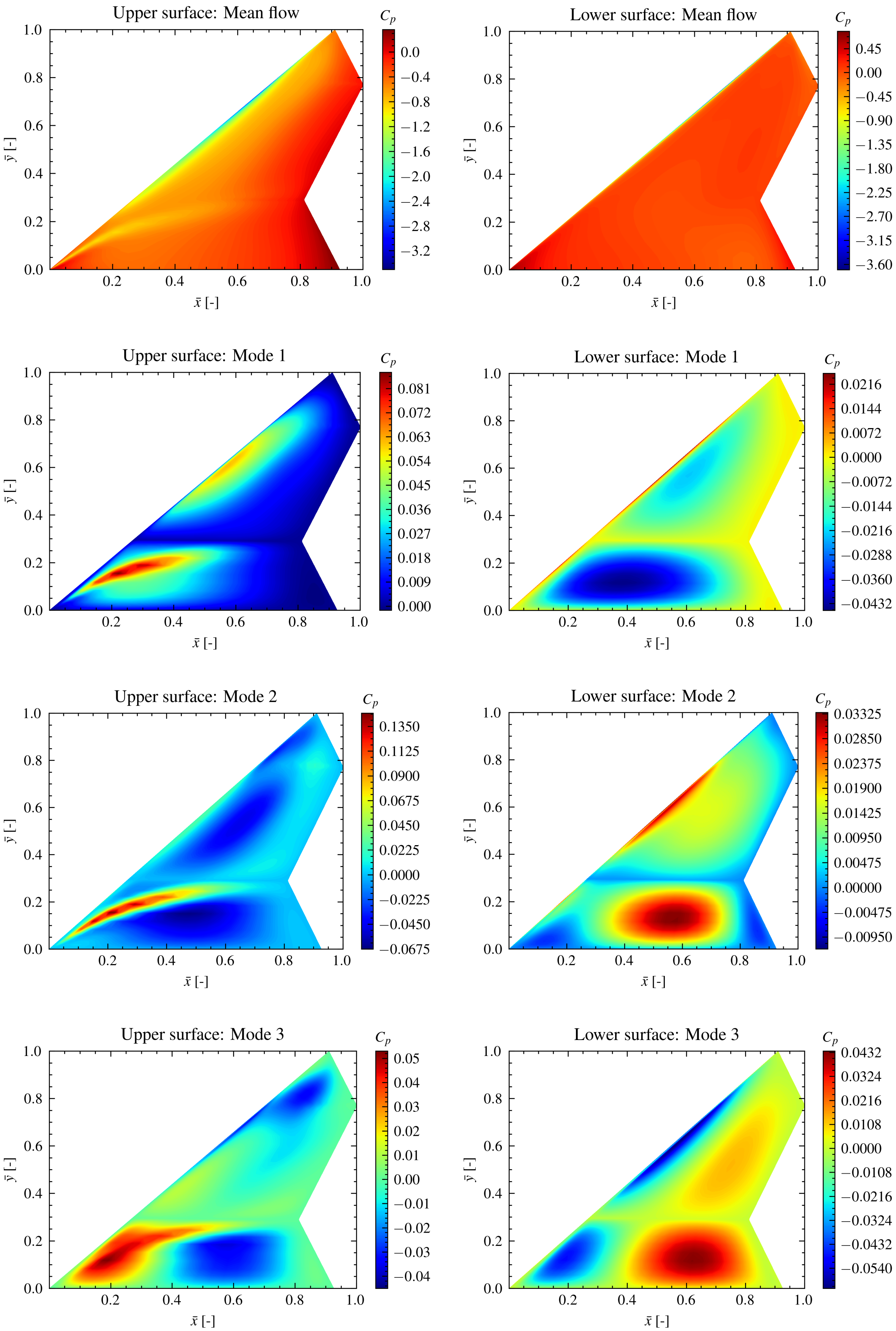


Figure 4.16: POD spatial mode representation of the MULDICON of the given training maneuver obtained from CFD simulations

Calculation of lift, normal force, and moment coefficient

One of the benefits of using the POD method is that the spatial POD modes only need to be constructed once in order to evaluate new high-fidelity samples. This also means that all the properties that are related to these spatial modes, only need to be calculated once. One can use this property to derive the integral aerodynamic loads in terms of the axial and normal force coefficient and the pitching moment coefficient. First, the integral aerodynamic load coefficients are calculated for each cell using the following equations:

$$\Delta F_k = -C_p * A_k * q_\infty * \vec{n}_k \tag{4.3}$$

$$\Delta M_k = \vec{r}_k \times \Delta F_k \tag{4.4}$$

The total integral aerodynamic load coefficients for each mode can then be calculated using the following relationship:

$$C_{F_{mode}} = \begin{bmatrix} C_X & C_Y & C_Z \end{bmatrix}^T = \frac{1}{\frac{1}{2}\rho V^2 S_{ref}} \sum_{k=1}^{N_b} \Delta F_k \tag{4.5}$$

$$C_{M_{mode}} = \begin{bmatrix} C_{M_x} & C_{M_Y} & C_{M_Z} \end{bmatrix}^T = \frac{1}{\frac{1}{2}\rho V^2 S_{ref} l_{ref}} \sum_{k=1}^{N_b} \Delta M_k \tag{4.6}$$

The calculations according to the equations above are repeated for each individual spatial POD mode and stored. The integral load coefficients for the reduced-order basis can be calculated by multiplying the corresponding time coefficients with the corresponding contribution of the integral load coefficients for each spatial POD mode.

4.2.4. Surrogate modeling: the LSTM model

This section will discuss the implementation of the surrogate model that is used for the proposed non-intrusive ROM. As discussed in Chapter 2 and Chapter 3, several surrogate modeling approaches exist in the use of ROMs. The neural network approach that was used for the MULDICON case as described by Papp [18] shows good promise, however as described in Chapter 3 the model was unable to capture time-history effects and the prediction of the wing tip surface pressure for higher angles of attack was inaccurate. Furthermore, as the model did not use a reduced-order basis for the projection of the full-order model, the training time of the neural network still was computational expensive as all the high-fidelity data was considered. This study will elaborate on the neural network approach and will exchange the surrogate model with a different neural network architecture in order to include time-history effects. Furthermore, this study will assess the computational performance of the surrogate model for a varying set of model parameters to determine the effects of the model parameters and highlight parameters of importance for both accuracy and computational expense. First, the architecture of the proposed surrogate model will be discussed. Second, data handling of the surrogate model will be discussed. Third, the model parameters used for the non-intrusive ROM will be discussed. Finally, the evaluation method of the model including verification of the surrogate model will be discussed.

The Long-Short Term Memory (LSTM) network

One of the most popular neural networks that is used for sequential data is the Long Short-Term Memory (LSTM) network. This network, as described in Chapter 3 has been developed to overcome the drawbacks of the previously developed Recurrent Neural Network (RNN). The surrogate model that is used for this study is the many-to-one LSTM network, for which a sequential set of state variables is constructed as input and the desired output of the surrogate model is the set of predicted time coefficients for the number of spatial POD modes. The LSTM model architecture is consisting out of a deep layer network with at least one LSTM layer and a fully connected layer to the output layer. An example of the surrogate model architecture is shown in Figure 4.17.

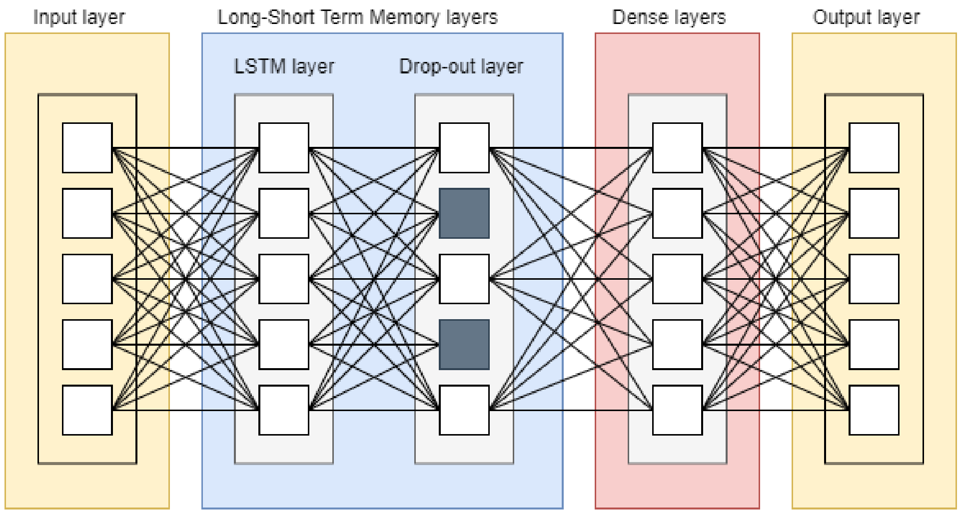


Figure 4.17: An example of the LSTM network architecture.

LSTM model parameters

This section will describe the LSTM model parameters that are used for the training and prediction of the surrogate model. As the LSTM network is constructed using Python and the provided Tensorflow[1] architecture most of the parameters that are used by the model are default settings. However, a few parameters are pre-defined by the user, which are also referred to as hyperparameters. The following hyperparameters are considered:

- **Number of LSTM layers**
  The number of LSTM layers determines the number of consecutive recurrent layers in the model architecture. Each LSTM layer is paired with a drop-out layer. The default is set to 1.
- **Number of LSTM units**
  The number of LSTM units determines the number of LSTM memory cells, as depicted in Figure 3.4, per layer. The default is set to 32.
- **Number of dense layers**
  The number of dense layers determines the number of consecutive dense layers after the LSTM layers. The default is set to 1.
- **Number of dense units**
  The number of dense units determines the number of nodes in a dense layer. The default is set to 32.
- **Number of time-steps used**
  The number of time steps determines the number of previous time steps (excluding the current time-step) that is considered for the prediction of the current time-step. The default is set to 5.
- **Batch size**
  The batch size is a hyperparameter used for training the LSTM model and determines the number of training examples utilized in one iteration. The default is set to 16.
- **Drop-out rate**
  The drop-out rate is a hyperparameter used for training the LSTM model and determines the percentage of nodes that are left out in the training of the layer in one iteration. The default is set to 0.2.
- **Number of spatial POD modes**
  The number of spatial POD modes determines the number of spatial modes that will be used to construct the projection of the full-order model. As the number of spatial POD modes is directly linked to the time coefficient, this hyperparameter will determine the output size of the model. The default is set to 5.

[1] **Tensorflow V2:** `www.tensorflow.org`

- **Model accuracy**
  The model accuracy is a hyperparameter used for training the LSTM model and determines how accurate the surface pressure projection of the training set is with respect to the full-order model. This hyperparameter will determine the number of iterations that are necessary to achieve this accuracy.

Next to the model parameter settings, also the optimization method that is used for training of the LSTM network and the model evaluation in terms of loss function are user-defined settings. The optimization method for training and evaluating the weights within the LSTM network is done using the ADAM method. The ADAM method combines the momentum gradient descent approach and adaptive methods.

Furthermore, in order to evaluate the training process of the LSTM network, the loss function that is given by the user should be minimized. The loss function is a measure of error between the predicted and desired outcome of the model. For the LSTM network, the loss function is defined as the mean-squared error between the predicted and desired outcome and is defined using the following relationship:

$$MSE(y_{predicted}, y_{true}) = \frac{1}{N}\sum_{i=1}^{N} \|y_{predicted} - y_{true}\|^2 \tag{4.7}$$

It must be noted that as the POD-LSTM network output is based on the selection of the maximum amount of spatial POD modes and hence is using a reduced-order basis, the desired outcome, y_{true}, is set to the projection of the full-order solution on the reduced-order basis constructed from the maximum amount of spatial POD modes. Therefore, the error measurement is defined as the error between the projection of the predicted time coefficients on the reduced-order basis and the full-order model projection on the reduced-order basis.

LSTM data handling

In order to enhance the training of the LSTM network architecture, the data is pre-processed before being used by the model. First, all the data, both input, and output, is normalized using a minimum-maximum scaler. The idea behind normalization is that the data is measured at different scales and hence the data does not contribute equally to the training of the model, creating a bias for each variable. In order to avoid this potential problem, the data is normalized prior to training the model. The working principle of the minimum-maximum scaler is that all the data will be scaled to a given range, e.g., from 0 to 1. A relationship for a simplified version of the minimum-maximum scaler is shown below:

$$x_{scaled} = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{4.8}$$

After the data is normalized, the data of the model needs to be split up into different sets of time series in order to be used by the LSTM model. This is done using the sliding window approach, where the input data is split up into time series with a user-defined number of time steps and the corresponding output is defined as the time coefficients corresponding to the to-be calculated time step, with a length of the user-defined number of spatial POD modes.

Model verification

In order to make sure that the model is working correctly, a model verification method is proposed. The method that is proposed is the evaluation of a 1-D signal that is obtained from synthetic data and should represent the upper surface pressure distribution of a pitching airfoil at a single coordinate $C_P(x, y)$. The 1-D input signal consists of the same set of state variables as used in the POD-LSTM ROM. The 1-D output signal is decomposed using the Fourier series method as described below:

$$f(x) = a_0 + \sum_{n=1}^{\infty} a_n cos(nx\frac{\pi}{L}) + \sum_{n=1}^{\infty} b_n sin(nx\frac{\pi}{L}) \tag{4.9}$$

The 1-D output signal is decomposed using the Fourier series in order to represent a similar situation as the usage of the POD method for the LSTM network. The spatial 'POD' modes ϕ_i are constructed using a random number generator and are considered constant, whereas the time coefficients are obtained from the decomposed series, e.g., $a_n \cos(nx\frac{\pi}{L})$ and $b_n \sin(nx\frac{\pi}{L})$, normalized with its corresponding spatial POD constant, i.e., $\frac{a_n cos(\frac{\pi}{L})}{\phi_n}$. As the time coefficients are constructed from sets of sine and cosine signals, the time coefficients are orthogonal and have the same working principle as the POD method holds. An example of the 1-D signal projection is shown in Figure 4.18.

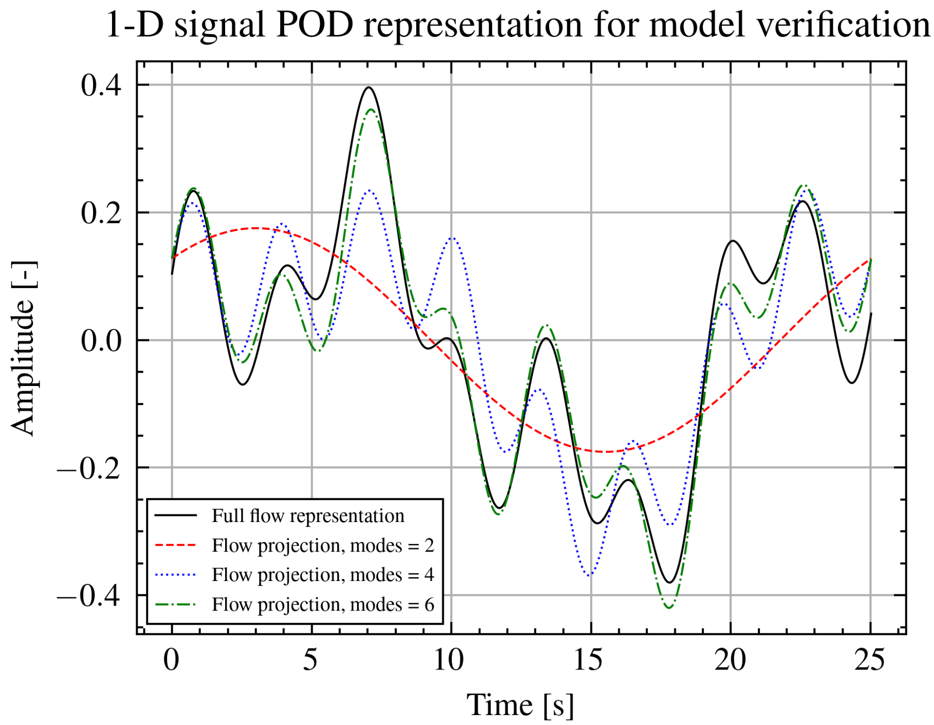


Figure 4.18: 1-D signal POD representation for verification of the LSTM surrogate model.

The model verification method can be used at any time in order to verify the working of the LSTM model and can give the user a general idea about the evaluation of the model performance, e.g., hyperparameter sensitivity and computational cost.

Systematic model evaluation approach

As described in Chapter 3, most of the current ROM approaches for the MULDICON case are derived from state-of-the-art applications. However, for most of these approaches, the variation of model parameters has not been included in the assessment of the ROM, whereas changing the user-defined parameters of a model could have a large influence on the outcome of the constructed ROM in terms of accuracy and computational cost. This section will discuss a systematic approach that is used in order to evaluate the model performance for different sets of model parameters.

The model evaluation approach is split up into two different parts: the model evaluation with respect to its reduced-order basis and the model evaluation of the LSTM network prediction.

- **Model evaluation with respect to the reduced-order basis**
  As one of the properties of the ROM is that it can be used for a reduced-order basis, as constructed from a limited number of spatial POD modes, it must be taken into account in the evaluation of the model performance. In short, the number of spatial POD modes that are used to construct the reduced-order basis is directly linked to the maximum accuracy that is achieved by the model, i.e., the higher the number of spatial POD modes used, the higher the accuracy. In order to remain consistent, it is necessary to evaluate the model performance for a given set of reduced-order basis. For this approach, it is assumed that the number of training samples is sufficient to cover the regressor space well and test samples can be approximated in the same order of magnitude as the training samples. Using this assumption, one can use the training samples as a reference to define a set of reduced-order bases while generalization of the model is maintained. The measurement of error that is used to evaluate the accuracy of the reduced-order basis is defined as the projection error of the training samples: $\boldsymbol{\epsilon}^{\boldsymbol{proj,training}}_{\boldsymbol{C_{p'}}}$. It must be

noted that for the purpose of this study, the projection error of the samples is based on the prediction of the surface pressure coefficients. However, depending on the application of the model, one is not restricted to the evaluation of the projection error of the surface pressure coefficients only but can be extended to the integral aerodynamic load coefficients as well.

- **Model evaluation of the LSTM network**
The model evaluation of the prediction of the POD-LSTM model will determine how well the POD-LSTM model is able to represent the reduced-order basis of the POD method. As earlier described, the model performance is evaluated in terms of the mean-squared error, whereas the loss function is defined as the error measurement between the prediction of the projected flow and the projection of the reduced-order basis. In order to provide a good comparison for different sets of model parameters, the relative accuracy of the model is prescribed in terms of the mean-squared error. The neural network error that is used as a measurement of performance is equal to zero if the POD-LSTM predictions are identical to the reduced-order basis projection of the full-order model.

The model evaluation methods as described above can be summarized into the following measurement errors:

- $\boldsymbol{\epsilon}_{C_{p'}}^{proj,training}$: The instantaneous projection error for the training samples in comparison to the full-order model in terms of MSE.

- $\bar{\boldsymbol{\epsilon}}_{C_{p'}}^{proj,training}$: The time-averaged projection error of the surface pressure coefficient for the training samples in comparison to the full-order model in terms of MSE.

- $\bar{\boldsymbol{\epsilon}}_{C_{p'}}^{NN}$: The time-averaged neural network error of the surface pressure coefficient for the training samples in comparison to the reduced-order basis in terms of MSE.

One of the practical aspects of this model evaluation approach is that the projection error and the neural network error are split, hence errors can easily be identified and conclusions can be drawn easily about the performance of both the offline and online stage of the ROM.

In order to determine the set of projection errors that are used to construct the reduced-order basis, and hence to determine the minimum amount of numbers necessary, the surface pressure coefficient projection error is calculated for an increase in the number of modes. The time-averaged projection error of the surface pressure coefficient of the training samples in decreasing order of magnitude with the minimum required spatial POD modes necessary is shown in Table 4.7, whereas the time-instantaneous projection error is shown in Figure 4.19. It can be seen that for a small time-averaged projection error of the surface pressure coefficients, only a small portion of the total number of modes is necessary. These results show that the use POD method has some great potential for Model Order Reduction (MOR). However, it must be noted that it is assumed that the training samples cover the regressor space well enough to produce a well generalized ROM, whereas in practice this may not be the case.

Table 4.7: Minimum number of spatial POD modes that is required for a given time-averaged projection error $\bar{\epsilon}_{C_{p'}}^{proj,training}$ of the training maneuver. The number of spatial POD modes necessary are depicted as percentage of the total number of modes, which is equal to 7,345 modes.

Time-averaged projection error $\bar{\boldsymbol{\epsilon}}_{C_{p'}}^{proj,training}$	Minimum number of spatial POD modes
1E-1	5 (0.07% of total number of modes)
1E-2	40 (0.5% of total number of modes)
1E-3	120 (1.6% of total number of modes)
1E-4	310 (4.2% of total number of modes)

Table 4.8: Levels of time-averaged projection error $\bar{\epsilon}^{proj,training}_{C_{p'}}$ with the corresponding neural network error $\bar{\epsilon}^{NN}_{C_{p'}}$ used for computational cost analysis.

Time-averaged projection error $\bar{\epsilon}^{proj,training}_{C_{p'}}$	Time-averaged neural network error $\bar{\epsilon}^{NN}_{C_{p'}}$
1E-1 (5 modes)	1E-2
	5E-3
1E-2 (40 modes)	1E-2
	5E-3

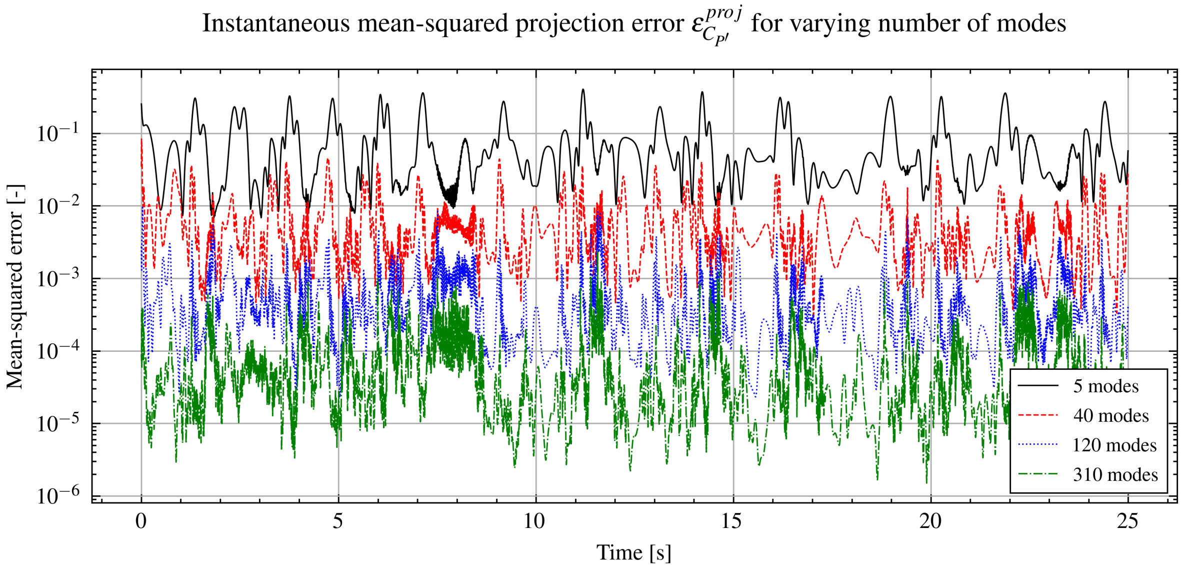


Figure 4.19: The instantaneous projection error $\epsilon^{proj}_{C_{p'}}$ of the training maneuver with respect to the full-order model for increasing the number of spatial POD modes.

For the systematic model evaluation approach, which is used to assess the model performance in terms of cost and accuracy for model parameter changes, four different POD-lstm networks will be used. The POD-lstm networks vary in terms of the projection error and the neural network error in terms of the time-averaged MSE of the surface pressure of the training samples. The model settings that will be used for the approach are shown in Table 4.8.

Baseline model: Sensitivity analysis

In order to obtain a comparable measurement of the model performance with the given model evaluation method settings as shown in Table 4.8, a baseline model is constructed that will be used for the prediction of the surface pressure coefficients. This baseline model is based on the outcome of a sensitivity analysis that is performed for different model parameters as shown in Figure 4.20. The sensitivity analysis is performed for a surface pressure projection error of $\bar{\epsilon}^{proj,training}_{C_{p'}}$ = 1E-1, corresponding to a total of 5 spatial POD modes used, with a maximum number of iterations equal to 500. The sensitivity analysis is performed over both training and test samples and the distribution of the cases is represented in the violin plot. The baseline model parameters are determined from the sensitivity analysis, selecting the model parameters that result in the lowest average mean-squared error of all cases where the variance is small. The baseline model parameters are shown in Table 4.9.

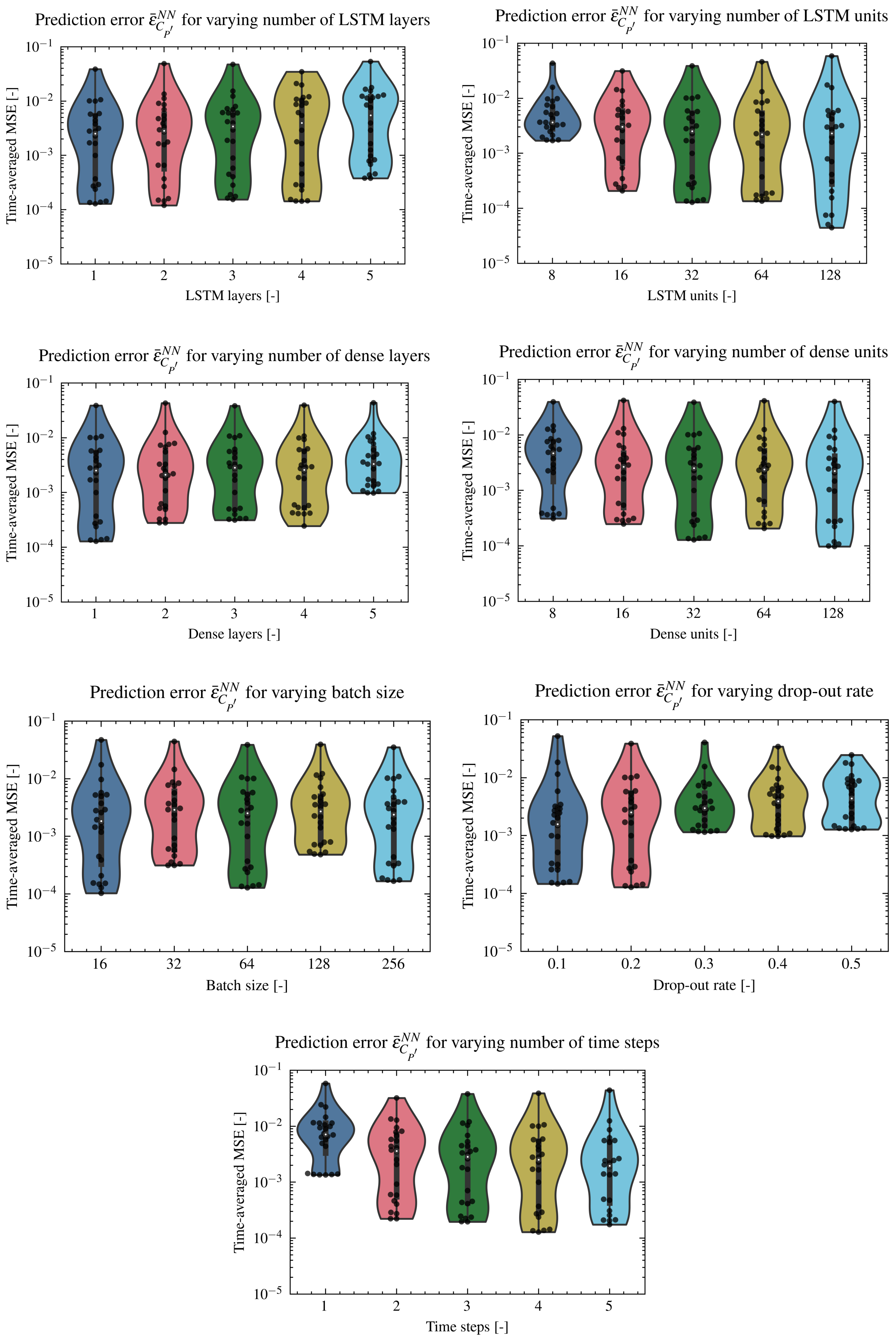


Figure 4.20: Sensitivity analysis shown in violin plots of the LSTM surrogate model for different hyperparameters for selected projection error $\bar{\epsilon}^{proj,training}_{C_{p'}}$ = 1E-1. The dots represent the time-averaged MSE for the test cases as described in Section 4.2.1.

Table 4.9: LSTM surrogate model baseline properties.

Hyperparameters	**Value**
Number of LSTM layers [-]	1
Number of LSTM units [-]	32
Number of dense layers [-]	1
Number of dense units [-]	128
Number of time-steps used [-]	5
Batch size [-]	16
Drop-out rate [-]	0.2
Model optimization method [-]	ADAM
Model loss function [-]	MSE

5

Simulation results of the POD-LSTM non-intrusive Reduced-Order Model

This chapter will discuss the results of the proposed non-intrusive POD-LSTM ROM. The results presented in this chapter can be divided into two different sections: the results of the offline stage and the results of the online stage. First, the offline stage results will be discussed including the projection error for an increasing number of spatial POD modes and the training results of the model. Second, the online stage results will be discussed including the model performance over the regressor space as well as the computational performance. Finally, a brief summary of the results will be provided.

5.1. Offline stage results

This section will discuss the results of the offline stage, as described in Chapter 4. First, the influence on the variation of the number of spatial POD modes and the variation of the spatial POD modes on the projection error for all cases will be discussed. Finally, the results of training the POD-LSTM ROM will be discussed.

5.1.1. POD Projection error

The projection error is described as the error between the projection of the full order model on the reduced-order basis, which is constructed from a limited number of spatial POD modes, and the full order model in terms of the mean-squared error. Next to the contribution of the TKE of each spatial POD mode, the projection error is a measure to determine the minimum number of spatial POD modes necessary in order to reconstruct the full order model with a given accuracy. In this chapter the results are shown for the projection error over all test samples for an increase in the number of spatial POD modes and a comparison is made between the use of different spatial POD modes.

Variation in the number of spatial POD modes

In order to assess the effectiveness of the POD method and the use of the reduced basis that is constructed from the training samples, the projection error has been calculated for all the test samples for an increase in the number of spatial POD modes. Depending on the application, one is interested in the projection error of the surface pressure coefficient or the integral aerodynamic loads, e.g., the normal force coefficient or the pitching moment coefficient. The purpose of this section is to determine if the training samples cover the regressor space sufficiently in order to construct a reduced-order basis that can be used for the prediction of the test case samples.

The results for the projection error of the surface pressure coefficient, the normal force coefficient, and the pitching moment coefficient for an increase in the number of spatial POD modes are shown in Figure 5.1, Figure 5.2 and Figure 5.3, respectively. From Figure 5.1 it can be seen that for an increase in the number of spatial POD modes used for construction of the reduced-order basis, the projection error of the training maneuver, as expected, decreases with an increase in the number of spatial POD modes used. In general, the same trend holds for most of the test samples, however, it can be seen that the test samples have a larger projection error than the training maneuver. Furthermore, it can be seen that for a limited amount of test samples the projection error increases with an increase in the number of spatial POD modes used. The same trends can be seen for the projection error of the normal force coefficient and the pitching moment coefficient, where the projection error is at least two orders of magnitude smaller. The test samples that are resulting in a large projection error are similar for both the projection of the surface pressure coefficient and the integral load coefficients. Overall, the figures are in agreement with the hypothesis that the projection error of the training maneuver is in the same orders of magnitude as the test samples, if the regressor space is covered well.

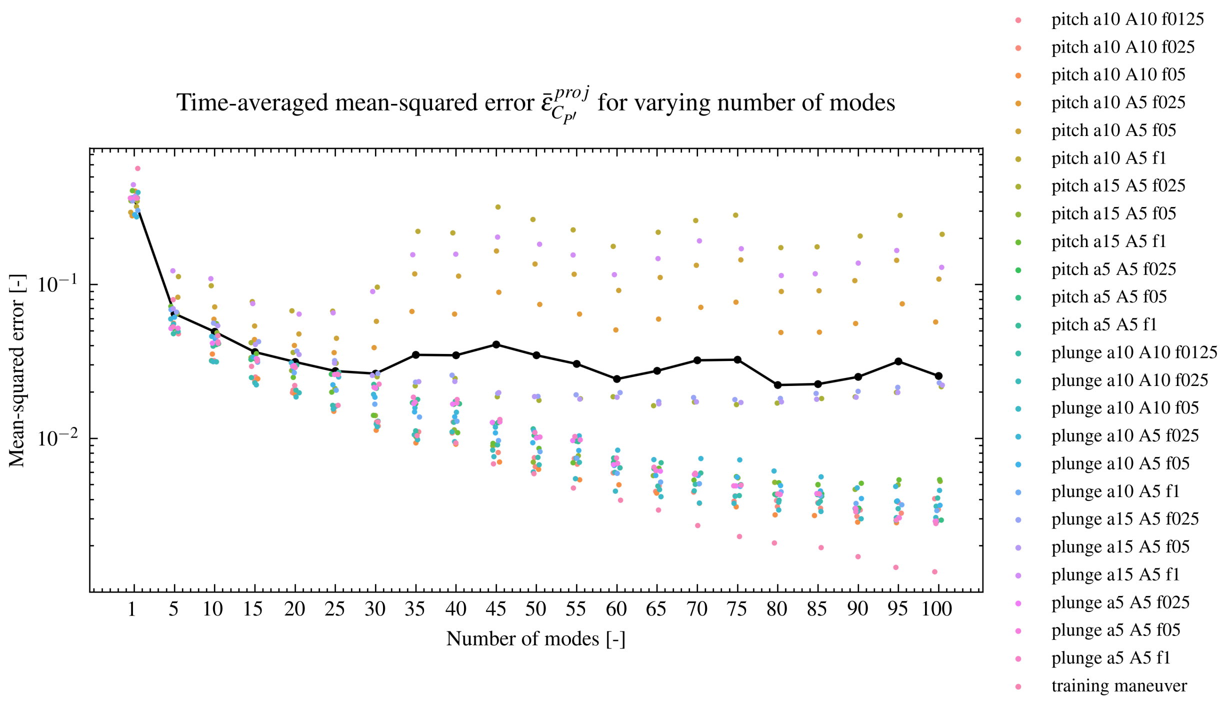


Figure 5.1: Time-averaged projection error $\bar{\varepsilon}^{proj}$ of the pressure coefficient for an increasing number of spatial POD modes in terms of average MSE over all test cases.

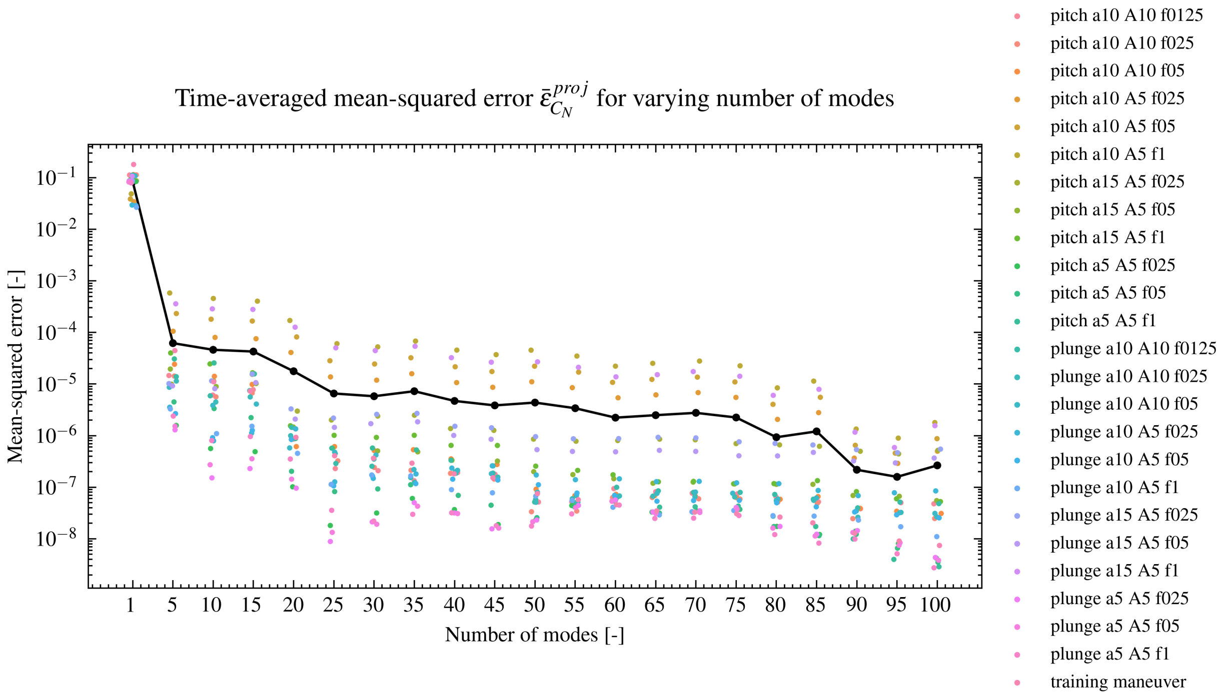


Figure 5.2: Time-averaged projection error $\bar{\epsilon}^{proj}$ of the normal force coefficient for an increasing number of spatial POD modes in terms of average MSE over all test cases.

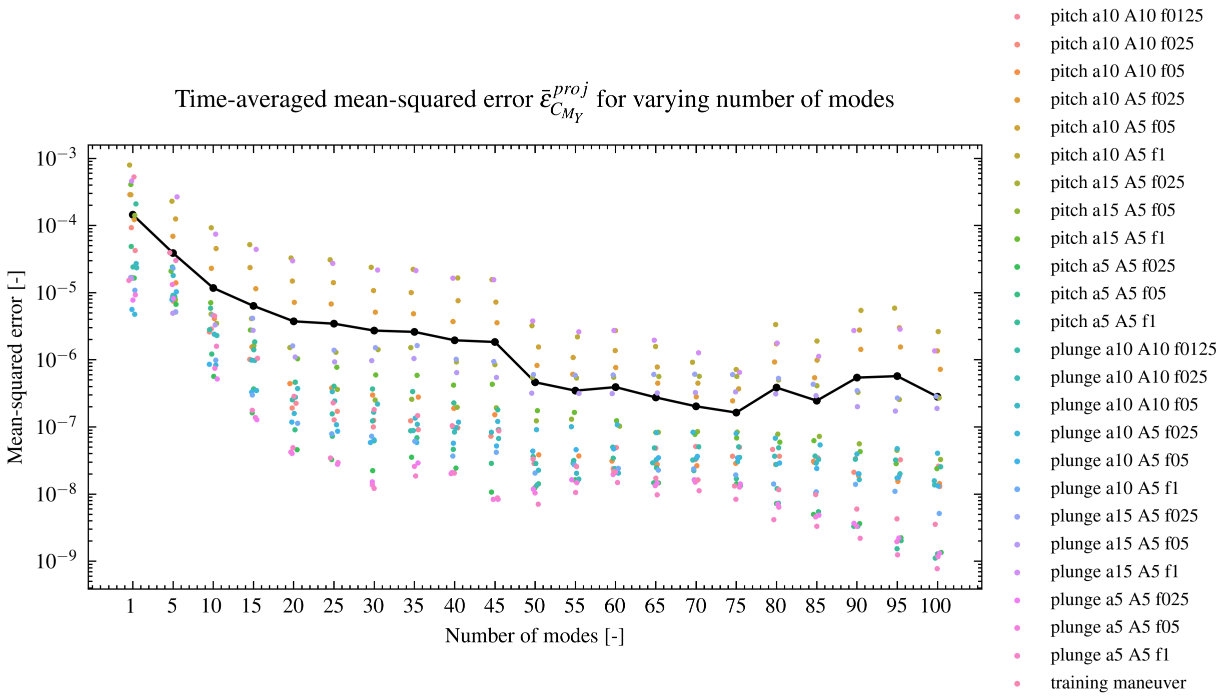


Figure 5.3: Time-averaged projection error $\bar{\epsilon}^{proj}$ of the pitch moment coefficient for an increasing number of spatial POD modes in terms of average MSE over all test cases.

In order to identify the limitations of the current training samples, the test samples that are resulting in the smallest and largest projection error are investigated and compared with the regressor space coverage of the training samples. The test samples that are selected for the comparison are obtained from the average surface pressure coefficient projection error of the training maneuver and are set to the standards as discussed in Chapter 4, $\bar{\epsilon}_{C_{p'}}^{proj,training}$ = 1E-1 and $\bar{\epsilon}_{C_{p'}}^{proj,training}$ = 1E-2, which are translated to a minimum number of spatial POD modes of 5 and 40, respectively. The largest projection errors are shown for test samples that are originated from the harmonic pitch oscillations with a nominal angle of attack value of 10 degrees with an amplitude of 5 degrees, whereas the plunge motions with the same values for the nominal angle of attack and amplitude are found to be at the lower range of errors. The comparison of the regressor space coverage for the training and the resulting smallest and largest projection error test samples is shown in Figure 5.4. Furthermore, the regressor space coverage for the most dominant state variables is shown in Figure 5.5 and Figure 5.6, respectively. From the figures, it can be seen that the training samples are concentrated near the boundaries of the signal and the coverage of the state variables within the smaller range of amplitudes is limited. As expected, the test samples with the highest projection error are in the same region where the coverage is limited.

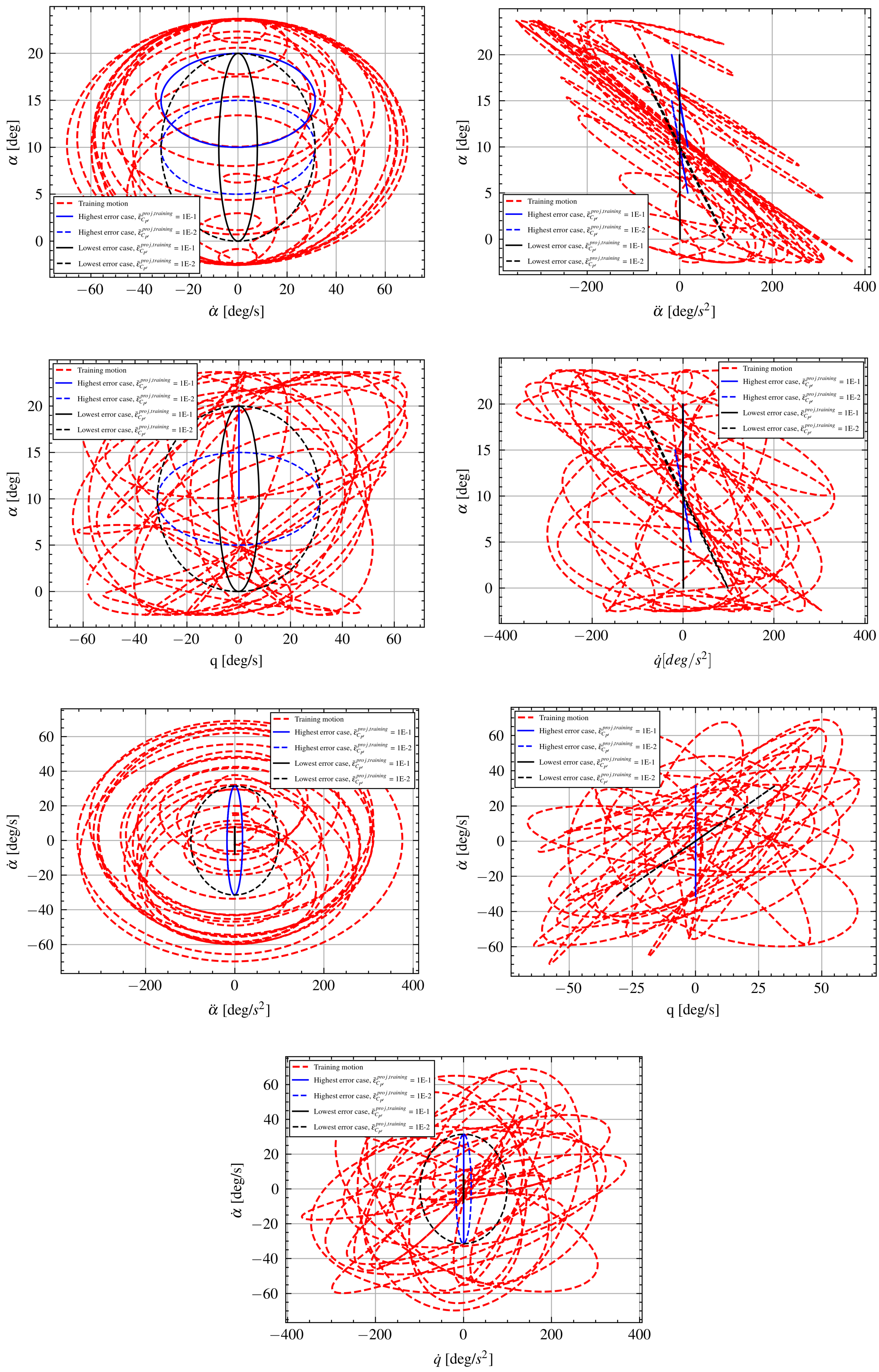


Figure 5.4: Regressor coverage of the training maneuver and the best and worst performing test cases with respect to the projection errors $\bar{\epsilon}^{proj,training}$ = 1E-1 and $\bar{\epsilon}^{proj,training}$ = 1E-2.

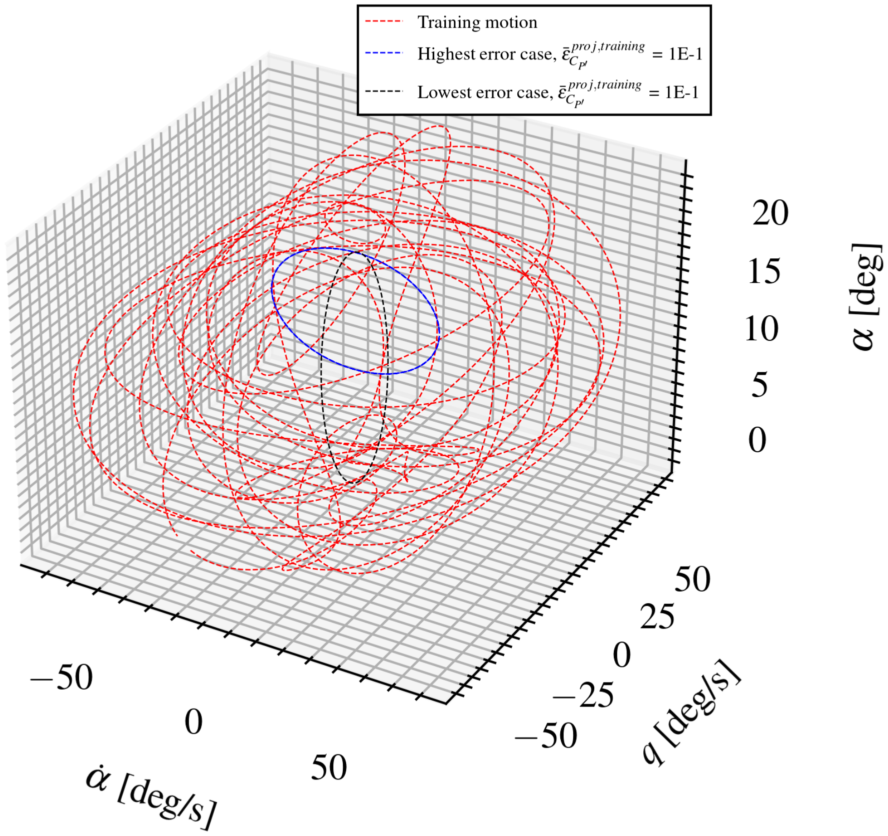


Figure 5.5: Regressor coverage of the training maneuver and the best and worst-performing test cases with respect to a projection error of $\bar{\epsilon}^{proj,training}$ = 1E-1, 5 modes, respectively.

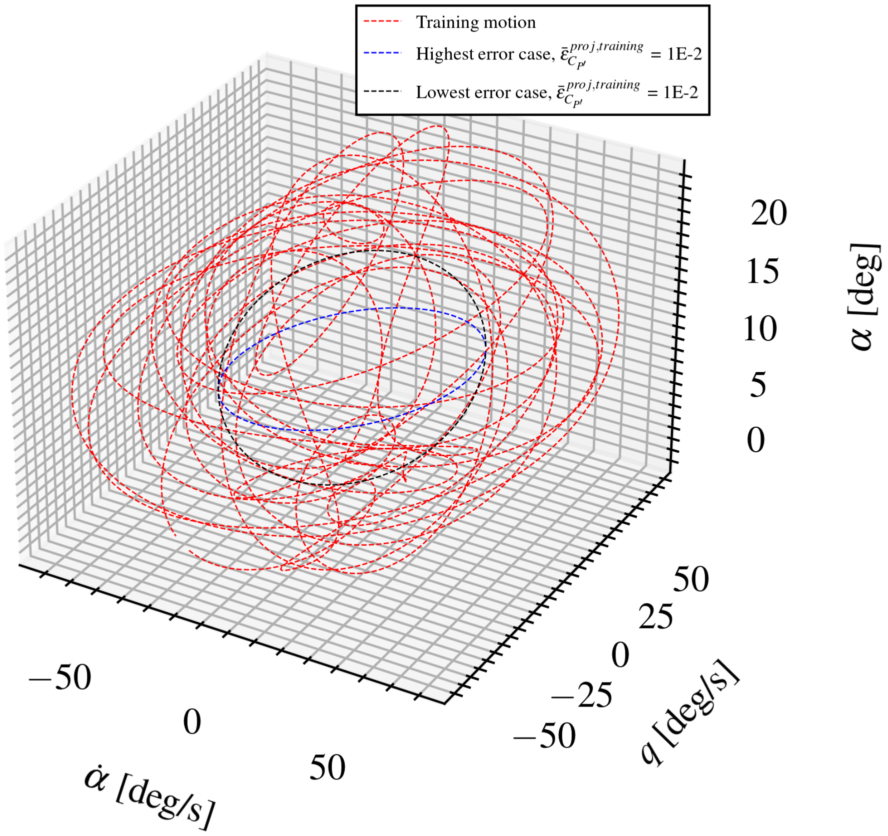
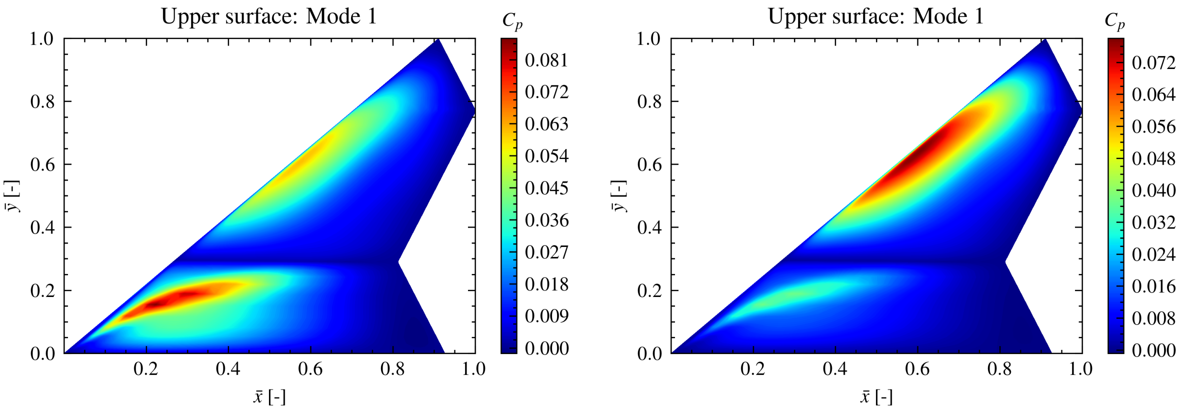


Figure 5.6: Regressor coverage of the training maneuver and the best and worst-performing test cases with respect to a projection error of $\bar{\epsilon}^{proj,training}$ = 1E-2, 40 modes, respectively.

Variation of spatial POD modes

Next to varying the number of spatial POD modes to construct a reduced-order basis, one can also make use of a variation of the individual spatial POD modes while the number of spatial POD modes that are used remains the same. The motivation to alter the individual spatial POD modes is to investigate if the projection error of the test samples can be minimized while the number of spatial POD modes that are used for the construction of the reduced-order basis can be limited. For example, as described by Papp [18], the described model had difficulties in predicting the wing tip surface pressure coefficients for higher angles of attack. Hence, one could focus on the design of spatial POD modes that put an emphasis on the tip region, with the expectation that this will reduce the number of spatial POD modes that are necessary to construct a reduced-order basis.

In Figure 5.7 an example of a variation in the individual spatial POD modes is shown, wherein this case an emphasis is put on the wing tip region to reduce the projection error near this region while the number of spatial POD modes that is necessary to construct this basis can be relatively small. From this, it can be seen that the vortices that are created at the leading edge in the altered POD basis are less dominant and the region of the wing at larger span has shown an increase in the area of interest. The results of the projection error for the original and the newly constructed spatial POD basis are shown in Figure 5.7 for prediction of the surface pressure coefficients, the normal force coefficient, and the pitching moment coefficient, where the results are presented as the average of all test samples over time in terms of MSE. It can be seen from the figure that the projection of the new spatial POD basis shows a decrease in terms of the projection error for both the surface pressure coefficient and the pitching moment coefficient, whereas for the normal force coefficient it actually shows an increase with respect to the original basis. From this figure, it can be concluded that the tip region is an area of interest for the determination of the pitching moment coefficient, i.e., the flow near the tip has a high influence on the error of the pitching moment coefficient. However, it can be concluded that a better projection of the flow towards the root of the MULDICON is of interest for the projection error of the normal force coefficient.



(a) Original spatial POD mode.

(b) Spatial POD mode with a linear weighted relationship along the span.

Figure 5.7: Differences between the original spatial POD modes and altered spatial POD modes with a linear-relationship towards the tip location for the first POD spatial mode of the upper surface.

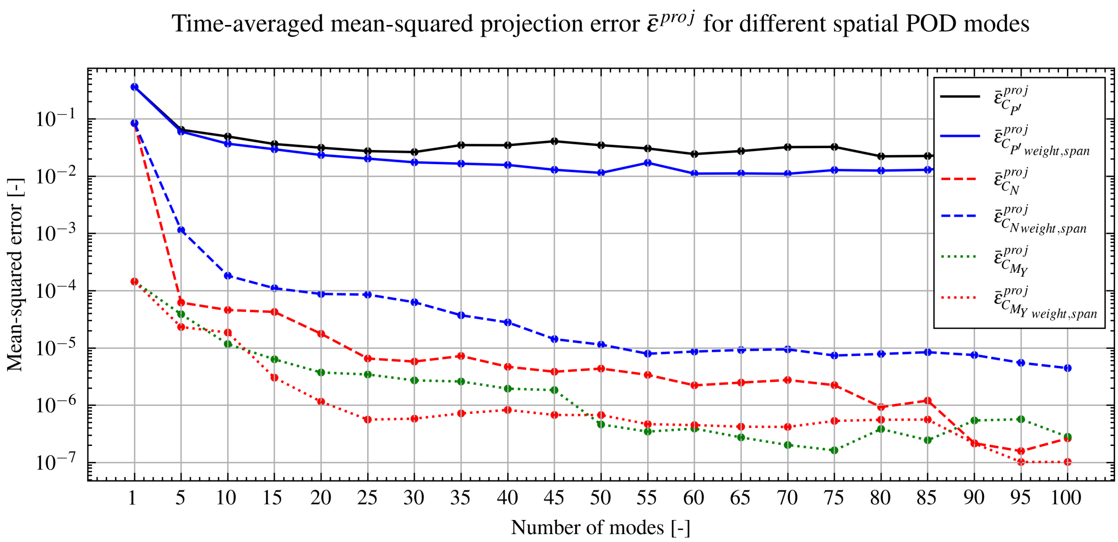


Figure 5.8: Time-averaged projection error $\bar{\varepsilon}^{proj}$ for variation of spatial POD modes in terms of average MSE over all test cases.

5.1.2. Limitations on the use of a reduced-order basis

As the Proper Orthogonal Decomposition (POD) method makes use of a reduced-order basis that is used to approximate the full-order solution, this method imposes an error that is directly related to the number of spatial POD modes that is used to construct the reduced-order basis. In other words, compromises have to be made between the amount of data that is used to construct the reduced-order basis and the error with respect to the full-order solution that is imposed by the use of this reduced-order basis. As already discussed in the previous sections, it was shown that the reduced-order basis that is constructed from the spatial POD modes obtained from the high-fidelity training samples shows good accuracy of the full-order solution with a limited number of spatial POD modes necessary. However, in order to determine the limitations of the use of the reduced-order basis a detailed investigation of the projection of the full-order solution is necessary. For this, one of the test samples has been investigated to identify any deficiencies in the use of the current reduced-order basis. In Figure 5.9 the input variables and the projection error of the surface pressure coefficients are shown for the pitch oscillation with $A_0 = 10$ deg , $A = 5$ deg and $f = 0.25$ Hz. This pitch oscillation was chosen as it has shown the largest relative projection error with respect to other test samples.

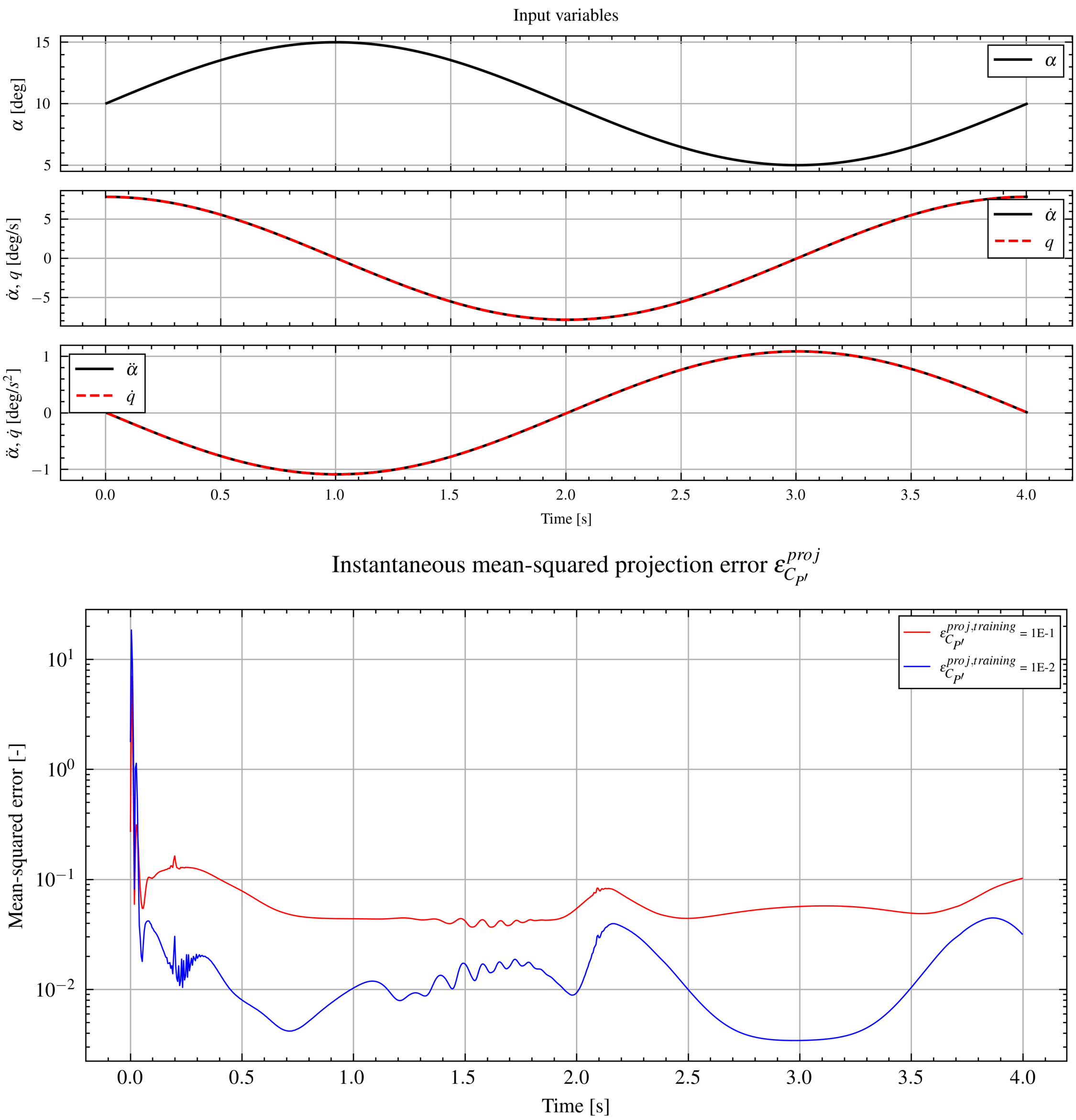
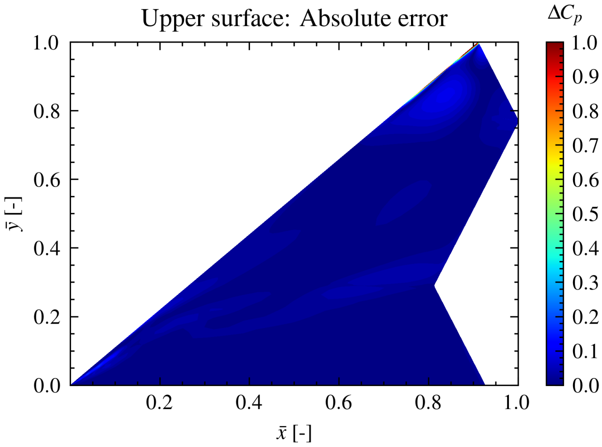


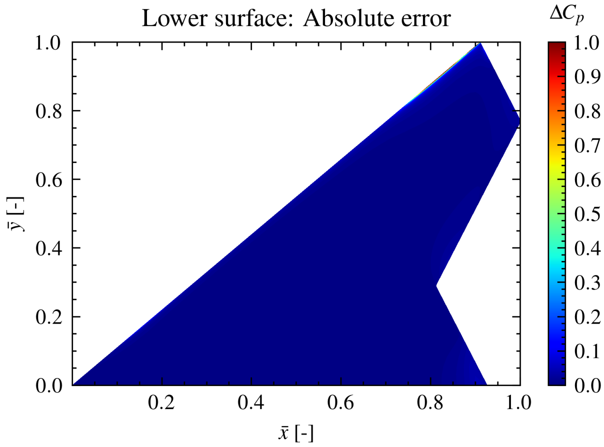
Figure 5.9: POD-LSTM input state variables for pitch oscillation A_0 = 10 deg , A = 5 deg, f = 0.25 Hz.

It can be seen from the figure that the average projection error of the test sample is in agreement with the threshold that was set for the construction of the reduced-order basis using the high-fidelity training samples. It can be seen that for both thresholds, the average projection error is in the same order of magnitude or even lower. This means that the spatial POD modes that are constructed from the high-fidelity training samples serve as a good basis for the prediction of unseen samples, e.g., the high-fidelity test samples. Hence, from this, it can be concluded that the high-fidelity training samples cover the regressor space well.

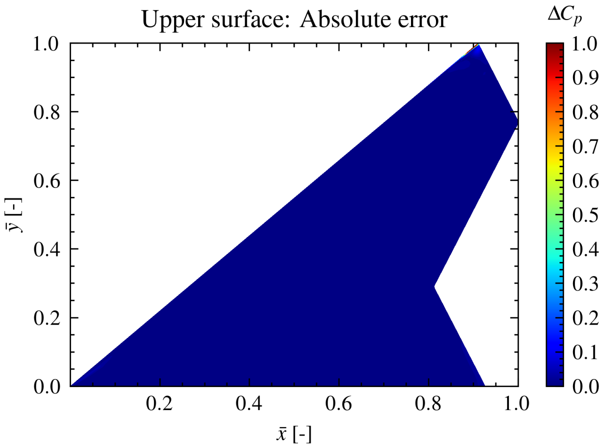
In Figure 5.10 the absolute projection error with respect to the full-order solution is shown for different projection error thresholds. It can be seen that for both reduced-order bases the absolute error is low and only large errors are shown at the leading edge towards the tip region, although this region is very small.



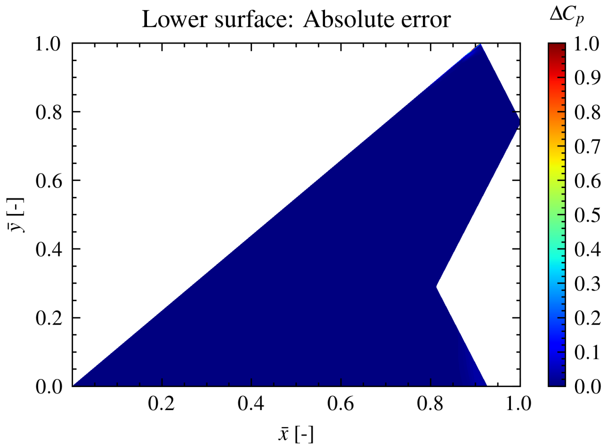
(a) Absolute projection error of the upper surface at $t = 2.2$ s for a reduced-order basis with $\bar{\epsilon}_{C_{p'}}^{proj,training} = $ 1E-1



(b) Absolute projection error of the lower surface at $t = 2.2$ s for a reduced-order basis with $\bar{\epsilon}_{C_{p'}}^{proj,training} = $ 1E-1



(c) Absolute projection error of the upper surface at $t = 2.2$ s for a reduced-order basis with $\bar{\epsilon}_{C_{p'}}^{proj,training} = $ 1E-2



(d) Absolute projection error of the lower surface at $t = 2.2$ s for a reduced-order basis with $\bar{\epsilon}_{C_{p'}}^{proj,training} = $ 1E-2.

Figure 5.10: Comparison of the absolute projection error for pitch oscillation $A_0 = 10$ deg, $A = 5$ deg, $f = 0.25$ Hz at $t = 2.2$ s for reduced-order basis with an accuracy of $\bar{\epsilon}_{C_{p'}}^{proj,training} = $ 1E-1 and $\bar{\epsilon}_{C_{p'}}^{proj,training} = $ 1E-2, respectively.

Overall, it is shown that the reduced-order basis that has been constructed from the high-fidelity training samples provides a good basis for the prediction of the test samples. Hence, the limitations of the reduced-order basis are therefore negligible.

5.1.3. POD-LSTM model training

This section will discuss the model evaluation of the training process for different settings of the POD-LSTM network. The training process has been evaluated in terms of the computational time and the computational count estimate for the variation of the number of LSTM layers, LSTM units, dense layers, dense units, the number of time steps used, and the batch size.

The computational time has been measured as the total time it takes to train the POD-LSTM model for a given accuracy, using the current processor time of the training process only. Furthermore, the training operation count estimate is measured, which is defined as the total number of iterations that were necessary to train the model times the number of trainable parameters that the model includes. The operation count estimate can be used as a reference for the computational time, as the same trend is expected, and can be used as a comparison between the model complexity and the computational time. In order to avoid large variation of the computational processor time, mainly due to running background processes and different weight initialization of the surrogate model, the simulations are run 5 times each. The results for the computational time and the operation count estimate as shown in this section are the averaged values over the 5 runs for each model setting. In total, 29 different model settings were run for each level of accuracy and time-averaged projection error resulting in a total of 580 simulations. In Table 5.1 the different model settings for the runs are shown.

Table 5.1: Model settings that are used for evaluation of the computational training time and operational count estimate for each level of accuracy and projection error.

Number of LSTM layers	Number of LSTM units	Number of dense layers	Number of dense units	Drop-out layer	Drop-out rate	Batch size	Time steps past
1	**32**	**1**	**128**	**TRUE**	**0.2**	**16**	**5**
2	32	1	128	TRUE	0.2	16	5
3	32	1	128	TRUE	0.2	16	5
4	32	1	128	TRUE	0.2	16	5
5	32	1	128	TRUE	0.2	16	5
1	16	1	128	TRUE	0.2	16	5
1	64	1	128	TRUE	0.2	16	5
1	128	1	128	TRUE	0.2	16	5
1	256	1	128	TRUE	0.2	16	5
1	32	0	128	TRUE	0.2	16	5
1	32	2	128	TRUE	0.2	16	5
1	32	3	128	TRUE	0.2	16	5
1	32	4	128	TRUE	0.2	16	5
1	32	1	32	TRUE	0.2	16	5
1	32	1	64	TRUE	0.2	16	5
1	32	1	256	TRUE	0.2	16	5
1	32	1	512	TRUE	0.2	16	5
1	32	1	128	TRUE	0.1	16	5
1	32	1	128	TRUE	0.3	16	5
1	32	1	128	TRUE	0.4	16	5
1	32	1	128	TRUE	0.5	16	5
1	32	1	128	TRUE	0.2	4	5
1	32	1	128	TRUE	0.2	8	5
1	32	1	128	TRUE	0.2	32	5
1	32	1	128	TRUE	0.2	64	5
1	32	1	128	TRUE	0.2	16	3
1	32	1	128	TRUE	0.2	16	4
1	32	1	128	TRUE	0.2	16	6
1	32	1	128	TRUE	0.2	16	7

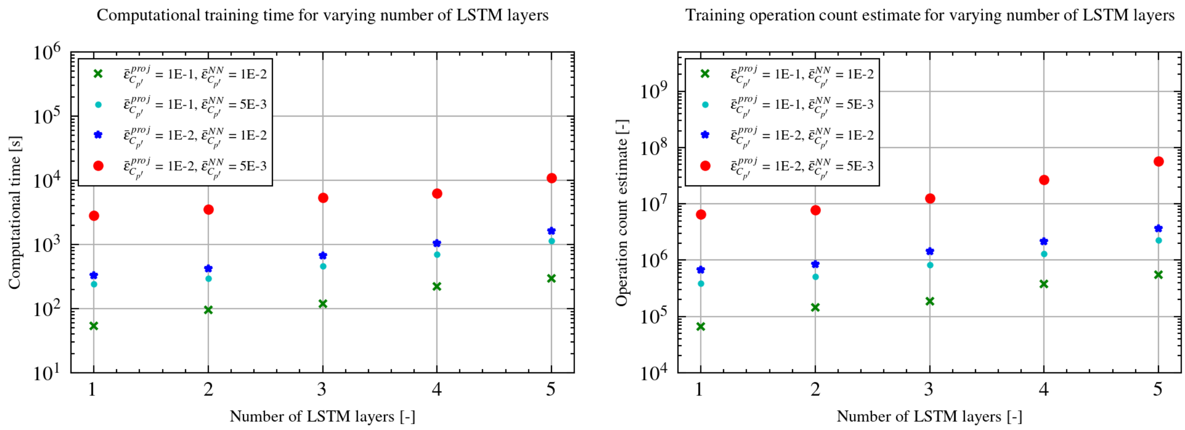
All the simulations were performed using a personal laptop running on the CPU only, whereas each simulation was run on one core. All the training and test samples are gathered from the in-house flow solver of the NLR, ENSOLV, which was run on the NLR cluster consisting out of 80 threads. The specifications are found in Table 5.2.

Table 5.2: Specifications for model runs and high fidelity simulations

	ENSOLV	**Surrogate model**
Platform (CPU)	NLR Computing Cluster @ 80 Threads	Intel Core i7-4700MQ CPU @ 2.40GHz

Variation of LSTM layers

For the number of LSTM layers, the results of the computational training time and the operation count estimate are shown in Figure 5.11. It can be seen that as the number of LSTM layers in the model is increased, the computational training time is increased as well, which is in agreement with the operation count estimate. Furthermore, it can be seen that although the number of LSTM layers is increased, the computational training time for higher accuracy models, i.e., lower projection and neural network errors, also is increased. From this, it can be concluded that the LSTM layers do not have a large influence on the accuracy of the model. The overall results for the given levels of accuracy and variation in the number of LSTM layers are shown in Table 5.3.



(a) Computational training time in seconds.

(b) Operation count estimate.

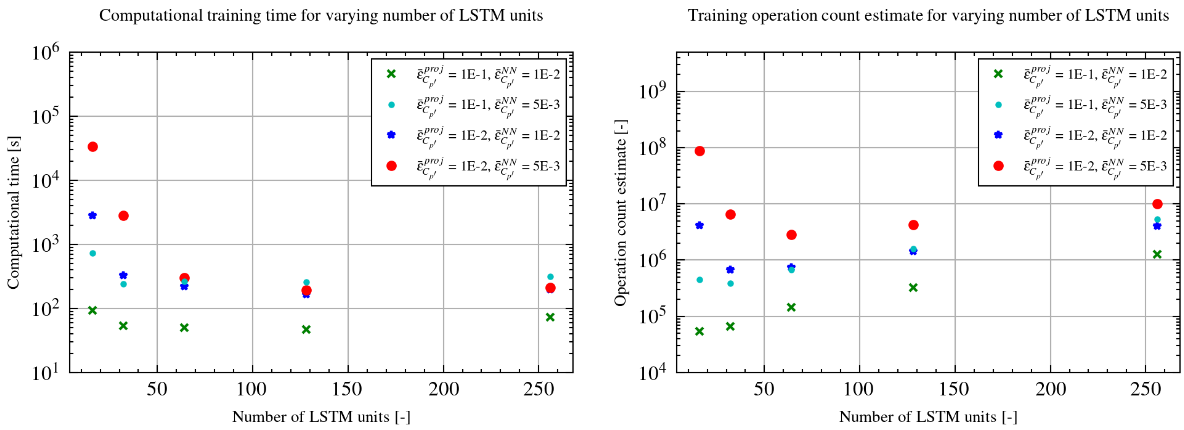
Figure 5.11: Comparison between the computational training time and operation count estimate for varying number of LSTM layers.

Table 5.3: Training results for a varying number of LSTM layers

LSTM layers	$\bar{\epsilon}^{proj,training}_{C_{p'}}$ [-]	$\bar{\epsilon}^{NN}_{C_{p'}}$ [-]	**Training time [s]**	**Trainable parameters [-]**	**Iterations [-]**	**Operation count estimate [-]**
1	**1.00E-01**	1.00E-02	53.23121	9733	6.8	66184.4
		5.00E-03	243.2572	9733	39.6	385426.8
	1.00E-02	1.00E-02	326.65	14248	48	683904
		5.00E-03	2798.653	14248	460.8	6565478
2	**1.00E-01**	1.00E-02	95.13692	18053	8	144424
		5.00E-03	293.6007	18053	28.2	509094.6
	1.00E-02	1.00E-02	421.0009	22568	38	857584
		5.00E-03	3546.738	22568	347.6	7844637
3	**1.00E-01**	1.00E-02	118.3952	26373	7	184611
		5.00E-03	460.6099	26373	31.66667	835145.1
	1.00E-02	1.00E-02	671.2726	30888	46.2	1427026
		5.00E-03	5340.424	30888	406.2	12546706
4	**1.00E-01**	1.00E-02	224.0489	34693	11	381623
		5.00E-03	700.5039	34693	38	1318334
	1.00E-02	1.00E-02	1034.935	39208	55.2	2164282
		5.00E-03	6265.555	39208	678.6	26606549
5	**1.00E-01**	1.00E-02	296.3622	43013	12.8	550566.4
		5.00E-03	1133.677	43013	52.2	2245279
	1.00E-02	1.00E-02	1640.479	47528	77	3659656
		5.00E-03	10882.64	47528	1200.4	57052611

Variation of LSTM units

For the number of LSTM units, the results of the computational time and the operation count estimate are shown in Figure 5.12. It can be seen that as the number of LSTM units increases, the computational time for models with a lower projection error, i.e., $\bar{\epsilon}^{proj,training}_{C_{p'}}$ = 1E-2, drastically decreases up until 128 LSTM units. The differences between the highest and lowest computational time is up to at least a factor of 150, where the minimum time measured is equal to approximately 3 minutes. Furthermore, as shown in the figure, it can be concluded that for higher projection errors, i.e., $\bar{\epsilon}^{proj,training}_{C_{p'}}$ = 1E-1, the number of LSTM units have less of an effect on the training time of the model. The overall results for the given levels of accuracy and variation in the number of LSTM units are shown in Table 5.4. It can be seen that the minimum computational training time for all levels of accuracy is found at 128 LSTM units.



(a) Computational training time in seconds.

(b) Operation count estimate.

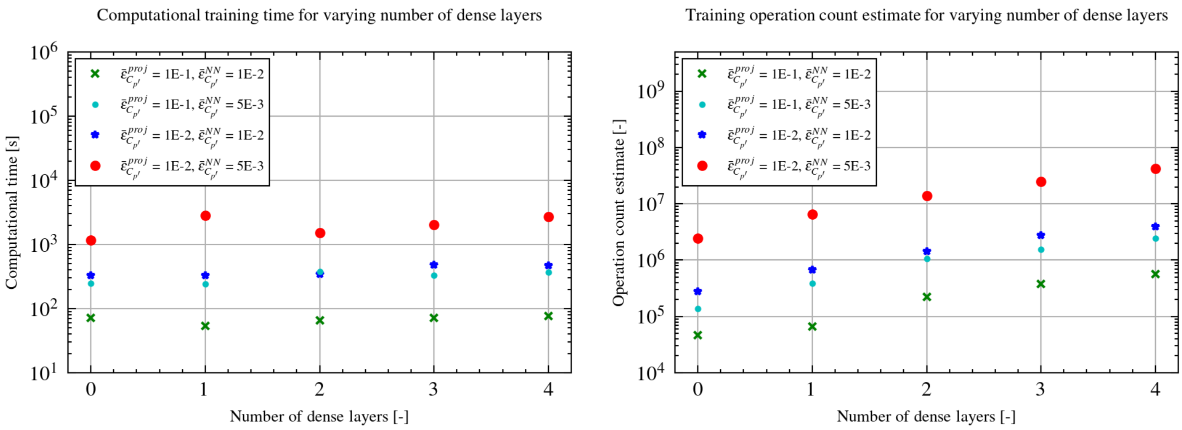
Figure 5.12: Comparison between the computational training time and operation count estimate for varying number of LSTM units.

Table 5.4: Training results for a varying number of LSTM units

LSTM units	$\bar{\epsilon}^{proj,training}_{C_{p'}}$ [-]	$\bar{\epsilon}^{NN}_{C_{p'}}$ [-]	Training time [s]	Trainable parameters [-]	Iterations [-]	Operation count estimate [-]
16	**1.00E-01**	1.00E-02	93.40714	4229	12.8	54131.2
		5.00E-03	734.8835	4229	107.4	454194.6
	1.00E-02	1.00E-02	2828.057	8744	476.4	4165642
		5.00E-03	33570.16	8744	10001	87448744
32	**1.00E-01**	1.00E-02	53.23121	9733	6.8	66184.4
		5.00E-03	243.2572	9733	39.6	385426.8
	1.00E-02	1.00E-02	326.65	14248	48	683904
		5.00E-03	2798.653	14248	460.8	6565478
64	**1.00E-01**	1.00E-02	50.63236	26885	5.4	145179
		5.00E-03	264.5275	26885	25.4	682879
	1.00E-02	1.00E-02	219.8688	31400	23.6	741040
		5.00E-03	299.4722	31400	91.4	2869960
128	**1.00E-01**	1.00E-02	46.91124	85765	3.8	325907
		5.00E-03	259.593	85765	18.4	1578076
	1.00E-02	1.00E-02	167.5636	90280	16	1444480
		5.00E-03	193.488	90280	47.4	4279272
256	**1.00E-01**	1.00E-02	72.84511	301829	4.2	1267682
		5.00E-03	315.9105	301829	17.6	5312190
	1.00E-02	1.00E-02	199.5296	306344	13.4	4105010
		5.00E-03	209.8117	306344	32.6	9986814

Variation of dense layers

For the number of dense layers, the results of the computational time and the operation count estimate are shown in Figure 5.13. It can be seen that as the number of dense layers increases, the computational time for the models with a low projection error, i.e., $\bar{\epsilon}^{proj,training}_{C_{p'}}$ = 1E-2, slightly increases. Furthermore, it can be concluded that the number of dense layers has no effect on the computational training time for the model with the highest projection error, i.e., $\bar{\epsilon}^{proj,training}_{C_{p'}}$ = 1E-1, as the computational training time stays relatively constant while the operation count estimate increases with the number of dense layers. The overall results for the given levels of accuracy and variation in the number of dense layers are shown in Table 5.5.



(a) Computational training time in seconds. (b) Operation count estimate.

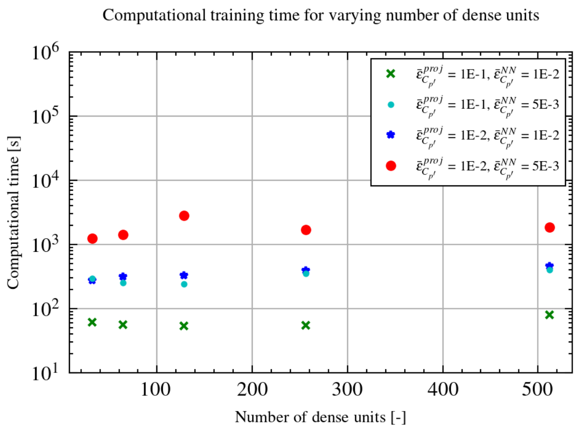
Figure 5.13: Comparison between the computational training time and operation count estimate for varying number of dense layers.

Table 5.5: Training results for a varying number of dense layers

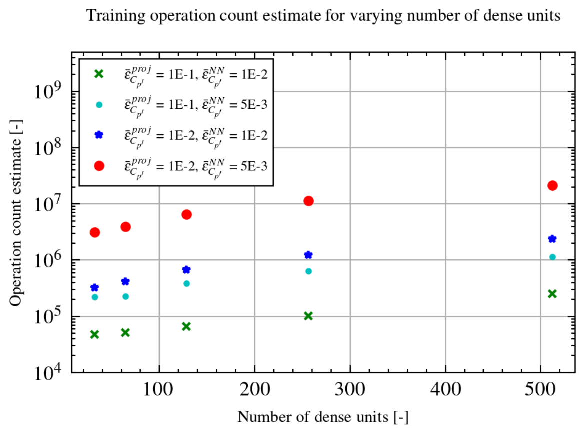
Dense layers	$\bar{\epsilon}^{proj,training}_{C_{p'}}$ [-]	$\bar{\epsilon}^{NN}_{C_{p'}}$ [-]	**Training time [s]**	**Trainable parameters [-]**	**Iterations [-]**	**Operation count estimate [-]**
0	**1.00E-01**	1.00E-02	72.09693	5029	9.2	46266.8
		5.00E-03	245.1009	5029	27.2	136788.8
	1.00E-02	1.00E-02	331.3637	6184	44.6	275806.4
		5.00E-03	1173.12	6184	398.8	2466179
1	**1.00E-01**	1.00E-02	53.23121	9733	6.8	66184.4
		5.00E-03	243.2572	9733	39.6	385426.8
	1.00E-02	1.00E-02	326.65	14248	48	683904
		5.00E-03	2798.653	14248	460.8	6565478
2	**1.00E-01**	1.00E-02	66.15296	26245	8.4	220458
		5.00E-03	374.0533	26245	40.4	1060298
	1.00E-02	1.00E-02	343.7377	30760	47	1445720
		5.00E-03	1528.83	30760	453.6	13952736
3	**1.00E-01**	1.00E-02	71.60106	42757	8.8	376261.6
		5.00E-03	331.2201	42757	36.6	1564906
	1.00E-02	1.00E-02	475.8283	47272	58.6	2770139
		5.00E-03	2030.686	47272	531.8	25139250
4	**1.00E-01**	1.00E-02	76.60156	59269	9.6	568982.4
		5.00E-03	364.7361	59269	41.6	2465590
	1.00E-02	1.00E-02	470.6361	63784	61.4	3916338
		5.00E-03	2721.934	63784	661.4	42186738

Variation of dense units

For the number of dense units, the results of the computational time and the operation count estimate are shown in Figure 5.14. It can be seen that as the number of dense units increases, the computational time for the models with a low projection error, i.e., $\bar{\boldsymbol{\epsilon}}^{proj,training}_{C_{p'}}$ = 1E-2, slightly increases. Furthermore, it can be concluded that the number of dense units has no effect on the computational training time for the model with the highest projection error, i.e., $\bar{\boldsymbol{\epsilon}}^{proj,training}_{C_{p'}}$ = 1E-1, as the computational training time stays relatively constant while the operation count estimate increases with the number of dense units for all levels of accuracy. The overall results for the given levels of accuracy and variation in the number of dense layers are shown in Table 5.6.



(a) Computational training time in seconds.



(b) Operation count estimate.

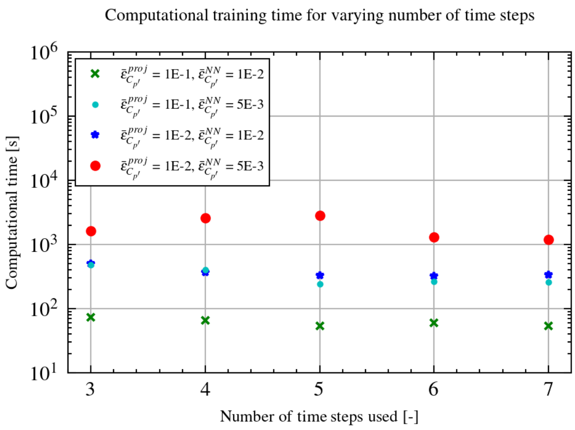
Figure 5.14: Comparison between the computational training time and operation count estimate for varying number of dense units.

Table 5.6: Training results for a varying number of dense units

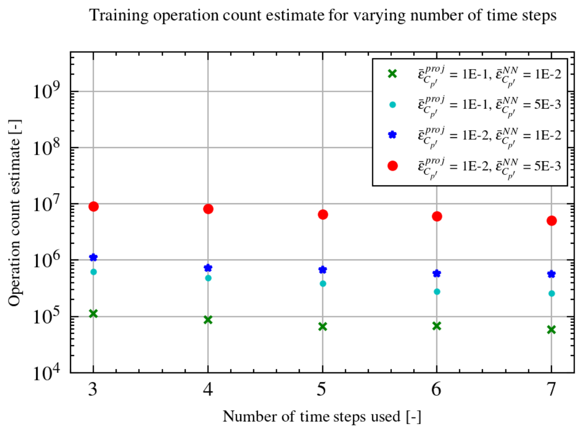
Dense units	$\bar{\epsilon}_{C_{p'}}^{proj,training}$ [-]	$\bar{\epsilon}_{C_{p'}}^{NN}$ [-]	Training time [s]	Trainable parameters [-]	Iterations [-]	Operation count estimate [-]
32	**1.00E-01**	1.00E-02	61.16608	6085	7.8	47463
		5.00E-03	294.2625	6085	36.8	223928
	1.00E-02	1.00E-02	277.8146	7240	45.2	327248
		5.00E-03	1252.241	7240	432.2	3129128
64	**1.00E-01**	1.00E-02	56.52176	7301	7	51107
		5.00E-03	254.0716	7301	31.2	227791.2
	1.00E-02	1.00E-02	315.6246	9576	43.6	417513.6
		5.00E-03	1411.94	9576	416.6	3989362
128	**1.00E-01**	1.00E-02	53.23121	9733	6.8	66184.4
		5.00E-03	243.2572	9733	39.6	385426.8
	1.00E-02	1.00E-02	326.65	14248	48	683904
		5.00E-03	2798.653	14248	460.8	6565478
256	**1.00E-01**	1.00E-02	55.49962	14597	7	102179
		5.00E-03	353.2781	14597	43.6	636429.2
	1.00E-02	1.00E-02	389.9887	23592	52	1226784
		5.00E-03	1699.312	23592	480.6	11338315
512	**1.00E-01**	1.00E-02	79.25485	24325	10.4	252980
		5.00E-03	399.447	24325	46.6	1133545
	1.00E-02	1.00E-02	460.7465	42280	56.2	2376136
		5.00E-03	1843.234	42280	510.4	21579712

Variation of time steps used

For the number of previous time steps used the results of the computational time and the operation count estimate are shown in Figure 5.15. It can be seen that as the number of previous time steps used increases, the computational time for the models with a low projection error, i.e., $\bar{\boldsymbol{\epsilon}}_{C_{p'}}^{proj,training}$ = 1E-2, decreases up to a factor of 2. For the 5 previous time steps used, the computational training time is maximum. Furthermore, it can be concluded that the number of previous time steps used has no effect on the computational training time for the model with the highest projection error, i.e., $\bar{\boldsymbol{\epsilon}}_{C_{p'}}^{proj,training}$ = 1E-1, as the computational training time stays relatively constant. As the previous time steps used do not affect the complexity of the model but only the amount of input data that is used to train the model, the operation count estimate remains relatively constant for all levels of accuracy. The overall results for the given levels of accuracy and variation in the number of previous time steps used are shown in Table 5.7.



(a) Computational training time in seconds.



(b) Operation count estimate.

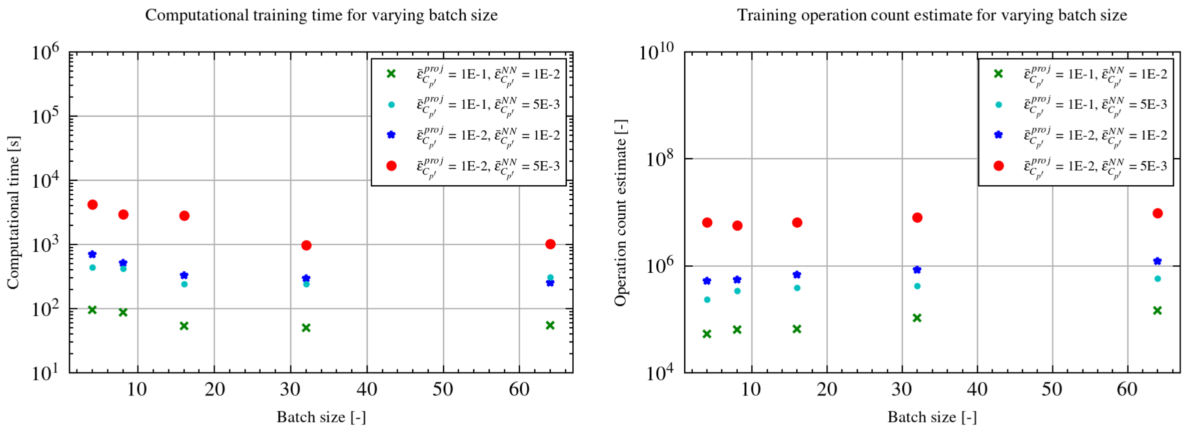
Figure 5.15: Comparison between the computational training time and operation count estimate for varying number of time steps used.

Table 5.7: Training results for a varying number of time steps used

Time steps used	$\bar{\epsilon}^{proj,training}_{C_{p'}}$ [-]	$\bar{\epsilon}^{NN}_{C_{p'}}$ [-]	**Training time [s]**	**Trainable parameters [-]**	**Iterations [-]**	**Operation count estimate [-]**
3	**1.00E-01**	1.00E-02	73.99291	9733	11.6	112902.8
		5.00E-03	484.9379	9733	63.8	620965.4
	1.00E-02	1.00E-02	506.7633	14248	79	1125592
		5.00E-03	1610.329	14248	637.4	9081675
4	**1.00E-01**	1.00E-02	65.16311	9733	9	87597
		5.00E-03	401.7217	9733	50	486650
	1.00E-02	1.00E-02	369.2927	14248	51.2	729497.6
		5.00E-03	2600.32	14248	571.8	8147006
5	**1.00E-01**	1.00E-02	53.23121	9733	6.8	66184.4
		5.00E-03	243.2572	9733	39.6	385426.8
	1.00E-02	1.00E-02	326.65	14248	48	683904
		5.00E-03	2798.653	14248	460.8	6565478
6	**1.00E-01**	1.00E-02	60.27392	9733	7	68131
		5.00E-03	265.1357	9733	29	282257
	1.00E-02	1.00E-02	326.3161	14248	41	584168
		5.00E-03	1298.474	14248	431.4	6146587
7	**1.00E-01**	1.00E-02	53.37125	9733	6	58398
		5.00E-03	259.1913	9733	26.4	256951.2
	1.00E-02	1.00E-02	339.8525	14248	39.4	561371.2
		5.00E-03	1194.519	14248	359.8	5126430

Variation of batch size

For the variation of the batch size, the results of the computational time and the operation count estimate are shown in Figure 5.16 It can be seen that as the batch size increases, the computational time for the models with a low projection error, i.e., $\bar{\epsilon}^{proj,training}_{C_{p'}}$ = 1E-2, decreases up to a factor of 4. After a batch size of 32, the computational training time remains constant. Furthermore, it can be concluded that the batch size has less of an effect on the computational training time for the model with the highest projection error, i.e., $\bar{\epsilon}^{proj,training}_{C_{p'}}$ = 1E-1, as the computational training time is reduced up to a factor 2 maximum. As the batch size does not affect the complexity of the model but only the amount of input and output data that is used simultaneously to train the model, the operation count estimate only slightly increases for all levels of accuracy. The overall results for the given levels of accuracy and variation of the batch size are shown in Table 5.8.



(a) Computational training time in seconds.

(b) Operation count estimate.

Figure 5.16: Comparison between the computational training time and operation count estimate for varying batch size.

Table 5.8: Training results for varying batch size

Batch sized	$\bar{\epsilon}^{proj,training}_{C_{p'}}$ [-]	$\bar{\epsilon}^{NN}_{C_{p'}}$ [-]	Training time [s]	Trainable parameters [-]	Iterations [-]	Operation count estimate [-]
4	**1.00E-01**	1.00E-02	95.02895	9733	5.4	52558.2
		5.00E-03	438.6905	9733	24.4	237485.2
	1.00E-02	1.00E-02	699.9768	14248	37	527176
		5.00E-03	4221.93	14248	454.2	6471442
8	**1.00E-01**	1.00E-02	88.35486	9733	6.6	64237.8
		5.00E-03	421.551	9733	35	340655
	1.00E-02	1.00E-02	509.2415	14248	38.4	547123.2
		5.00E-03	2934.361	14248	396	5642208
16	**1.00E-01**	1.00E-02	53.23121	9733	6.8	66184.4
		5.00E-03	243.2572	9733	39.6	385426.8
	1.00E-02	1.00E-02	326.65	14248	48	683904
		5.00E-03	2798.653	14248	460.8	6565478
32	**1.00E-01**	1.00E-02	50.54407	9733	10.8	105116.4
		5.00E-03	242.5923	9733	43.4	422412.2
	1.00E-02	1.00E-02	293.0864	14248	59.2	843481.6
		5.00E-03	978.4491	14248	574	8178352
64	**1.00E-01**	1.00E-02	54.49008	9733	15	145995
		5.00E-03	310.0013	9733	59.4	578140.2
	1.00E-02	1.00E-02	252.7937	14248	86.2	1228178
		5.00E-03	1031.841	14248	679.6	9682941

POD-LSTM model training evaluation

The results as discussed above should provide the user of the POD-LSTM model insights in the variation of the given hyperparameters. From the results, it can be concluded that the variation of the LSTM units has the largest effect on the computational training time, with differences up to a factor of 150. For all the other hyperparameters differences up to a factor of 2 up until 4 are found for the computational training and can therefore be considered small with respect to the variation of the number of LSTM units used. The maximum computational training time for the lowest projection error during this analysis, i.e., $\bar{\epsilon}^{proj,training}_{C_{p'}} =$ 1E-2, was found to be up to more than 9 hours, while the minimum training time was only 3 minutes. An overview of the minimum and maximum computational training time found for the variance of the hyperparameters including the average training time for all the levels of accuracy as well as a comparison with respect to the high-fidelity simulations and previously constructed surrogate models is shown in Section 5.1.3. From the table, it can be concluded that for the construction of the POD-LSTM model the average training time increases with an increasing level of accuracy, i.e., where both projection error and neural network error are small. Furthermore, it can be seen that the hyperparameters have a large effect on the computational training time that is necessary to train the model to the given level of accuracy. From the hyperparameter sensitivity analysis it can be concluded that if the hyperparameters are considered to be optimal, the maximum training time of the POD-LSTM model can be minimized to only 4 minutes for both the lowest projection error and prediction error, i.e., $\bar{\boldsymbol{\epsilon}}_{C_{p'}}^{proj,training}$ = 1E-2 and $\bar{\boldsymbol{\epsilon}}_{C_{p'}}^{NN}$ = 5E-3. The total training time of the POD-LSTM model is the summation of the simulation time of the high-fidelity samples and the training of the POD-LSTM model, which in the case of the optimal hyperparameter settings is negligible with respect to the simulation time of the high-fidelity samples. The POD-LSTM model shows a significant decrease in the model training time with respect to the previously constructed surrogate model as described by Papp, however, it must be noted that the accuracy of the CNN is significantly higher by up to a factor of 1000. Due to time constraints, a fair comparison between the different surrogate models was not possible and will be proposed in future work.

Table 5.9: POD-LSTM computational training time comparison with respect to high-fidelity simulations and previous constructed surrogate model from D. Papp [18]

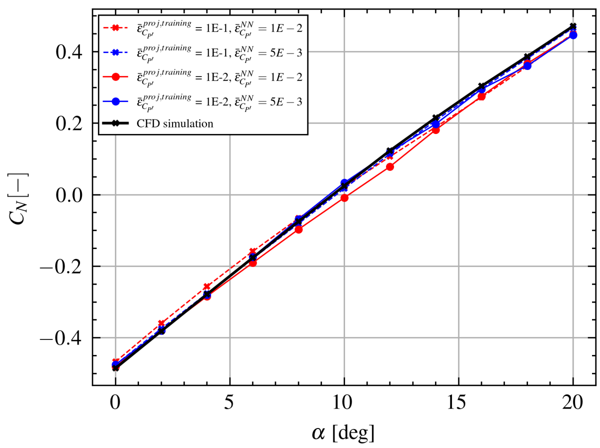
Model	$\bar{\epsilon}_{C_{p'}}^{proj,training}$ [-]	$\bar{\epsilon}_{C_{p'}}^{NN}$ [-]	Minimum training time	Maximum training time	Average training time
POD-LSTM (@2.4 GHz)	**1.00E-01**	1.00E-02	0.8 m	8.5 m	1.6 m
		5.00E-03	4.0 m	48.9 m	7.7 m
	1.00E-02	1.00E-02	0.8 m	47.1 m	8.5 m
		5.00E-03	3.2 m	9 h 19 m	58.1 m
ENSOLV (Training samples)					9600 h/120 h (@80 threads)
CNN (Papp @2.6 GHz)		**7.00E-05**			17 h 56 m (boosted with GPU tesla k40m @ 2880 cores)

5.2. Online stage results: POD-LSTM non-intrusive ROM predictions

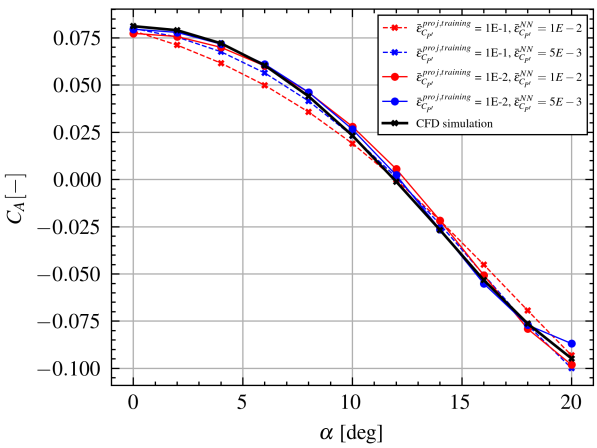
This section will discuss the online stage results, hence the performance of the prediction capabilities of the POD-LSTM model after it is trained. The model settings that are used for the analysis of the online stage results are based on the baseline model parameters as described in Chapter 4. This section will go into detail about the steady simulation results and the unsteady simulations results for harmonic oscillations and will provide a comparison with previous research. Finally, the computational performance of the POD-LSTM is shown for the prediction time.

5.2.1. Steady simulations

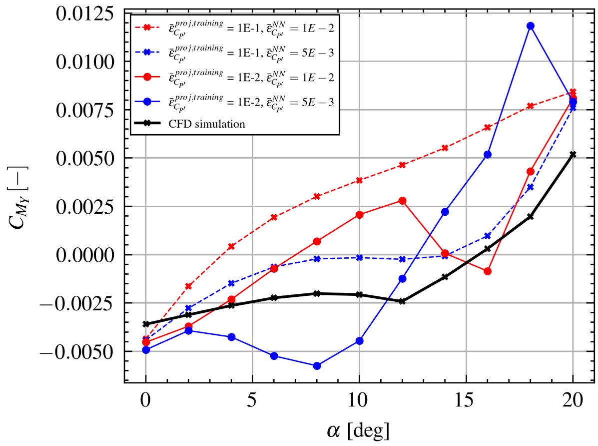
The steady simulations are performed between an angle of attack range between 0 and 20 degrees. The comparison between the high-fidelity CFD simulations that are gathered from ENSOLV, the in-house solver of the NLR, and the prediction of the POD-LSTM model for the steady simulations is shown Figure 5.17. From the figure it can be seen that both the normal force coefficient and the axial force coefficient are covered well for the given model, where the model with projection error $\bar{\boldsymbol{\epsilon}}^{\boldsymbol{proj}}$ = 1E-2 and prediction error $\bar{\epsilon}^{NN}$ = 5E-3 shows the best fit. However, the prediction of the POD-LSTM model for the pitching moment coefficient for all ranges of the angle of attack is inaccurate. For the prediction of the pitching moment coefficient, it can be seen that the models that were run with a higher projection error, i.e., in particular, $\bar{\boldsymbol{\epsilon}}_{C_{p'}}^{proj,training}$ = 1E-1 and $\bar{\boldsymbol{\epsilon}}_{C_{p'}}^{NN}$ = 5E-3, are able to capture the general trend of the CFD simulations, although the values of the prediction are off. Interesting to note from this figure, is that the models with a lower projection error, i.e., $\bar{\boldsymbol{\epsilon}}_{C_{p'}}^{proj,training}$ = 1E-2, are less accurate than the models with a higher projection error and show abrupt changes in the value of the pitching moment coefficient, which is not in trend with what is expected for higher accuracy models.



(a) POD-LSTM prediction of the normal force coefficient for steady simulations.



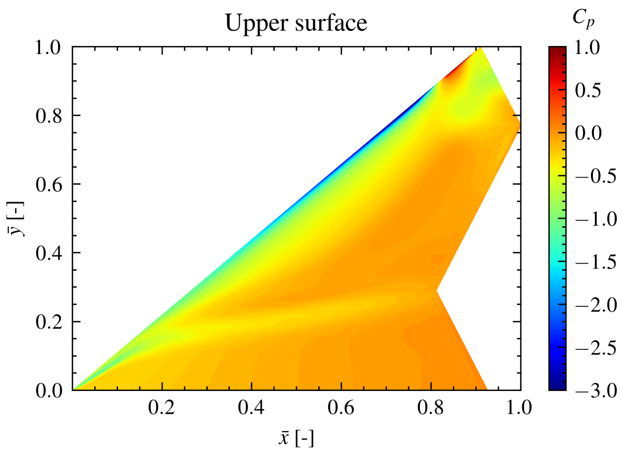
(b) POD-LSTM prediction of the axial force coefficient for steady simulations.



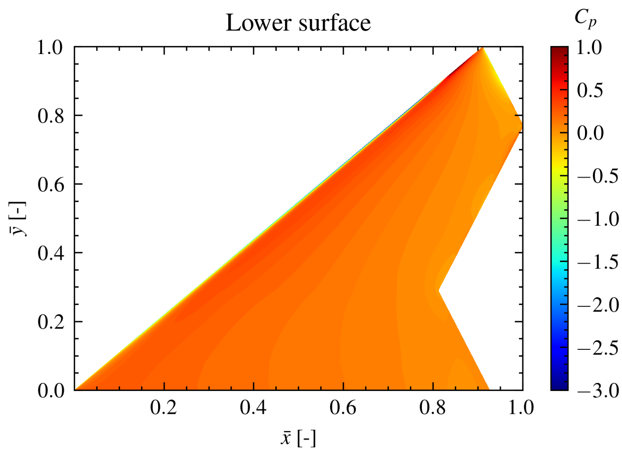
(c) POD-LSTM prediction of the pitching moment coefficient for steady simulations.

Figure 5.17: POD-LSTM prediction of the integral load coefficients for steady simulations.

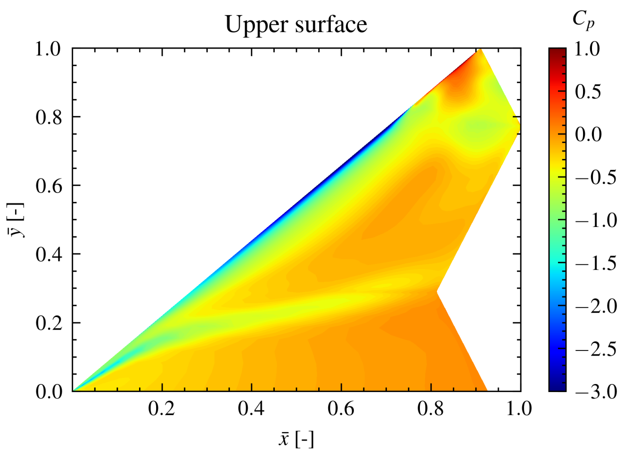
In order to determine the limitations of the POD-LSTM model and to identify the errors of the prediction of the pitching moment coefficient, the results of the steady simulations are investigated in more detail. For this analysis, the absolute error of the surface pressure coefficient is investigated for the different levels of accuracy. In particular, the angles of attack of 18 and 20 degrees are of interest, as these values are in the scope of the non-linear aerodynamic regime of the UCAV MULDICON and show large differences for the different levels of accuracy. In Figure 5.18 the distribution of the instantaneous surface pressure coefficients for both the upper and lower surface of the MULDICON are shown for the angles of attack of 18 and 20 degrees, respectively. From the figure, it can be seen that the flow both includes the development of the leading-edge vortex and wingtip separation.



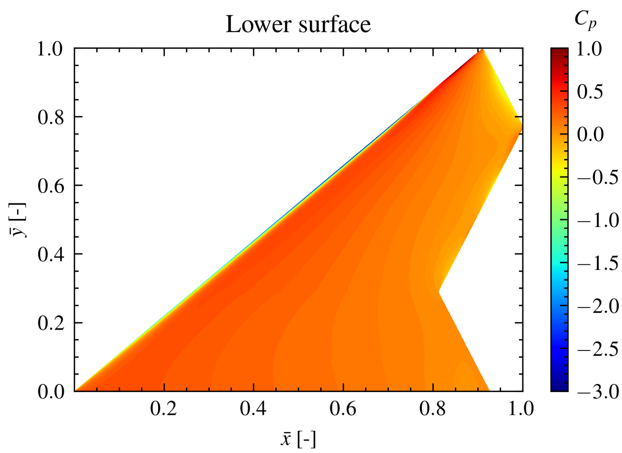
(a) Upper surface pressure distribution for steady simulations for $\alpha = 18$ deg.



(b) Lower surface pressure distribution for steady simulations for $\alpha = 18$ deg.



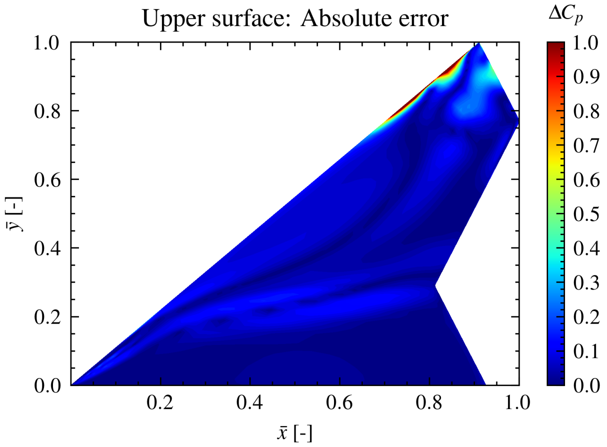
(c) Upper surface pressure distribution for steady simulations for $\alpha = 20$ deg.



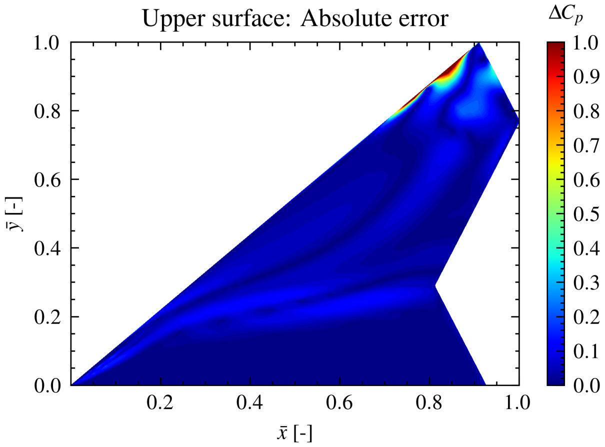
(d) Lower surface pressure distribution for steady simulations for $\alpha = 20$ deg.

Figure 5.18: Surface pressure distribution for steady simulations for $\alpha = 20$ deg obtained from CFD simulations.

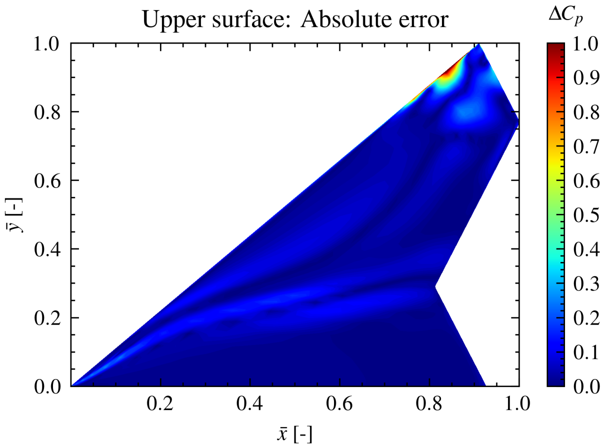
The absolute error of the surface pressure coefficient, i.e., the error between the prediction of the POD-LSTM model and the high-fidelity CFD simulations, for the upper and lower surface of the MULDICON at an angle of attack of 18 and 20 degrees is shown in Figure 5.19, Figure 5.20 and Figure 5.21, Figure 5.22 respectively. It can be seen from the figures that for both angles of attack, the error of the prediction of the lower surface pressure coefficients is marginal, while the main contribution of the total prediction error for the surface pressure coefficient is due to inaccurate prediction of the surface pressure near the tip region at the upper surface. As already discussed in the offline stage results section, it was shown that an inaccurate flow representation of the surface pressure coefficients near the tip region leads to higher errors for the pitching moment coefficient. Hence, the inaccurate prediction of the pitching moment coefficient for the steady simulations is caused by the inaccurate prediction of the surface pressure coefficient near the tip. Furthermore, it can be seen that the models with a higher projection error, although unexpected, show a smaller error near the tip region than the models with a lower projection error.



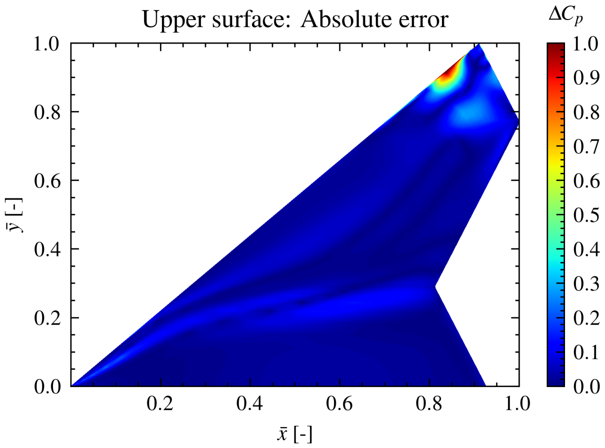
(a) Absolute prediction error at α = 18 deg for surrogate model $\bar{\epsilon}_{C_{P'}}^{proj,training}$ = 1E-1, $\bar{\epsilon}_{C_{P'}}^{NN,training}$ = 1E-2.



(b) Absolute prediction error at α = 18 deg for surrogate model $\bar{\epsilon}_{C_{P'}}^{proj,training}$ = 1E-1, $\bar{\epsilon}_{C_{P'}}^{NN,training}$ = 5E-3.

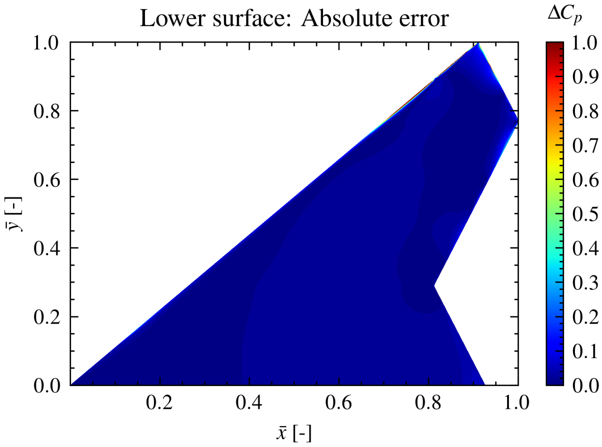


(c) Absolute prediction error at α = 18 deg for surrogate model $\bar{\epsilon}_{C_{P'}}^{proj,training}$ = 1E-2, $\bar{\epsilon}_{C_{P'}}^{NN,training}$ = 1E-2.

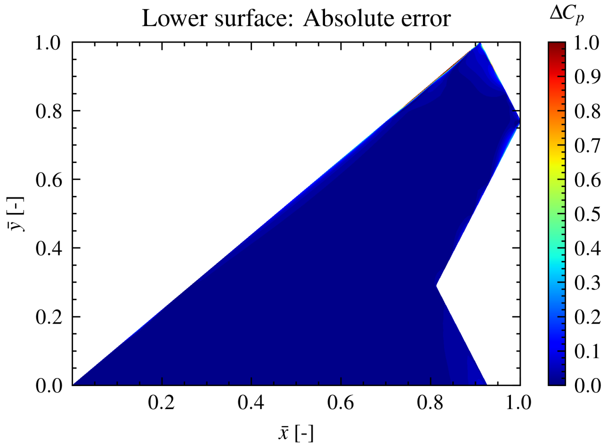


(d) Absolute prediction error at α = 18 deg for surrogate model $\bar{\epsilon}_{C_{P'}}^{proj,training}$ = 1E-2, $\bar{\epsilon}_{C_{P'}}^{NN,training}$ = 5E-3.

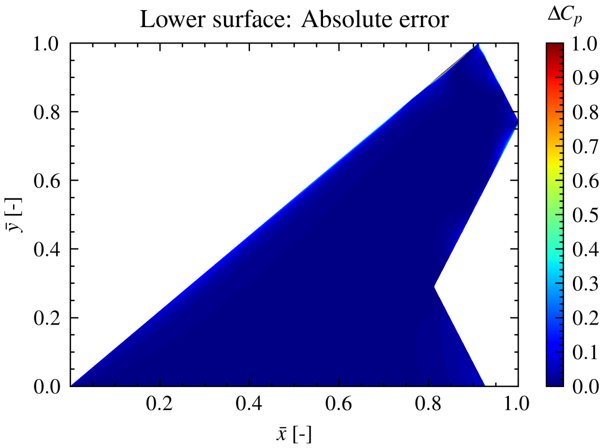
Figure 5.19: POD-LSTM absolute prediction error of the upper surface pressure distribution for different surrogate model settings for steady simulations at α = 18 deg.



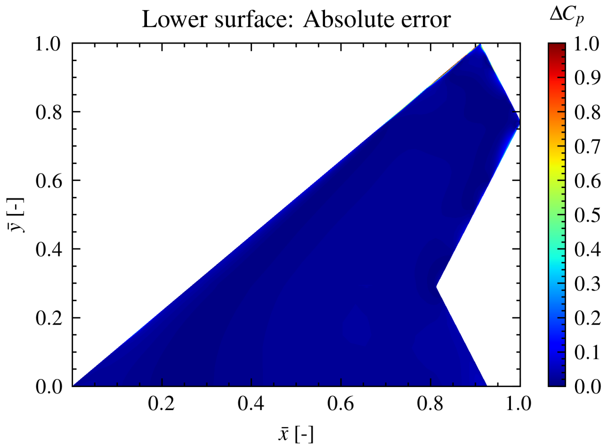
(a) Absolute prediction error at α = 18 deg for surrogate model $\bar{\epsilon}_{C_{P'}}^{proj,training}$ = 1E-1, $\bar{\epsilon}_{C_{P'}}^{NN,training}$ = 1E-2.



(b) Absolute prediction error at α = 18 deg for surrogate model $\bar{\epsilon}_{C_{P'}}^{proj,training}$ = 1E-1, $\bar{\epsilon}_{C_{P'}}^{NN,training}$ = 5E-3.

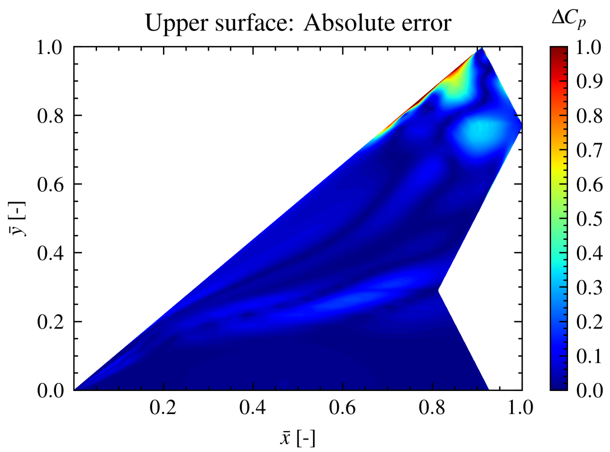


(c) Absolute prediction error at α = 18 deg for surrogate model $\bar{\epsilon}_{C_{P'}}^{proj,training}$ = 1E-2, $\bar{\epsilon}_{C_{P'}}^{NN,training}$ = 1E-2.

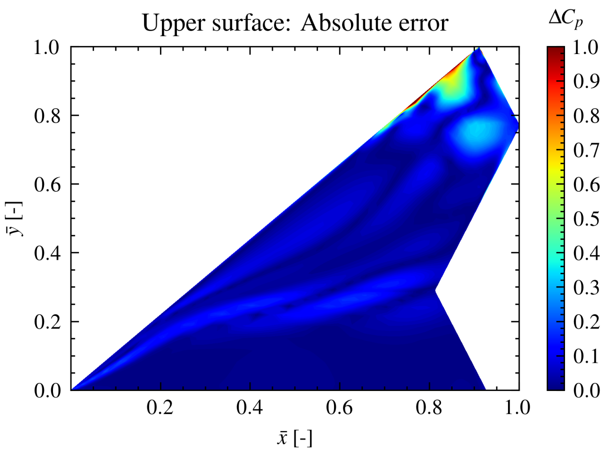


(d) Absolute prediction error at α = 18 deg for surrogate model $\bar{\epsilon}_{C_{P'}}^{proj,training}$ = 1E-2, $\bar{\epsilon}_{C_{P'}}^{NN,training}$ = 5E-3.

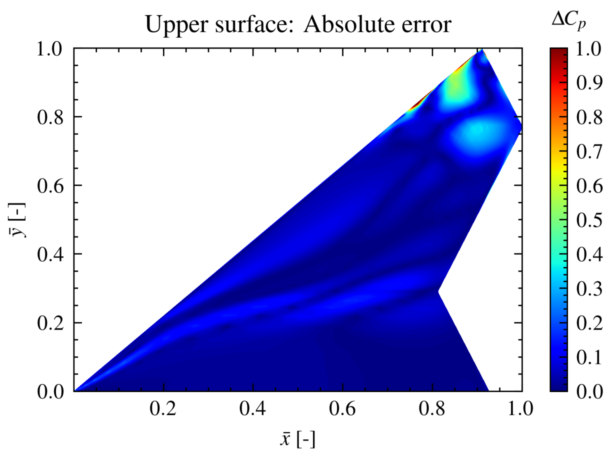
Figure 5.20: POD-LSTM absolute prediction error of the lower surface pressure distribution for different surrogate model settings for steady simulations at α = 18 deg.



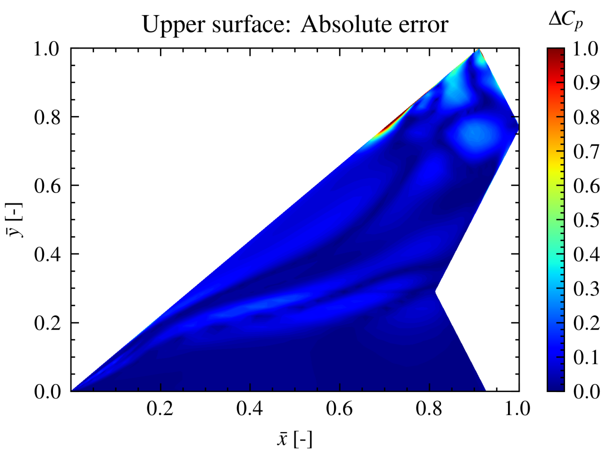
(a) Absolute prediction error at $\alpha = 20$ deg for surrogate model $\bar{\epsilon}_{C_{p'}}^{proj,training} = 1E\text{-}1$, $\bar{\epsilon}_{C_{p'}}^{NN,training} = 1E\text{-}2$.



(b) Absolute prediction error at $\alpha = 20$ deg for surrogate model $\bar{\epsilon}_{C_{p'}}^{proj,training} = 1E\text{-}1$, $\bar{\epsilon}_{C_{p'}}^{NN,training} = 5E\text{-}3$.

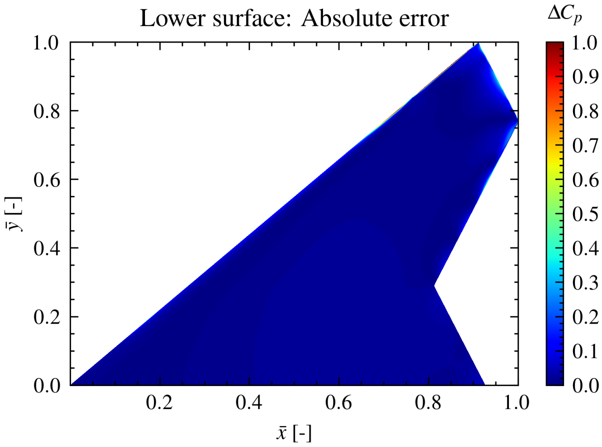


(c) Absolute prediction error at $\alpha = 20$ deg for surrogate model $\bar{\epsilon}_{C_{p'}}^{proj,training} = 1E\text{-}2$, $\bar{\epsilon}_{C_{p'}}^{NN,training} = 1E\text{-}2$.

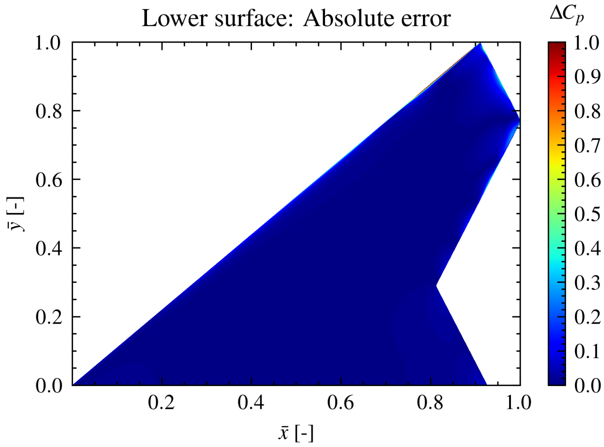


(d) Absolute prediction error at $\alpha = 20$ deg for surrogate model $\bar{\epsilon}_{C_{p'}}^{proj,training} = 1E\text{-}2$, $\bar{\epsilon}_{C_{p'}}^{NN,training} = 5E\text{-}3$.

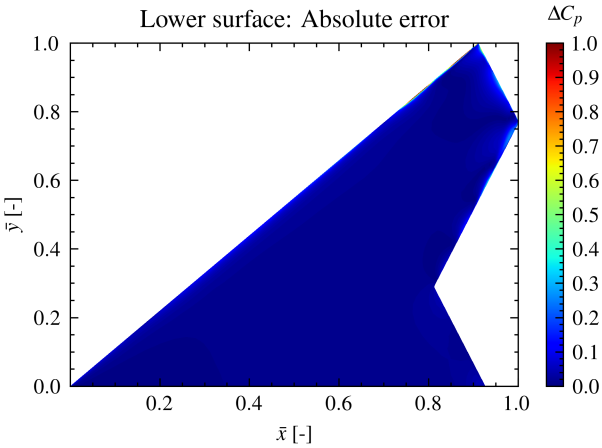
Figure 5.21: POD-LSTM absolute prediction error of the upper surface pressure distribution for different surrogate model settings for steady simulations at $\alpha = 20$ deg.



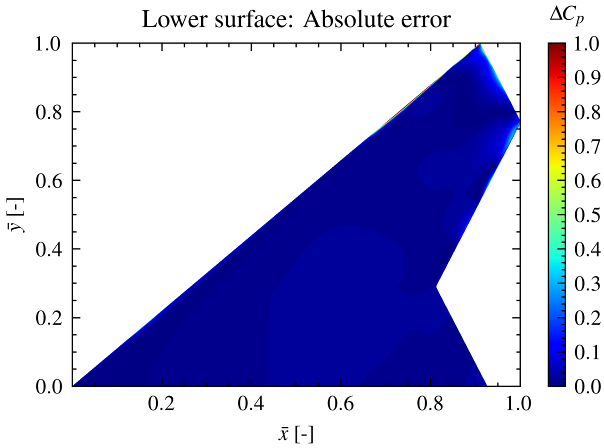
(a) Absolute prediction error at $\alpha = 20$ deg for surrogate model $\bar{\epsilon}_{C_{P'}}^{proj,training} = $ 1E-1, $\bar{\epsilon}_{C_{P'}}^{NN,training} = $ 1E-2.



(b) Absolute prediction error at $\alpha = 20$ deg for surrogate model $\bar{\epsilon}_{C_{P'}}^{proj,training} = $ 1E-1, $\bar{\epsilon}_{C_{P'}}^{NN,training} = $ 5E-3.



(c) Absolute prediction error at $\alpha = 20$ deg for surrogate model $\bar{\epsilon}_{C_{P'}}^{proj,training} = $ 1E-2, $\bar{\epsilon}_{C_{P'}}^{NN,training} = $ 1E-2.



(d) Absolute prediction error at $\alpha = 20$ deg for surrogate model $\bar{\epsilon}_{C_{P'}}^{proj,training} = $ 1E-2, $\bar{\epsilon}_{C_{P'}}^{NN,training} = $ 5E-3.

Figure 5.22: POD-LSTM absolute prediction error of the lower surface pressure distribution for different surrogate model settings for steady simulations at $\alpha = 20$ deg.

5.2.2. Unsteady simulations: Harmonic oscillations

This section will discuss the results that are obtained from the unsteady simulations. The unsteady simulations are performed using different sets of test samples for harmonic pitch and plunge motions. An overview of the performed simulations is shown in Section 4.2.1.

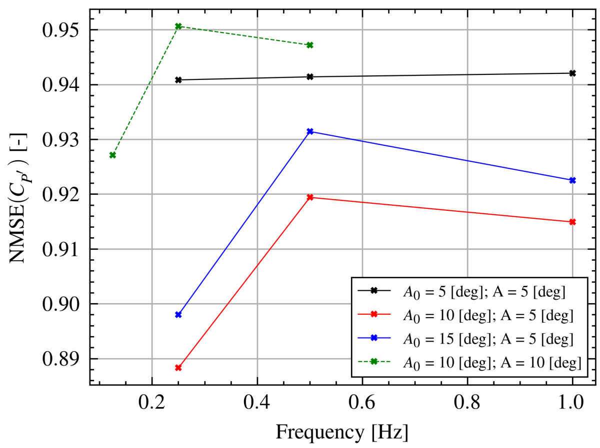
In order to be able to compare the accuracy of the prediction for the different harmonic motions, a new metric is introduced: the Normalized Mean Squared Error (NMSE). This metric will determine the performance of each model, whereas an exact representation of the model will result into a NMSE of 1. The lower the NMSE, the worse the accuracy of the model. The expression for the NMSE is as follows:

$$NMSE(y_{predicted}, y_{true}) = 1 - \frac{\| y_{predicted} - y_{true} \|^2}{\| y_{true} - \mathbb{E}(y_{true}) \|^2} \tag{5.1}$$

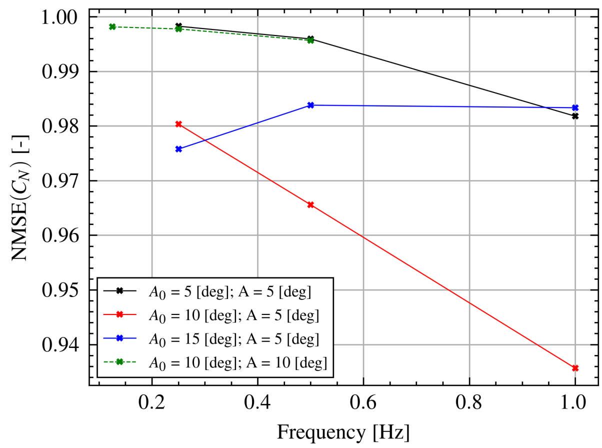
This section will compare the different harmonic motions and will investigate the cause of the errors for the given motions. First, the harmonic pitch motions will be investigated. Second, the harmonic plunge motions will be investigated. Finally, a comparison between the proposed model and previous studies will be shown.

Pitch oscillations

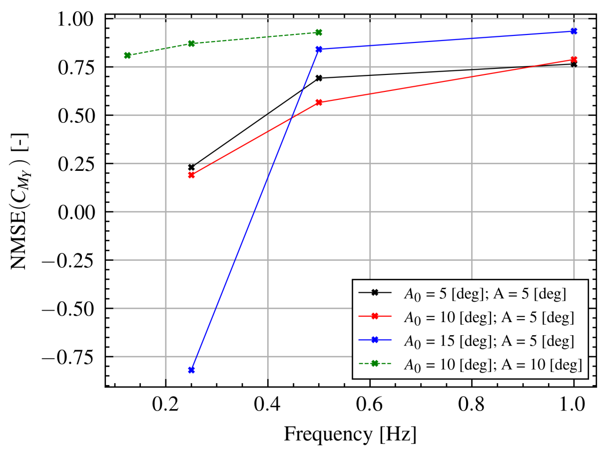
This section will discuss the results of the prediction of POD-LSTM model for the unsteady harmonic pitch motions. In Figure 5.23 the results are shown for the prediction of the surface pressure coefficient, the normal force coefficient, and the pitching moment coefficient in terms of NMSE and are labeled by frequency for each pitch maneuver. The NMSE is measured as error between the prediction of the POD-LSTM model and the CFD simulations. From the figures, it can be seen that on average the POD-LSTM model for pitch maneuvers with a frequency of 0.5 Hz is producing the best results for the prediction of the surface pressure coefficient. Moreover, it can be seen that both the surface pressure coefficients and the normal force coefficient are predicted with a higher accuracy than the pitching moment coefficient. Interesting to note from the figures is that the harmonic pitch motions with a nominal value of 10 degrees with an amplitude of 5 degrees have the lowest accuracy of all cases, which is in agreement with the high projection error for these motions as found in the offline stage results. In order to determine the deficiencies of the POD-LSTM model for unsteady simulations, the case with the lowest accuracy for the surface pressure coefficient will be investigated.



(a) LSTM prediction accuracy of the surface pressure coefficient in terms of NMSE for varying frequency.



(b) LSTM prediction accuracy of the normal force coefficient in terms of NMSE for varying frequency.



(c) LSTM prediction accuracy of the pitching moment coefficient in terms of NMSE for varying frequency.

Figure 5.23: LSTM prediction accuracy of the pressure coefficients, normal coefficients, and pitching moment coefficients in terms of NMSE for pitch oscillations at varying nominal values and amplitudes.

As already mentioned above, the POD-LSTM model shows the highest prediction error for the test case of the harmonic pitch motions with a nominal value of 10 degrees and an amplitude of 5 degrees. To identify the limitations of the POD-LSTM model, the harmonic pitch motion with a nominal value of 10 degrees and an amplitude of 5 degrees with a frequency of 0.25 Hz will be investigated, as this test case resulted in the highest prediction error for the surface pressure coefficient. The input state variables as well as the error of the surface pressure coefficient for the given test sample in terms of MSE are shown in Figure 5.24. It can be seen from the figure that for both the models with a high projection error, $\bar{\boldsymbol{\epsilon}}^{\boldsymbol{proj,training}}$ = 1E-1, and low projection error, $\bar{\boldsymbol{\epsilon}}^{\boldsymbol{proj,training}}$ = 1E-2, the error of the surface pressure coefficient prediction is relatively high near the initial states of the harmonic pitch motion, which are $\alpha \approx 10$ deg. and $\dot{\alpha}, q \approx$ (-) 7.5 deg/s, respectively.

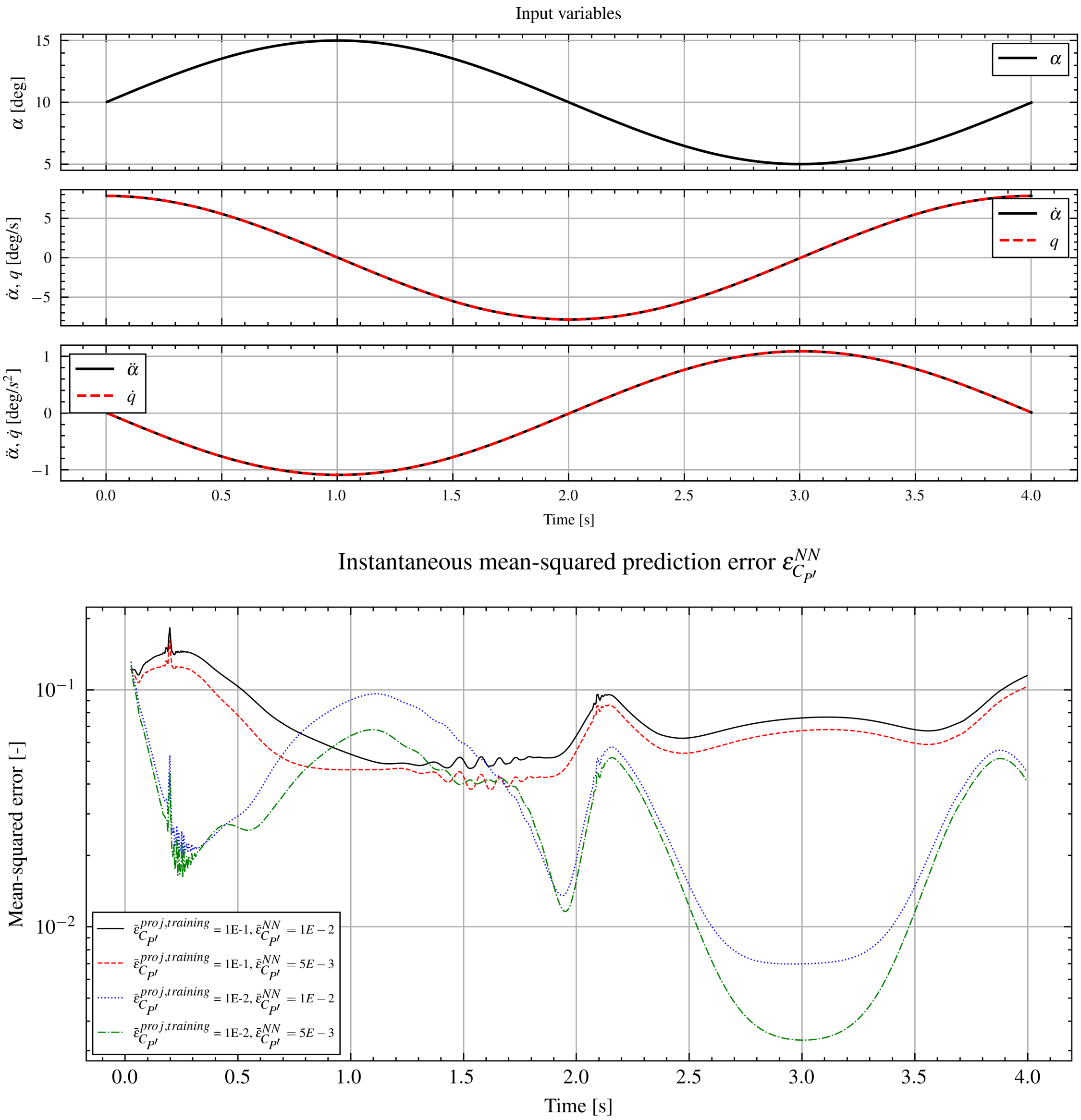


Figure 5.24: POD-LSTM input state variables for pitch oscillation $A_0 = 10$ deg , $A = 5$ deg, $f = 0.25$ Hz.

The prediction of the POD-LSTM model for the normal force coefficient, axial force coefficient, and the pitching moment coefficient is shown in Figure 5.25, Figure 5.26 and Figure 5.27, respectively. It can be seen that all the integral load coefficients are covered well by the POD-LSTM model with errors that are all in the same order of magnitude. Furthermore, it can be noted that the differences between the prediction errors for both projection errors, i.e., $\bar{\boldsymbol{\epsilon}}^{\boldsymbol{proj,training}}$ = 1E-1 and $\bar{\boldsymbol{\epsilon}}^{\boldsymbol{proj,training}}$ = 1E-2, can vary up to a factor 100. Overall, the model with the lowest projection error and largest prediction error, i.e., $\bar{\boldsymbol{\epsilon}}^{\boldsymbol{proj,training}}$ = 1E-1 and $\bar{\boldsymbol{\epsilon}}^{\boldsymbol{NN}}$ = 5E-3, has the lowest errors for the prediction of the integral load coefficients. From this it can be concluded that the number of spatial POD modes used for the construction of the reduced-order basis do not have a large influence on the prediction of the integral aerodynamic load coefficients for this case, whereas the number of spatial POD modes is of importance for the surface pressure coefficient prediction.

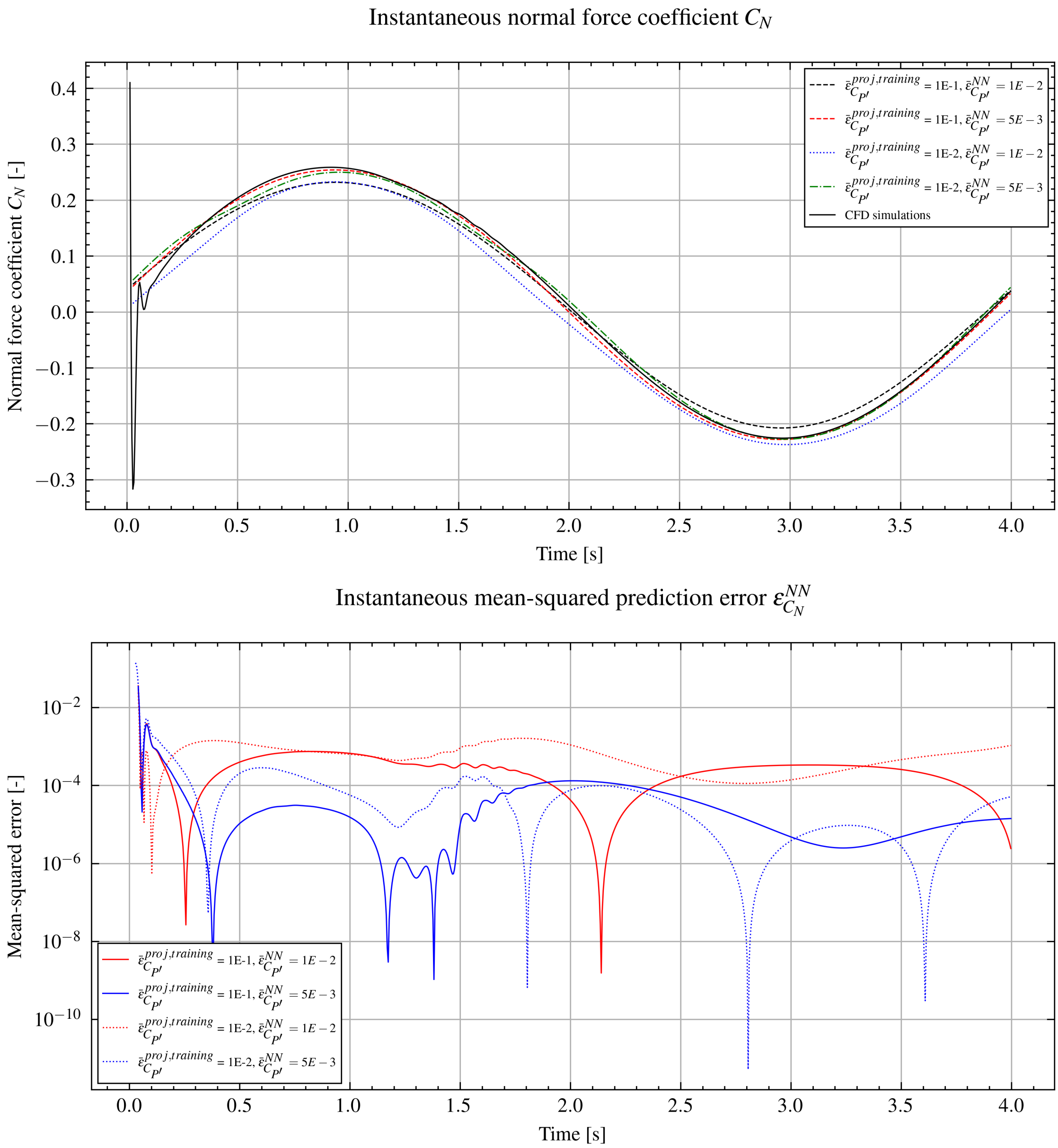


Figure 5.25: POD-LSTM prediction of the normal force coefficient for pitch oscillation $A_0 = 10$ deg , $A = 5$ deg, $f = 0.25$ Hz.

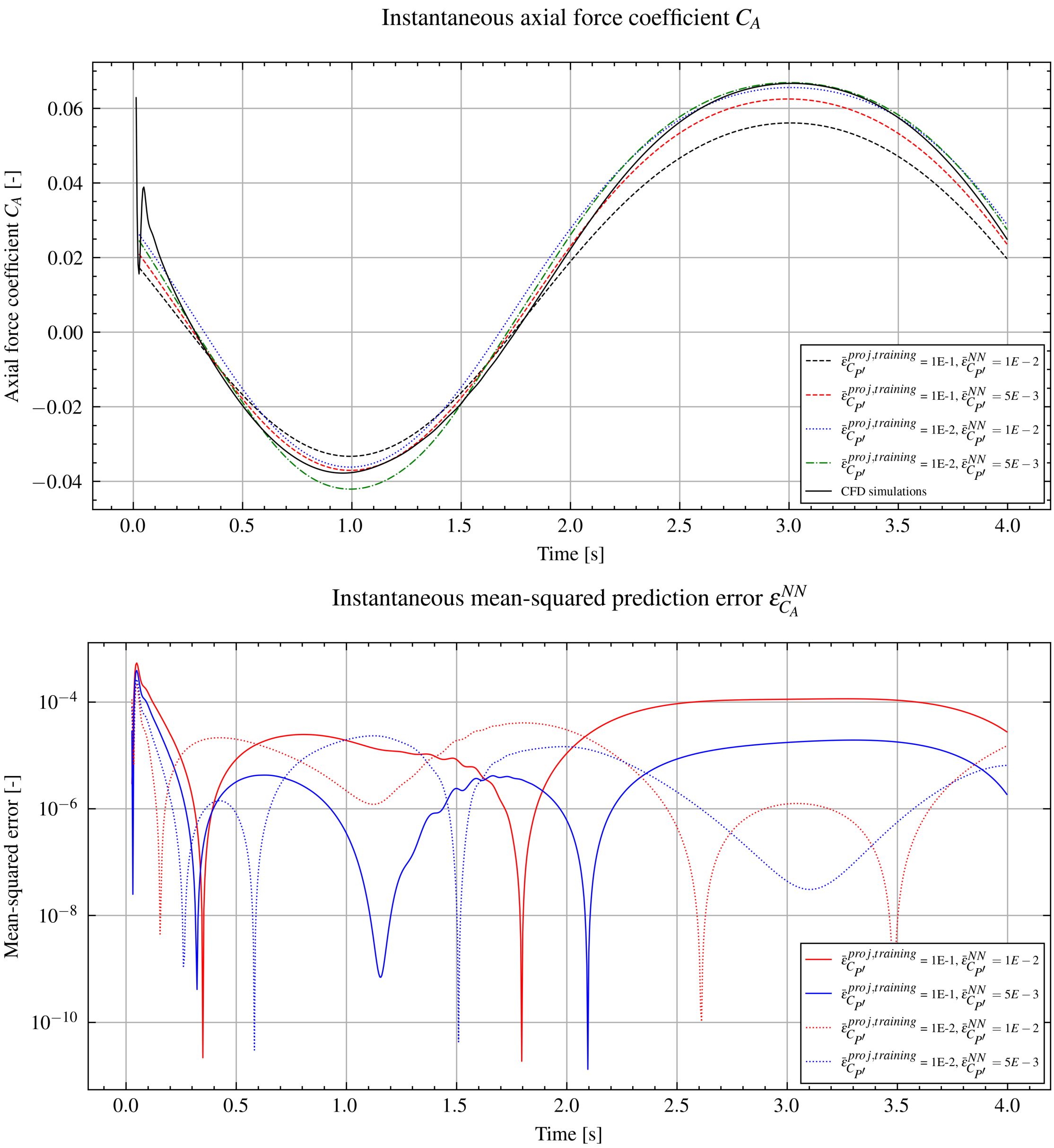


Figure 5.26: POD-LSTM prediction of the axial force coefficient for pitch oscillation $A_0 = 10$ deg , $A = 5$ deg, $f = 0.25$ Hz.

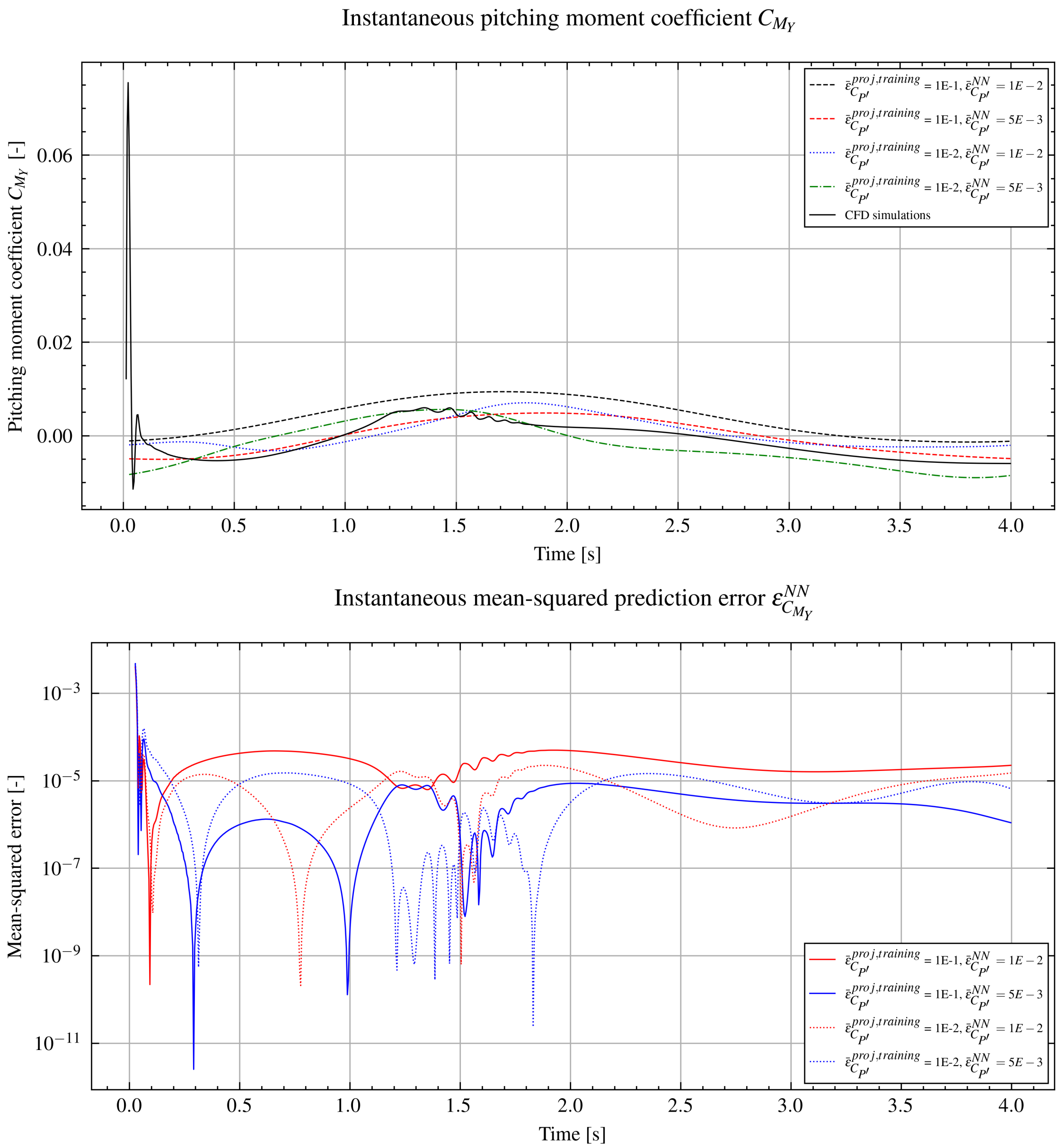
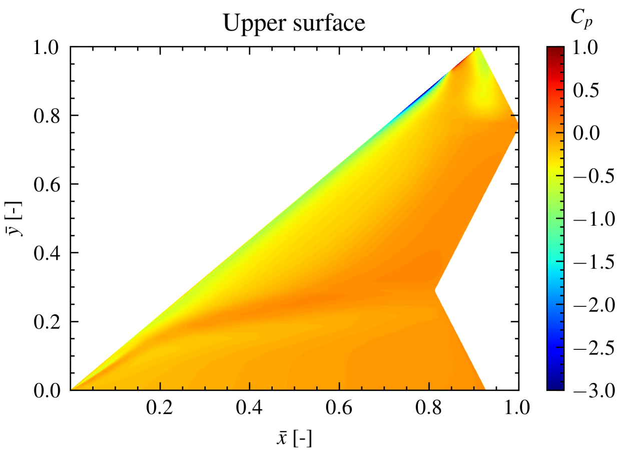
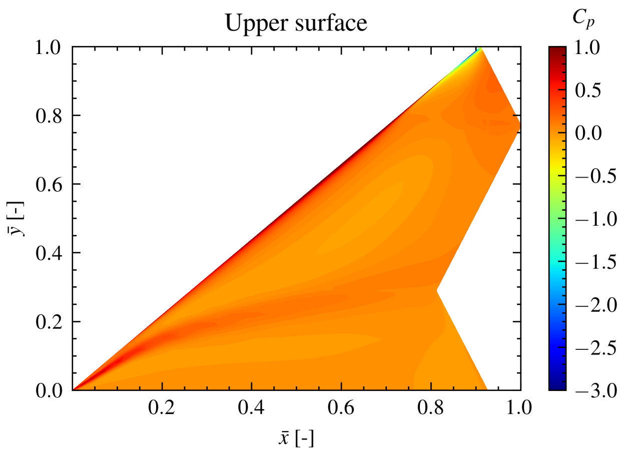


Figure 5.27: POD-LSTM prediction of the pitching moment coefficient for pitch oscillation A_0 = 10 deg , A = 5 deg, f = 0.25 Hz.

In order to identify the deficiencies of the constructed POD-LSTM model, the instantaneous surface pressure coefficients for two notable time steps are investigated. In Figure 5.28 the instantaneous surface pressure coefficients of the upper surface are shown for the CFD simulation at t = 1.1 s and t = 2.1 s. As already discussed during the steady simulation results, the prediction of the upper surface is identified as the main source of errors, hence for illustration purposes, only the upper surface pressure coefficients are taken into account for the analysis. It can be seen from the figures that at these time steps, a leading-edge wing vortex is produced and wingtip separation occurs.



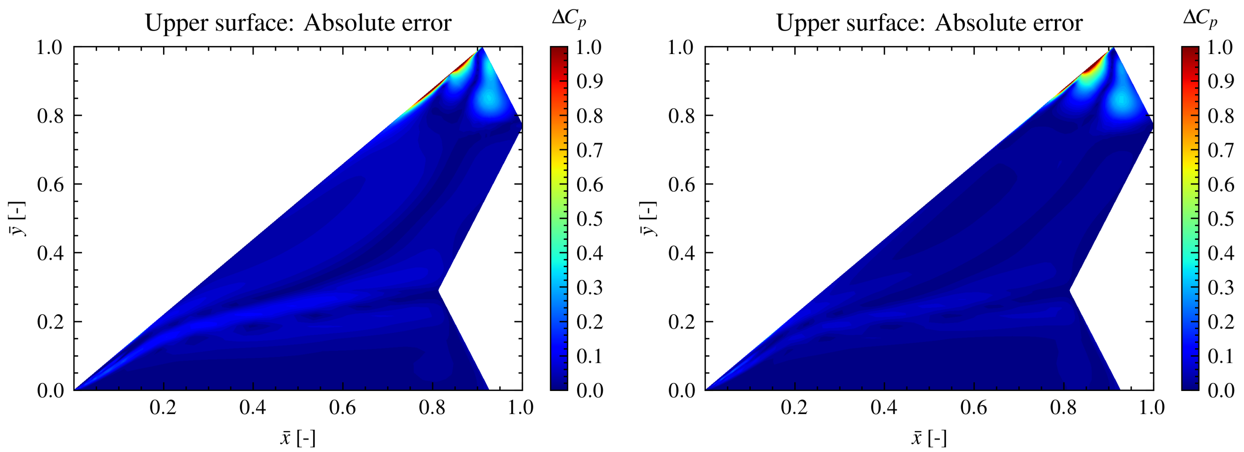
(a) Upper surface pressure distribution at t = 1.1 s.



(b) Upper surface pressure distribution at t = 2.1 s.

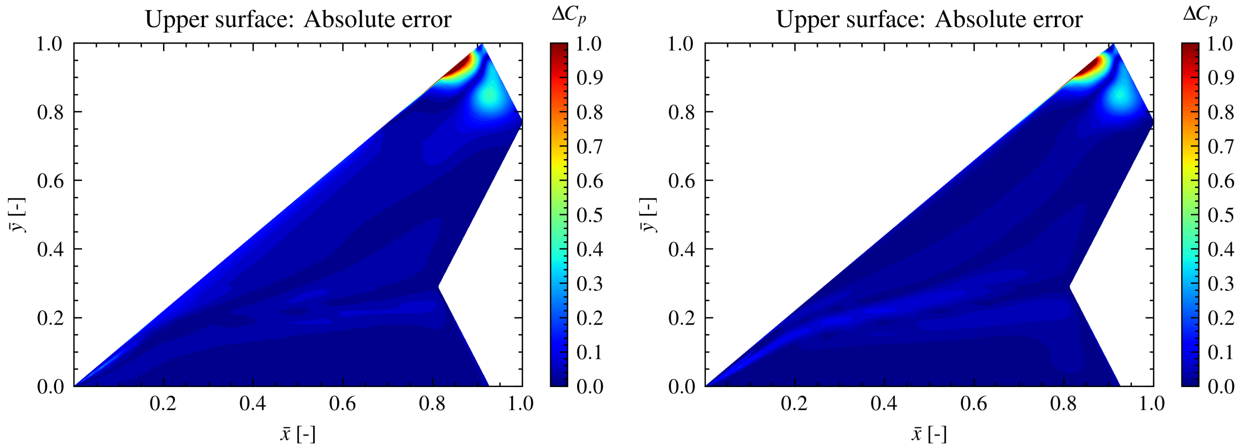
Figure 5.28: Upper surface pressure distribution for pitch oscillation A_0 = 10 deg, A = 5 deg, f = 0.25 Hz at t = 1.1 s and t = 2.1 s obtained from CFD simulations.

In Figure 5.29 and Figure 5.30 the absolute error of the surface pressure coefficient prediction is shown for the different levels of model accuracy for t = 1.1 s and t = 2.1 s, respectively. From both figures, it can be seen that the largest error is found near the tip region of the upper surface of the wing. As described before, and also confirmed by the prediction of the pitching moment coefficient, the inaccurate prediction of the wing tip region is the main cause of the errors in the prediction of the pitching moment coefficient. Furthermore, it can be seen that the error region at the wingtip is larger for models with a lower projection error, i.e., $\bar{\boldsymbol{\epsilon}}^{\boldsymbol{proj,training}}$ = 1E-2. From Figure 5.30 it can be concluded that even a small error region near the wing tip can result in a high prediction error of the pitching moment coefficient. Next to inaccurate prediction of the surface pressure coefficient near the wing tip region, small errors are found at the region where the leading-edge vortex is developed.



(a) Absolute prediction error at $t = 1.1$ s for surrogate model $\bar{\epsilon}_{C_{P'}}^{proj,training} = 1\text{E-}1$, $\bar{\epsilon}_{C_{P'}}^{NN,training} = 1\text{E-}2$.

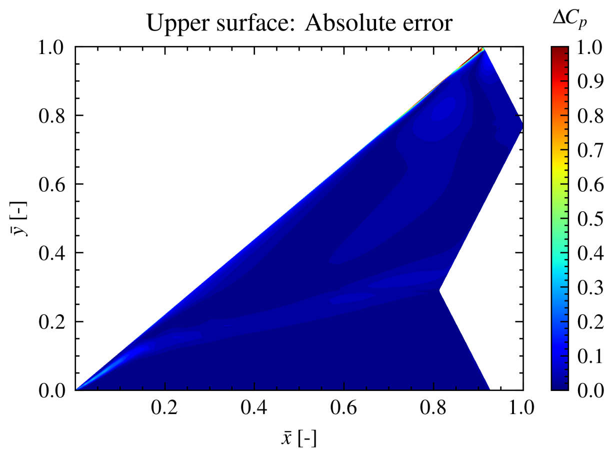
(b) Absolute prediction error at $t = 1.1$ s for surrogate model $\bar{\epsilon}_{C_{P'}}^{proj,training} = 1\text{E-}1$, $\bar{\epsilon}_{C_{P'}}^{NN,training} = 5\text{E-}3$.



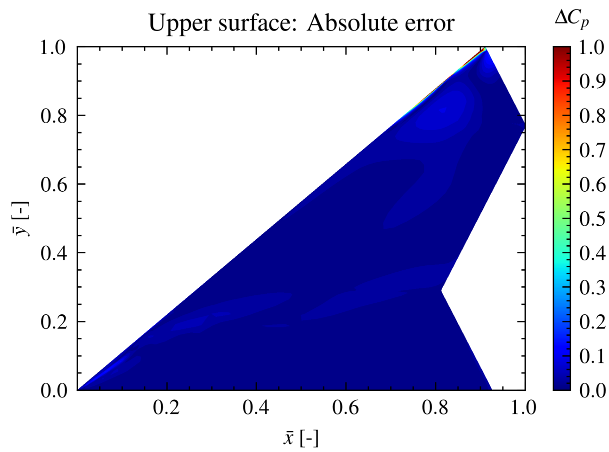
(c) Absolute prediction error at $t = 1.1$ s for surrogate model $\bar{\epsilon}_{C_{P'}}^{proj,training} = 1\text{E-}2$, $\bar{\epsilon}_{C_{P'}}^{NN,training} = 1\text{E-}2$.

(d) Absolute prediction error at $t = 1.1$ s for surrogate model $\bar{\epsilon}_{C_{P'}}^{proj,training} = 1\text{E-}2$, $\bar{\epsilon}_{C_{P'}}^{NN,training} = 5\text{E-}3$.

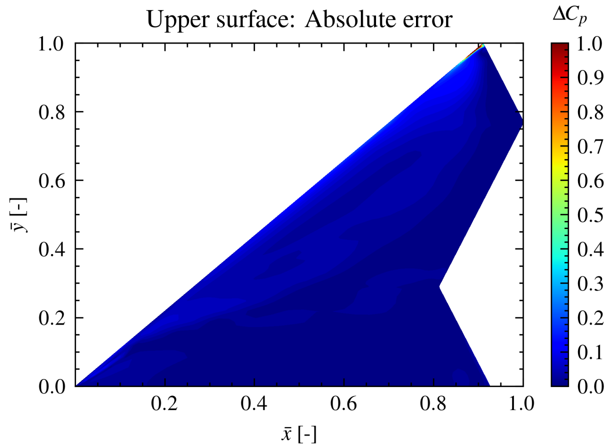
Figure 5.29: POD-LSTM absolute prediction error of the upper surface pressure distribution for different surrogate model settings for pitch oscillation $A_0 = 10$ deg, $A = 5$ deg, $f = 0.25$ Hz at $t = 1.1$ s.



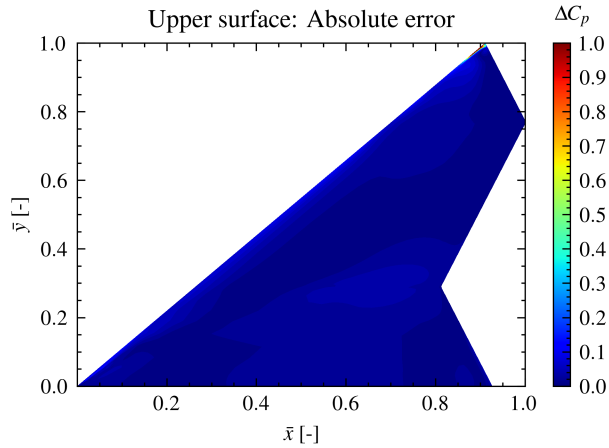
(a) Absolute prediction error at $t = 2.1$ s for surrogate model $\bar{\epsilon}_{C_{p'}}^{proj,training} = 1\text{E-}1$, $\bar{\epsilon}_{C_{p'}}^{NN,training} = 1\text{E-}2$.



(b) Absolute prediction error at $t = 2.1$ s for surrogate model $\bar{\epsilon}_{C_{p'}}^{proj,training} = 1\text{E-}1$, $\bar{\epsilon}_{C_{p'}}^{NN,training} = 5\text{E-}3$.



(c) Absolute prediction error at $t = 2.1$ s for surrogate model $\bar{\epsilon}_{C_{p'}}^{proj,training} = 1\text{E-}2$, $\bar{\epsilon}_{C_{p'}}^{NN,training} = 1\text{E-}2$.



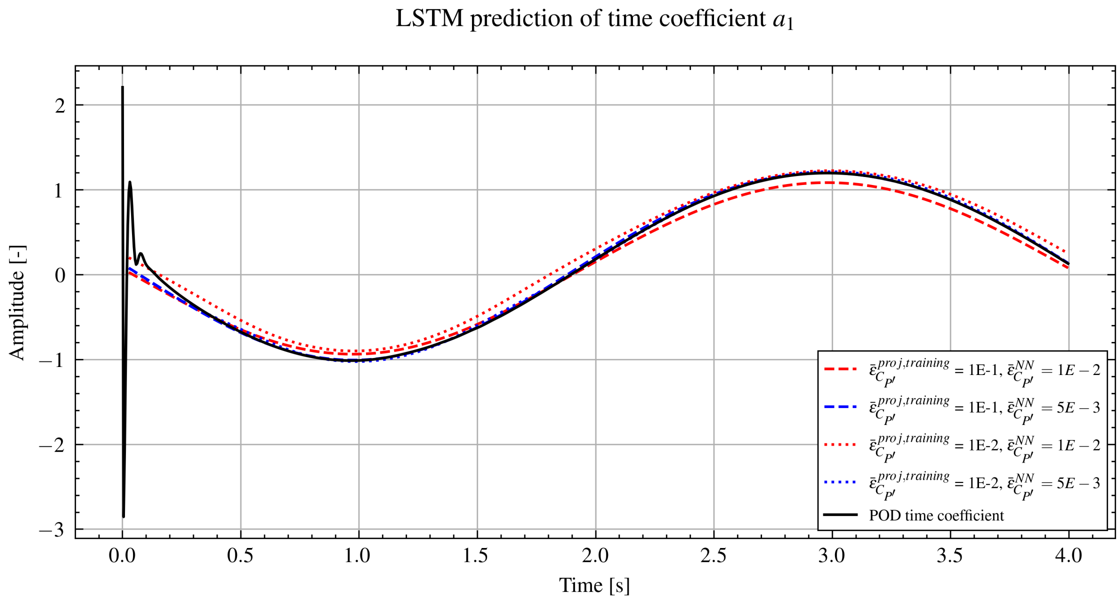
(d) Absolute prediction error at $t = 2.1$ s for surrogate model $\bar{\epsilon}_{C_{p'}}^{proj,training} = 1\text{E-}2$, $\bar{\epsilon}_{C_{p'}}^{NN,training} = 5\text{E-}3$.

Figure 5.30: POD-LSTM absolute prediction error of the upper surface pressure distribution for different surrogate model settings for pitch oscillation $A_0 = 10$ deg, $A = 5$ deg, $f = 0.25$ Hz at $t = 2.1$ s.

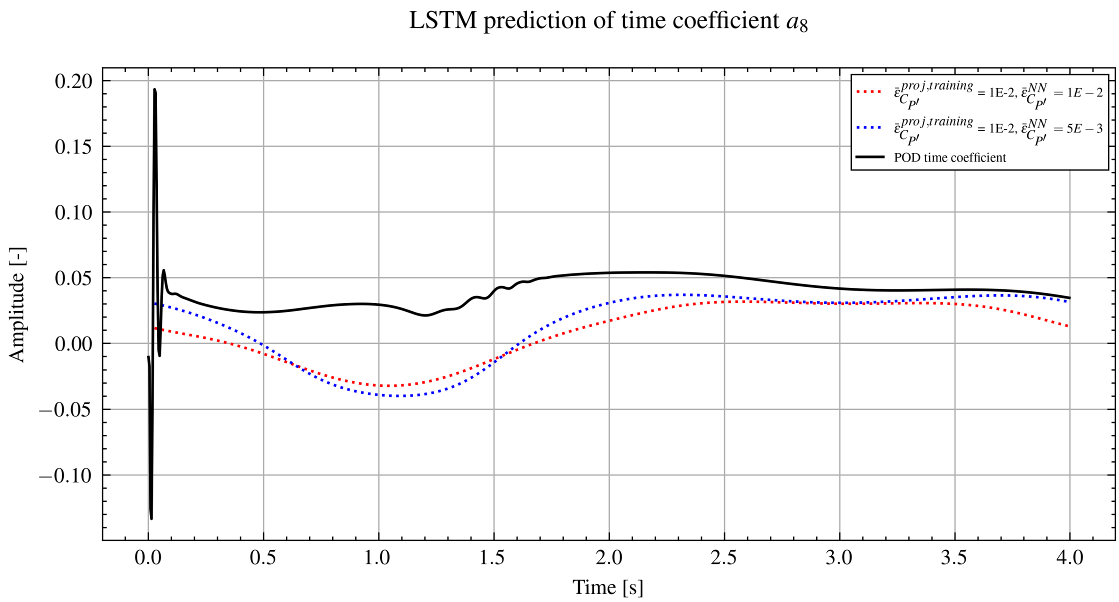
As the accuracy of the POD-LSTM model is directly related to the prediction of the time coefficients for each spatial POD mode, investigation of the prediction of each time coefficient can tell more about the limitations of the current model. In Table 5.10 the NMSE for each predicted time coefficient is shown for the different levels of accuracy of the model. From the table, it can be seen that the time coefficient of the first spatial POD mode is predicted best for all model settings. Furthermore, it can be seen that on average the models with the a higher projection error, i.e., $\bar{\boldsymbol{\epsilon}}_{\boldsymbol{C_{p'}}}^{\boldsymbol{proj,training}} = 1\text{E-}1$, show better results for the prediction of the time coefficients. This means that although the projection error is lower, due to an increase in the number of spatial POD modes used for the , the induced error due to the larger prediction errors for each time coefficient could cause a reconstruction of the surface pressure field with higher errors. For the models with a projection error of $\bar{\boldsymbol{\epsilon}}_{\boldsymbol{C_{p'}}}^{\boldsymbol{proj,training}} = 1\text{E-}2$, the higher ranked time coefficients for the spatial POD modes relatively show larger errors in comparison to the lower ranked time coefficients. The main reason for the inaccurate prediction of higher-ranked time coefficients is an increase in the frequency of each time coefficient due to the origin of the spatial POD modes that are constructed. As the lower ranked spatial POD modes contribute more towards the TKE of the flow, these modes contain more dominant flow properties which are varying less throughout the time domain with respect to higher-ranked spatial POD modes. This variation of flow properties throughout the maneuver can be translated into low and high frequency signals respectively, i.e., lower-ranked spatial POD modes have lower frequency time coefficients than higher-ranked spatial POD modes. It must be noted that the error of the prediction of the time coefficients for the first five spatial POD modes is higher for the models with a lower projection error, i.e., $\bar{\boldsymbol{\epsilon}}_{\boldsymbol{C_{p'}}}^{\boldsymbol{proj,training}} = 1\text{E-}2$. This can be explained as these models incorporate more spatial POD modes and hence need to cover a larger range of frequencies, which results in a prediction that is less accurate for the lower frequency ranges,i.e., the lower-ranked spatial POD modes. The prediction of the time coefficients of the first and eighth spatial POD mode are shown in Figure 5.31.

Table 5.10: NMSE evaluation of the POD-LSTM ROM time coefficient prediction for pitch oscillation A_0 = 10 deg, A = 5 deg, f = 0.25 Hz.

Mode [-]	**NMSE per time coefficient [-]**			
	$\bar{\epsilon}^{proj}_{C_{p'}} = 1E\text{-}1$		$\bar{\epsilon}^{proj}_{C_{p'}} = 1E\text{-}2$	
	$\bar{\epsilon}^{NN}_{C_{p'}} = 1E\text{-}2$	$\bar{\epsilon}^{NN}_{C_{p'}} = 5E\text{-}3$	$\bar{\epsilon}^{NN}_{C_{p'}} = 1E\text{-}2$	$\bar{\epsilon}^{NN}_{C_{p'}} = 5E\text{-}3$
1	0.981263	0.991624	0.979089	0.991343
2	0.033209	0.746636	0.799141	0.570572
3	0.461558	0.556417	0.645733	0.455129
4	0.735079	0.845859	0.841301	0.723033
5	0.208956	0.068972	-0.71327	-0.50738
6			-2.31708	-0.17276
7			0.675327	0.77854
8			-6.83557	-6.23126
9			-2.00816	-2.63086
10			-0.52383	-1.1797
11			0.203696	0.499216
12			0.776713	0.777103
13			-0.04337	-0.22462
14			-0.01448	0.400898
15			-0.45998	-0.87836
16			-0.35598	-0.88312
17			-1.46686	-1.05606
18			0.046836	0.456528
19			-0.70079	0.02242
20			0.180511	0.265406
21			-0.07525	0.10441
22			-0.63752	-0.99657
23			-0.30731	-0.70437
24			-0.47114	-0.70987
25			-0.15808	-0.15973
26			0.345687	0.597983
27			-0.10847	0.194035
28			0.658356	0.602494
29			-0.50314	-0.04392
30			0.014272	0.00649
31			-0.64797	-0.34648
32			-0.44626	-0.51413
33			-0.96068	-0.09198
34			-0.13758	0.011881
35			-0.64277	-0.24847
36			-0.06339	-0.19489
37			-0.28854	-0.18207
38			-1.09002	-0.43355
39			-0.10235	-0.38886
40			0.189914	0.219939
Average [-]	0.48	0.64	-0.39	-0.28



(a) POD-LSTM prediction of time coefficient a_1 for pitch oscillation A_0 = 10 deg , A = 5 deg, f = 0.25 Hz.



(b) POD-LSTM prediction of time coefficient a_8 for pitch oscillation A_0 = 10 deg , A = 5 deg, f = 0.25 Hz.

Figure 5.31: POD-LSTM prediction of time coefficient for different spatial modes for pitch oscillation A_0 = 10 deg , A = 5 deg, f = 0.25 Hz. **Top:** POD-LSTM prediction with high accuracy. **Bottom:** POD-LSTM prediction with low accuracy.

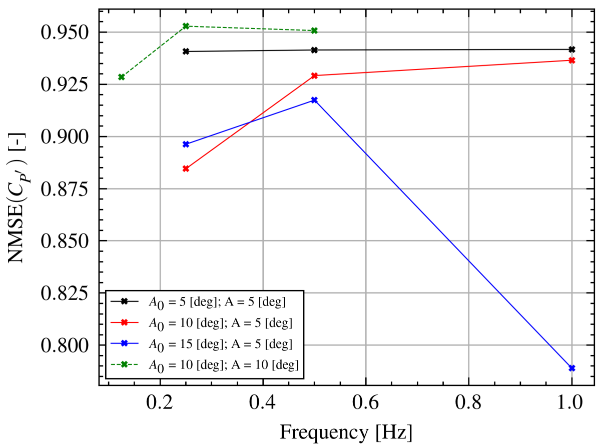
In Table 5.11 an overview is shown of the time-averaged prediction error of the full-order solution and a comparison is provided with respect to the projection error of the constructed reduced-order basis for all harmonic pitch motions. It can be seen from the table that most of the projection errors are smaller than the projection error threshold set for the training maneuver, hence the reduced-order basis is able to accurately capture all dominant flow properties of the test samples within a few spatial POD modes only. Furthermore, it can be seen that the prediction error is relatively close towards the projection error and hence the LSTM surrogate model shows good prediction performance with respect to its reduced-order basis.

Table 5.11: Time-averaged prediction and projection error for harmonic pitch motions in terms of MSE.

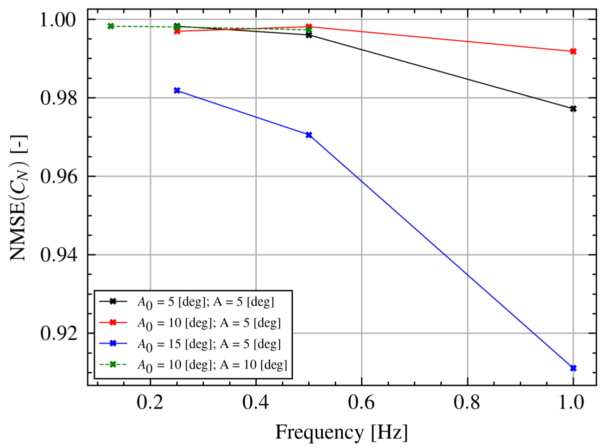
Motion variables			**Prediction error in MSE [-]**				**Projection error in MSE [-]**	
			$\bar{\epsilon}^{proj}_{C_{p'}}$ = 1E-1		$\bar{\epsilon}^{proj}_{C_{p'}}$ = 1E-2		$\bar{\epsilon}^{proj}_{C_{p'}}$ = 1E-1	$\bar{\epsilon}^{proj}_{C_{p'}}$ = 1E-2
Nominal AoA A_0 [deg]	Amplitude A [deg]	Frequency f [Hz]	$\bar{\epsilon}^{NN}_{C_{p'}}$ = 1E-2	$\bar{\epsilon}^{NN}_{C_{p'}}$ = 5E-3	$\bar{\epsilon}^{NN}_{C_{p'}}$ = 1E-2	$\bar{\epsilon}^{NN}_{C_{p'}}$ = 5E-3		
5	**5**	0.25	0.06454	0.05773	0.02252	0.01918	0.05323	0.01535
		0.5	0.06349	0.05686	0.02227	0.01885	0.05329	0.01550
		1	0.06085	0.05475	0.02194	0.01839	0.05403	0.01619
10	**5**	0.25	0.07508	0.06452	0.03842	0.03017	0.07190	0.05210
		0.5	0.07915	0.06654	0.02953	0.02267	0.08604	0.08933
		1	0.08629	0.07285	0.02569	0.02547	0.11741	0.16774
	10	0.125	0.05805	0.05792	0.03604	0.02540	0.04391	0.00975
		0.25	0.05565	0.05169	0.02540	0.01795	0.04577	0.00896
		0.5	0.05550	0.05276	0.02219	0.02101	0.05201	0.00838
15	**5**	0.25	0.07761	0.06658	0.03995	0.03403	0.06448	0.02280
		0.5	0.06853	0.05943	0.02821	0.02278	0.06054	0.01056
		1	0.07274	0.06163	0.02503	0.02929	0.06725	0.01060

Plunge oscillations

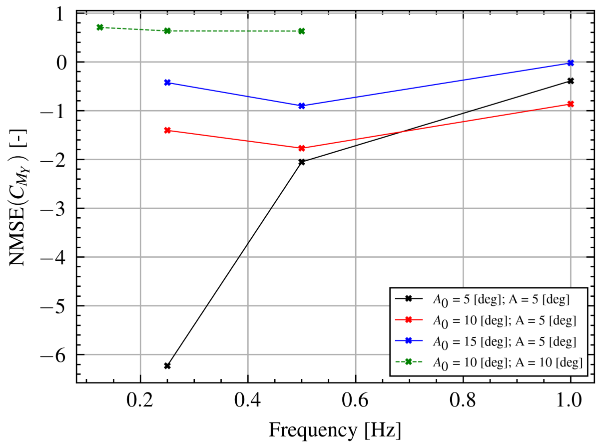
This section will discuss the results of the prediction of the POD-LSTM model for the unsteady harmonic plunge motions. In Figure 5.32 the results are shown for the prediction of the surface pressure coefficients, the normal force coefficient and the pitching moment coefficient in terms of NMSE and are labeled by frequency for each plunge maneuver. The NMSE is measured as error between the prediction of the POD-LSTM model and the CFD simulations. From the figures, it can be seen that on average the POD-LSTM model for plunge maneuvers with a frequency of 0.5 Hz is producing the best results for the prediction of the surface pressure coefficient. Moreover, it can be seen that both the surface pressure coefficients and the normal force coefficient are predicted with higher accuracy than the pitching moment coefficient. It is notable that the harmonic plunge motions with a nominal value of 15 degrees with an amplitude of 5 degrees have the lowest accuracy of all cases. This section will go into the details of the results of the case with the lowest accuracy for the surface pressure coefficient for a frequency of 0.25 Hz. This case has been chosen as it provides good comparison with respect to the pitch maneuver with the same nominal value for the angle of attack and frequency.



(a) LSTM prediction accuracy of the surface pressure coefficient in terms of NMSE for varying frequency.



(b) LSTM prediction accuracy of the normal force coefficient in terms of NMSE for varying frequency.



(c) LSTM prediction accuracy of the pitching moment coefficient in terms of NMSE for varying frequency.

Figure 5.32: LSTM prediction accuracy of the pressure coefficients, normal coefficients, and pitching moment coefficients in terms of NMSE for plunge oscillations at varying nominal values and amplitudes.

To provide good comparison with the previously investigated pitch motion, the harmonic plunge motion with a nominal value of 10 degrees and an amplitude of 5 degrees with a frequency of 0.25 Hz will be investigated. The input state variables as well as the error of the surface pressure coefficient for the given test sample in terms of MSE are shown in Figure 5.33. It can be seen from the figure that for both the models with a high projection error, $\bar{\epsilon}^{proj,training}$ = 1E-1, and low projection error, $\bar{\epsilon}^{proj,training}$ = 1E-2, the error of the surface pressure coefficient prediction, just as for the pitch motion, is relatively high near the initial states of the harmonic plunge motion, which are $\alpha \approx$ 10 deg., $\dot{\alpha} \approx$ (-) 7.5 deg/s and q = 0 deg/s, respectively.

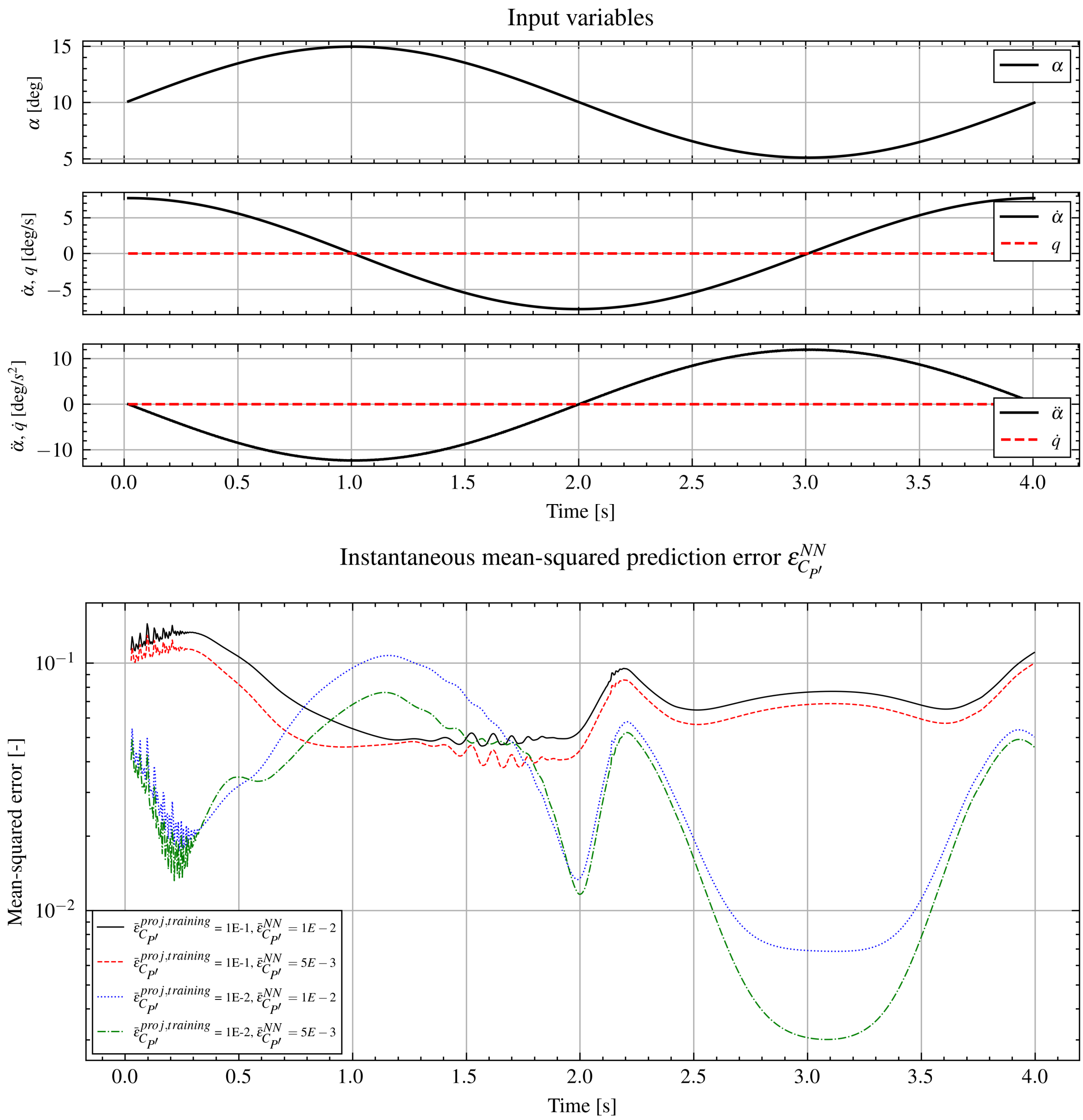


Figure 5.33: POD-LSTM input state variables for plunge oscillation $A_0 = 10$ deg , $A = 5$ deg, $f = 0.25$ Hz.

The prediction of the POD-LSTM model for the normal force coefficient, axial force coefficient and the pitching moment coefficient is shown in Figure 5.34, Figure 5.35 and Figure 5.36, respectively. It can be seen that the normal force coefficient and the axial force coefficient are covered well by the POD-LSTM model with errors that are all in the same order of magnitude, whereas the trend of the pitching moment coefficient is harder to predict and hence less accurate.

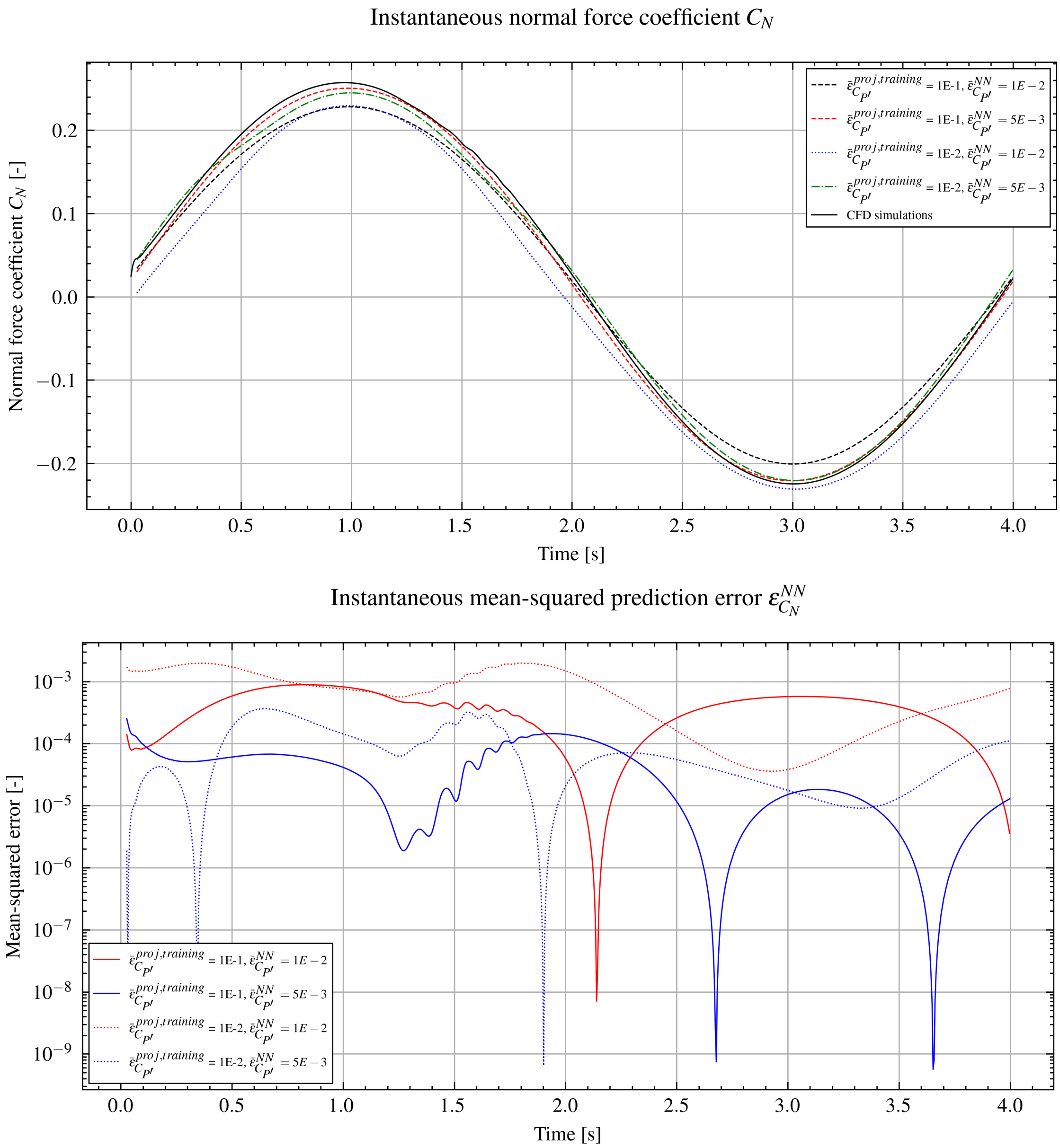


Figure 5.34: POD-LSTM prediction of the normal force coefficient for plunge oscillation $A_0 = 10$ deg , $A = 5$ deg, $f = 0.25$ Hz.

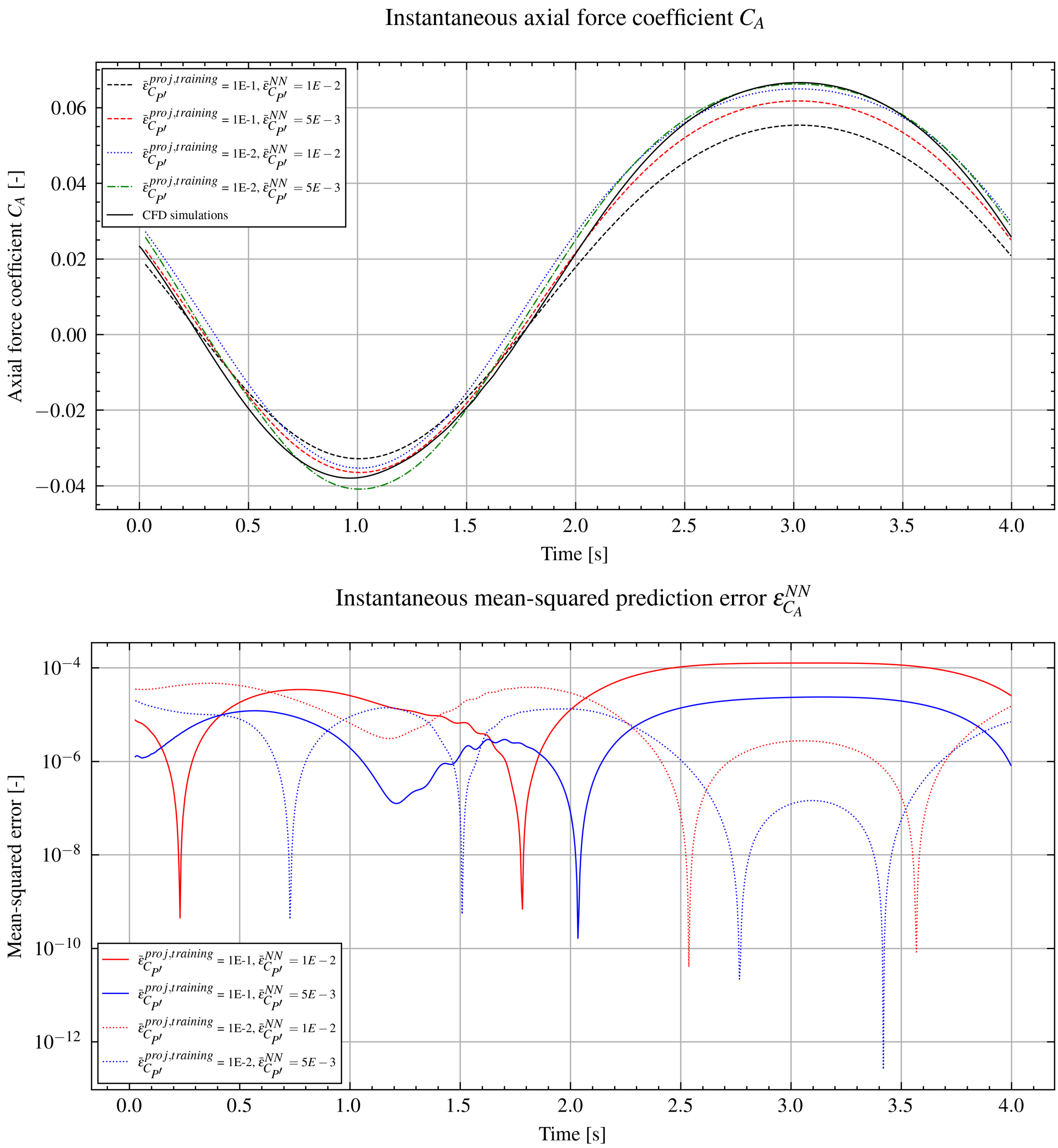


Figure 5.35: POD-LSTM prediction of the axial force coefficient for plunge oscillation $A_0 = 10$ deg , $A = 5$ deg, $f = 0.25$ Hz.

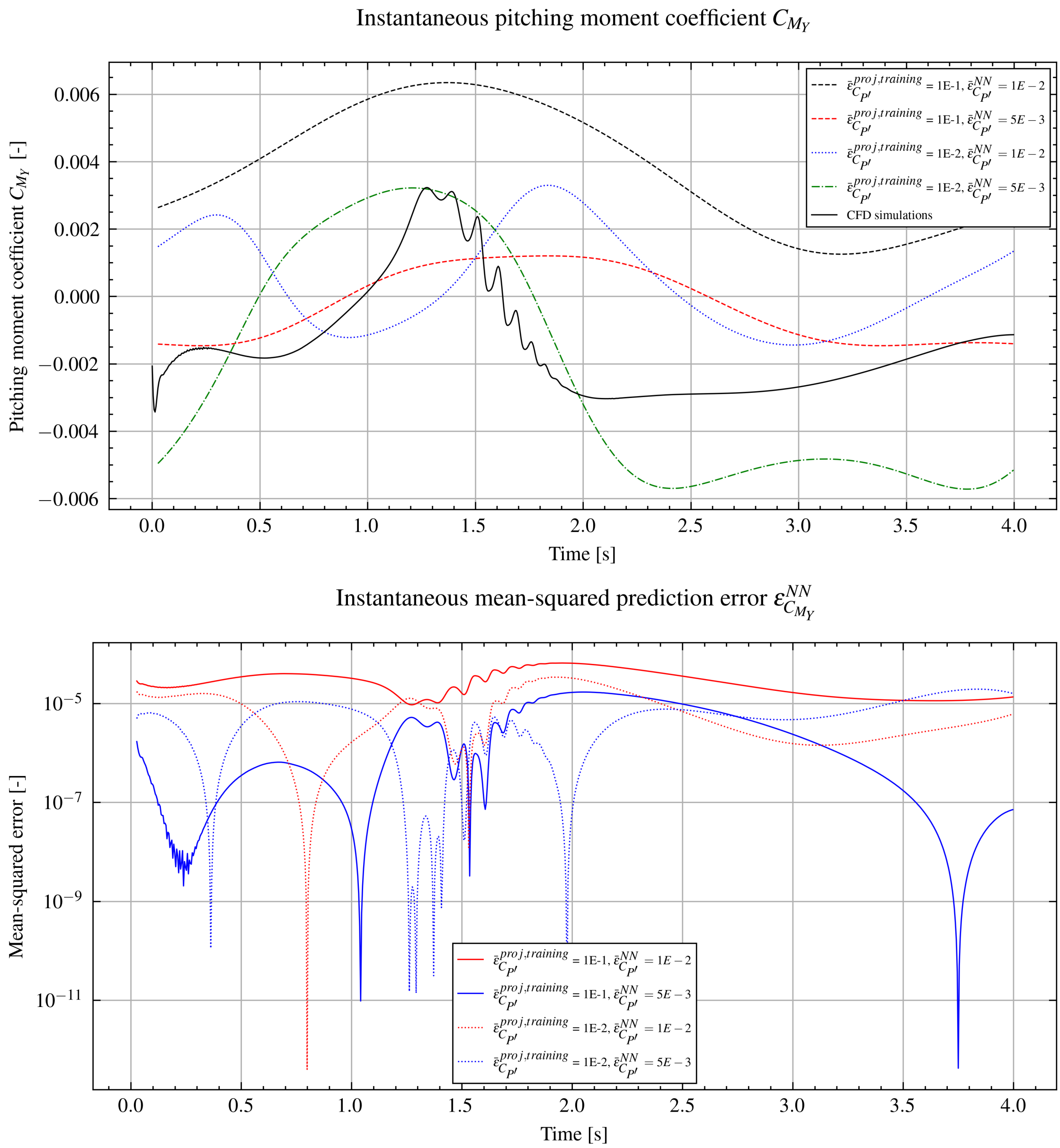
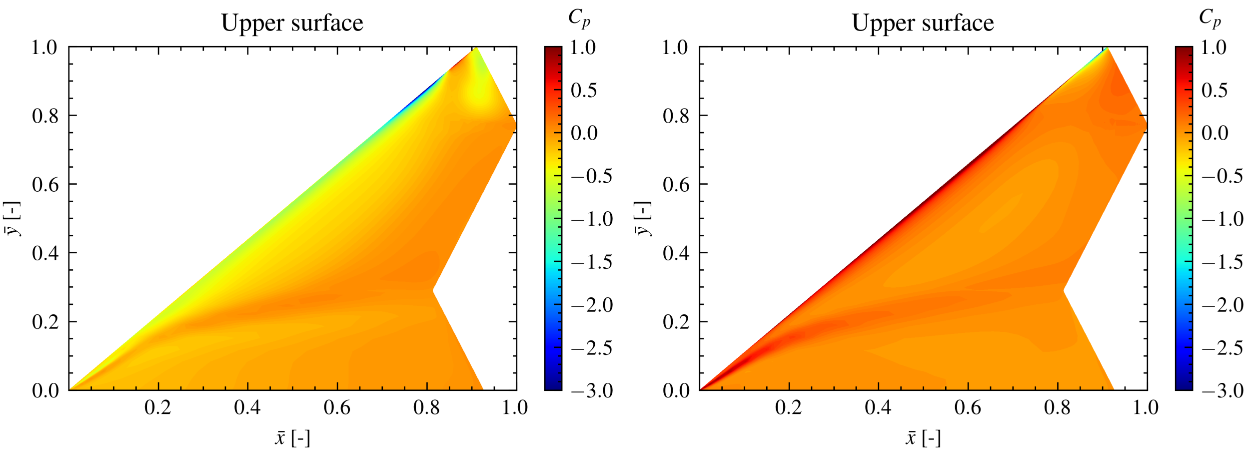


Figure 5.36: POD-LSTM prediction of the pitching moment coefficient for plunge oscillation $A_0 = 10$ deg , $A = 5$ deg, $f = 0.25$ Hz.

The instantaneous surface pressure coefficients for two notable time steps are investigated. In Figure 5.37 the instantaneous surface pressure coefficients of the upper surface are shown for the CFD simulation at $t = 1.1$ s and $t = 2.2$ s. As already discussed during the steady simulation results, the prediction of the upper surface is identified as the main source of errors, hence for illustration purposes only the upper surface pressure coefficients are taken into account for the analysis. It can be seen from the figures that at these time steps, a leading-edge wing vortex is produced and wing tip separation occurs, which are similar flow properties as in the previous discussed pitch maneuver.

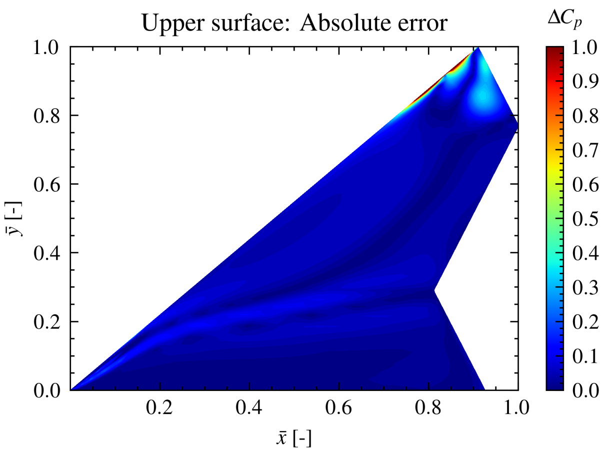


(a) Upper surface pressure distribution at $t = 1.1$ s.

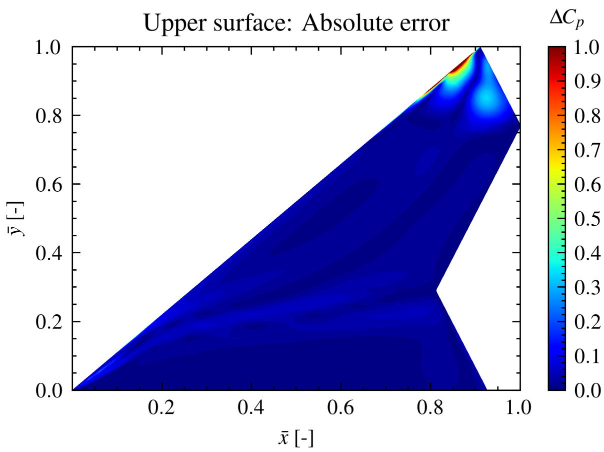
(b) Upper surface pressure distribution at $t = 2.2$ s.

Figure 5.37: Upper surface pressure distribution for plunge oscillation $A_0 = 10$ deg, $A = 5$ deg, $f = 0.25$ Hz at $t = 1.1$ s and $t = 2.2$ s obtained from CFD simulations.

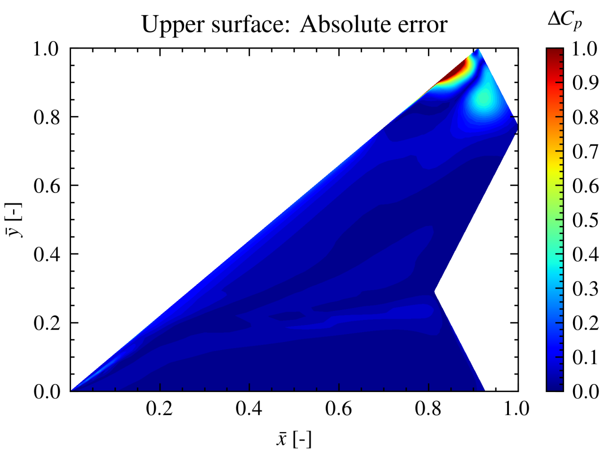
In Figure 5.38 and Figure 5.39 the absolute error of the surface pressure coefficient prediction is shown for the different levels of model accuracy for $t = 1.1$ s and $t = 2.2$ s, respectively. The figures show similar results as for the previously described pitch maneuver, where it can be seen that the largest error is found near the tip region of the upper surface of the wing. Furthermore, it can be seen that the error region at the wing tip is significantly larger for models with a lower projection error, i.e., $\bar{\boldsymbol{\epsilon}}^{\boldsymbol{proj,training}} = 1\text{E-}2$.



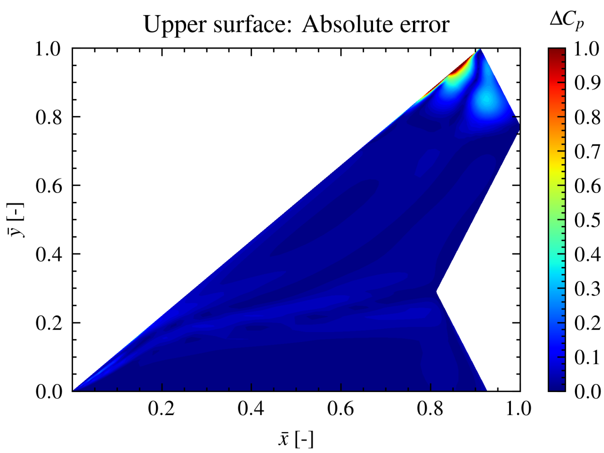
(a) Absolute prediction error at $t = 1.1$ s for surrogate model $\bar{\epsilon}_{C_{P'}}^{proj,training} = 1\text{E-}1$, $\bar{\epsilon}_{C_{P'}}^{NN,training} = 1\text{E-}2$.



(b) Absolute prediction error at $t = 1.1$ s for surrogate model $\bar{\epsilon}_{C_{P'}}^{proj,training} = 1\text{E-}1$, $\bar{\epsilon}_{C_{P'}}^{NN,training} = 5\text{E-}3$.

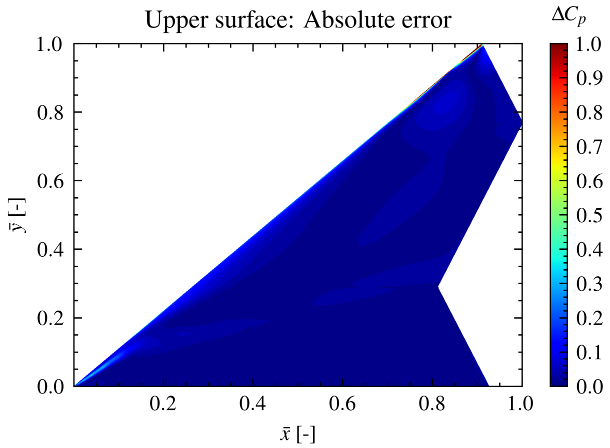


(c) Absolute prediction error at $t = 1.1$ s for surrogate model $\bar{\epsilon}_{C_{P'}}^{proj,training} = 1\text{E-}2$, $\bar{\epsilon}_{C_{P'}}^{NN,training} = 1\text{E-}2$.

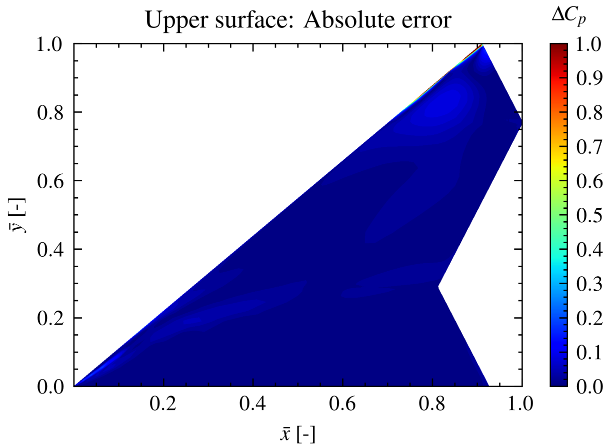


(d) Absolute prediction error at $t = 1.1$ s for surrogate model $\bar{\epsilon}_{C_{P'}}^{proj,training} = 1\text{E-}2$, $\bar{\epsilon}_{C_{P'}}^{NN,training} = 5\text{E-}3$.

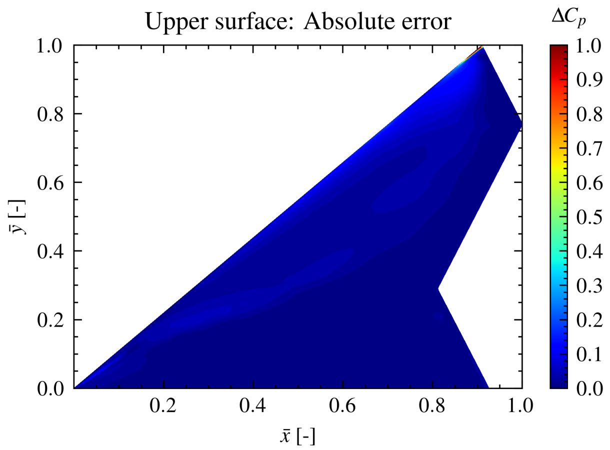
Figure 5.38: POD-LSTM absolute prediction error of the upper surface pressure distribution for different surrogate model settings for plunge oscillation $A_0 = 10$ deg, $A = 5$ deg, $f = 0.25$ Hz at $t = 1.1$ s.



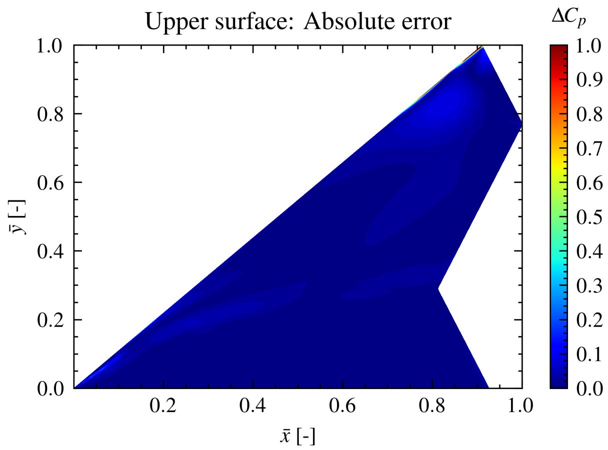
(a) Absolute prediction error at $t = 2.2$ s for surrogate model $\bar{\epsilon}^{proj,training}_{C_{p'}} = $ 1E-1, $\bar{\epsilon}^{NN,training}_{C_{p'}} = $ 1E-2.



(b) Absolute prediction error at $t = 2.2$ s for surrogate model $\bar{\epsilon}^{proj,training}_{C_{p'}} = $ 1E-1, $\bar{\epsilon}^{NN,training}_{C_{p'}} = $ 5E-3.



(c) Absolute prediction error at $t = 2.2$ s for surrogate model $\bar{\epsilon}^{proj,training}_{C_{p'}} = $ 1E-2, $\bar{\epsilon}^{NN,training}_{C_{p'}} = $ 1E-2.



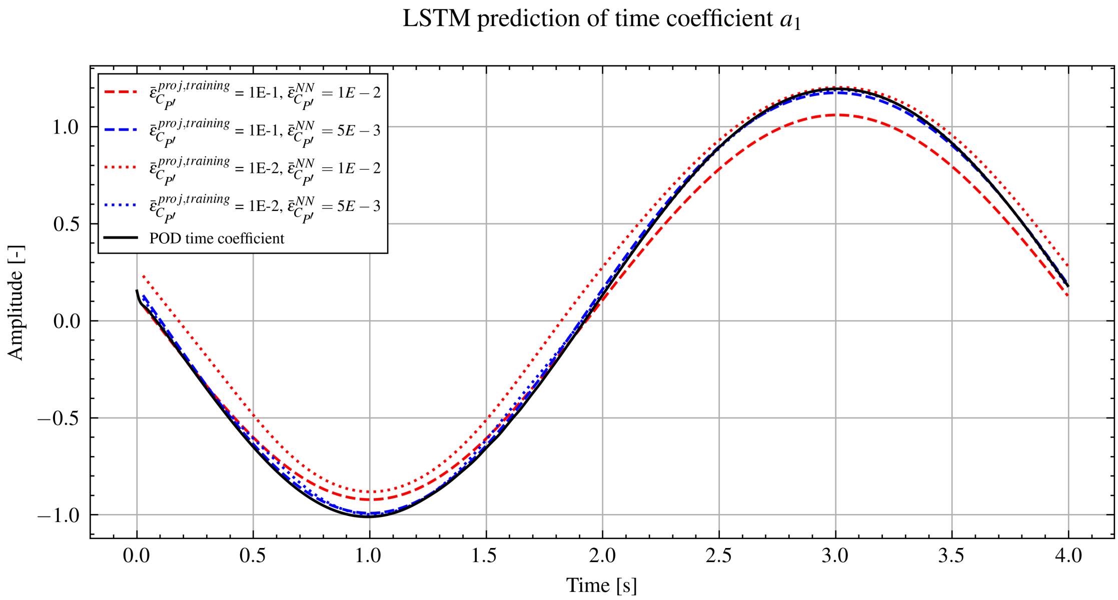
(d) Absolute prediction error at $t = 2.2$ s for surrogate model $\bar{\epsilon}^{proj,training}_{C_{p'}} = $ 1E-2, $\bar{\epsilon}^{NN,training}_{C_{p'}} = $ 5E-3.

Figure 5.39: POD-LSTM absolute prediction error of the upper surface pressure distribution for different surrogate model settings for plunge oscillation $A_0 = 10$ deg, $A = 5$ deg, $f = 0.25$ Hz at $t = 2.2$ s.

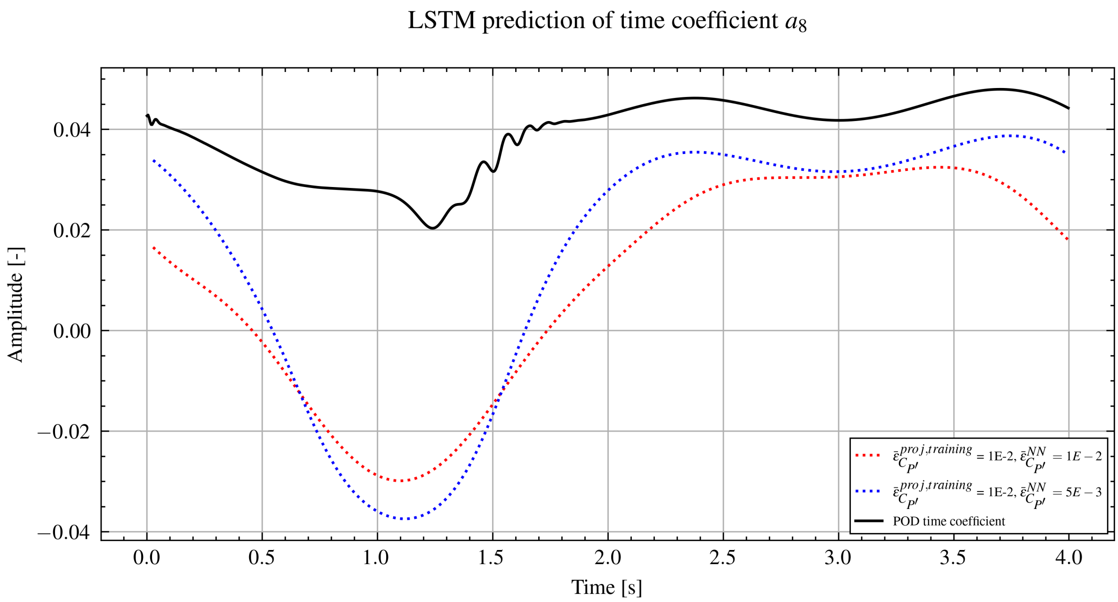
In Table 5.12 the results of the prediction of the time coefficients in terms of NMSE are shown for the plunge maneuver. Similar as to the pitch maneuver, it can be seen that the prediction of the time coefficient for the first spatial POD mode shows the best results, whereas the prediction of the time coefficient for the eight spatial mode shows the worst results. Furthermore, just as for the pitch maneuver the same trend is seen in an increase in the error for higher ranked spatial POD modes. However, it must be noted that the average errors of the predicted time coefficients for the plunge maneuver are lower in comparison to the pitch maneuver. The predicted time coefficients are shown in Figure 5.40 for the first and the eighth spatial POD mode.

Table 5.12: NMSE evaluation of the POD-LSTM ROM time coefficient prediction for plunge oscillation $A_0 = 10$ deg, $A = 5$ deg, $f = 0.25$ Hz.

Mode [-]	**NMSE per time coefficient [-]**			
	$\bar{\epsilon}^{proj}_{C_{p'}} = 1\text{E-}1$		$\bar{\epsilon}^{proj}_{C_{p'}} = 1\text{E-}2$	
	$\bar{\epsilon}^{NN}_{C_{p'}} = 1\text{E-}2$	$\bar{\epsilon}^{NN}_{C_{p'}} = 5\text{E-}3$	$\bar{\epsilon}^{NN}_{C_{p'}} = 1\text{E-}2$	$\bar{\epsilon}^{NN}_{C_{p'}} = 5\text{E-}3$
1	0.987597	0.999456	0.979027	0.999141
2	0.002314	0.906767	0.959135	0.727625
3	0.533132	0.756888	0.887484	0.588762
4	0.844971	0.983046	0.991609	0.856213
5	0.384702	-0.0025	-1.36612	-1.16012
6			-2.7072	-0.02696
7			0.652951	0.798574
8			-17.3595	-15.3851
9			-2.42486	-3.0963
10			-0.81407	-1.69329
11			0.221225	0.794415
12			0.864203	0.850571
13			-0.07308	-0.28556
14			-0.03451	0.405687
15			-0.4484	-0.81783
16			-0.276	-0.83969
17			-2.52954	-1.79902
18			0.088711	0.628851
19			-1.5935	-0.0167
20			0.119574	0.209105
21			-0.07169	0.153527
22			-0.68162	-1.01259
23			-0.31696	-0.69602
24			-0.56934	-0.99178
25			-0.56383	-0.44433
26			0.407143	0.698398
27			-0.1211	0.225404
28			0.644257	0.577546
29			-0.94987	-0.06137
30			0.002612	-0.17331
31			-1.13471	-0.63137
32			-0.45923	-0.57248
33			-1.89943	-0.21893
34			-0.09427	0.064395
35			-0.77327	-0.18432
36			0.265912	-0.58942
37			-0.49429	-0.34554
38			-1.17515	-0.46745
39			-0.14763	-0.52536
40			0.190386	0.240615
Average [-]	0.55	0.73	-0.80	-0.58



(a) POD-LSTM prediction of time coefficient a_1 for plunge oscillation A_0 = 10 deg , A = 5 deg, f = 0.25 Hz.



(b) POD-LSTM prediction of time coefficient a_8 for plunge oscillation A_0 = 10 deg , A = 5 deg, f = 0.25 Hz.

Figure 5.40: POD-LSTM prediction of time coefficient for different spatial modes for plunge oscillation A_0 = 10 deg , A = 5 deg, f = 0.25 Hz. **Top:** POD-LSTM prediction of time coefficient with high accuracy. **Bottom:** POD-LSTM prediction of time coefficients with low accuracy.

In Table 5.13 an overview is shown of the time-averaged prediction error of the full-order solution and a comparison is provided with respect to the projection error of the constructed reduced-order basis for all harmonic plunge motions. The results are similar as to the harmonic pitch motions, with the most notable result that the projection error is in the same orders of magnitude or even lower than the set threshold for the projection errors of the training maneuver.

Table 5.13: Time-averaged prediction and projection error for harmonic plunge motions in terms of MSE.

Motion variables			**Prediction error in MSE [-]**				**Projection error in MSE [-]**	
			$\bar{\epsilon}^{proj,training}_{C_{p'}}$ = 1E-1		$\bar{\epsilon}^{proj,training}_{C_{p'}}$ = 1E-2		$\bar{\epsilon}^{proj,training}_{C_{p'}}$ = 1E-1	$\bar{\epsilon}^{proj}_{C_{p'}}$ = 1E-2
Nominal AoA A_0 [deg]	Amplitude A [deg]	Frequency f [Hz]	$\bar{\epsilon}^{NN}_{C_{p'}}$ = 1E-2	$\bar{\epsilon}^{NN}_{C_{p'}}$ = 5E-3	$\bar{\epsilon}^{NN}_{C_{p'}}$ = 1E-2	$\bar{\epsilon}^{NN}_{C_{p'}}$ = 5E-3		
5	**5**	0.25	0.06510	0.05823	0.02262	0.01927	0.05333	0.01538
		0.5	0.06485	0.05810	0.02251	0.01891	0.05366	0.01552
		1	0.06457	0.05819	0.02285	0.01855	0.05540	0.01609
10	**5**	0.25	0.07390	0.06371	0.03918	0.03094	0.05960	0.01420
		0.5	0.07731	0.06451	0.02626	0.01951	0.06108	0.01321
		1	0.08644	0.07110	0.01800	0.01843	0.06790	0.01204
	10	0.125	0.05793	0.05769	0.03546	0.02493	0.04409	0.00989
		0.25	0.05619	0.05197	0.02458	0.01708	0.04620	0.00945
		0.5	0.06181	0.05223	0.01731	0.01928	0.05237	0.00900
15	**5**	0.25	0.07937	0.06881	0.04019	0.03464	0.06487	0.02401
		0.5	0.08049	0.06731	0.03459	0.02948	0.06696	0.02286
		1	0.12850	0.10914	0.07273	0.08828	0.12669	0.12885

5.2.3. POD-LSTM model performance

To assess the performance of the POD-LSTM model and to provide a comparison with previous studies, the model performance in terms of prediction time is discussed in this section. The assessment of the model performance is done by predicting the training samples, where the maneuver has been repeated 5 times total in order to provide an accurate measurement of the prediction time. Furthermore, an assessment is done for the prediction time for varying hyperparameters of the model. It must be noted that only hyperparameters are included that influence the model complexity or the amount of input data that is used for the model prediction, hence the batch size is not considered during this assessment. Each prediction is done 5 times for each hyperparameter in order to avoid the influence of background processes during the run. This section will discuss the total maneuver prediction time as well as the prediction time for each time step. The assessment has been done using the specifications that were shown in Table 5.2.

Variation of LSTM layers

In Figure 5.41 the computational prediction time of the full maneuver is shown for varying number of LSTM layers. From the figure it can be seen that the prediction time almost linearly increases with increasing number of LSTM layers. The overall prediction time results, including the results per time step are shown in Table 5.14.

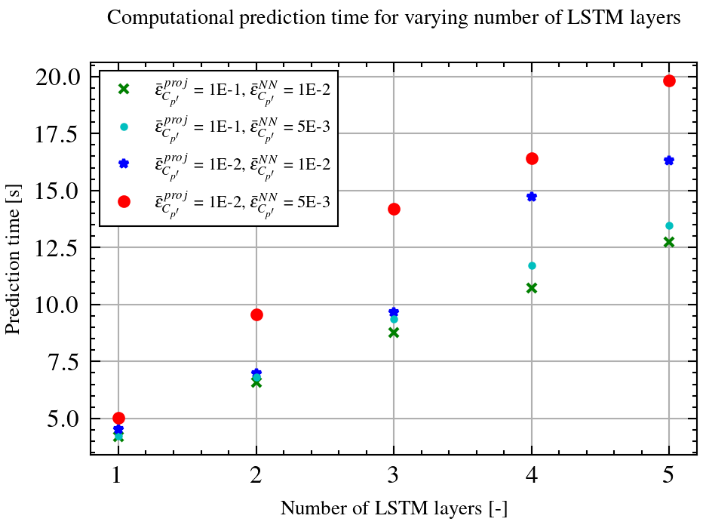


Figure 5.41: Computational prediction time for varying number of LSTM layers.

Table 5.14: Prediction results for a varying number of LSTM layers

LSTM layers	$\bar{\epsilon}_{C_{p'}}^{proj,training}$ [-]	$\bar{\epsilon}_{C_{p'}}^{NN}$ [-]	Total prediction time [s]	Prediction time per time step [s]
1	**1.00E-01**	1.00E-02	4.191268	0.000148
		5.00E-03	4.224438	0.000149
	1.00E-02	1.00E-02	4.52364	0.00016
		5.00E-03	5.022778	0.000177
2	**1.00E-01**	1.00E-02	6.573286	0.000232
		5.00E-03	6.827334	0.000241
	1.00E-02	1.00E-02	6.966079	0.000246
		5.00E-03	9.557676	0.000337
3	**1.00E-01**	1.00E-02	8.766485	0.000309
		5.00E-03	9.357569	0.00033
	1.00E-02	1.00E-02	9.677004	0.000342
		5.00E-03	14.21434	0.000502
4	**1.00E-01**	1.00E-02	10.71022	0.000378
		5.00E-03	11.72726	0.000414
	1.00E-02	1.00E-02	14.72005	0.00052
		5.00E-03	16.42994	0.00058
5	**1.00E-01**	1.00E-02	12.75674	0.00045
		5.00E-03	13.47559	0.000476
	1.00E-02	1.00E-02	16.31249	0.000576
		5.00E-03	19.83834	0.0007

Variation of LSTM units

In Figure 5.42 the computational prediction time of the full maneuver is shown for varying number of LSTM layers. From the figure it can be seen that the prediction time almost linearly increases with increasing number of LSTM units. The overall prediction time results, including the results per time step, are shown in Table 5.15.

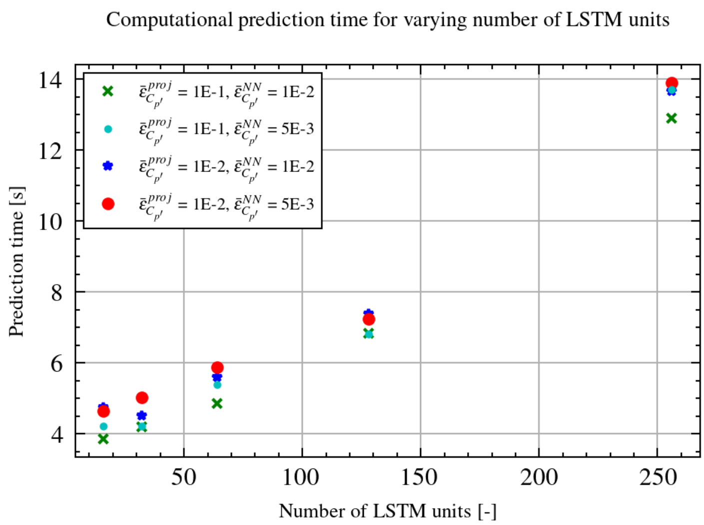


Figure 5.42: Computational prediction time for varying number of LSTM units.

Table 5.15: Prediction results for a varying number of LSTM units

LSTM units	$\bar{\epsilon}^{proj,training}_{C_{p'}}$ [-]	$\bar{\epsilon}^{NN}_{C_{p'}}$ [-]	Total prediction time [s]	Prediction time per time step [s]
16	**1.00E-01**	1.00E-02	6.573286	0.000232
		5.00E-03	6.827334	0.000241
	1.00E-02	1.00E-02	4.755023	0.000168
		5.00E-03	4.652098	0.000164
32	**1.00E-01**	1.00E-02	4.191268	0.000148
		5.00E-03	4.224438	0.000149
	1.00E-02	1.00E-02	4.52364	0.00016
		5.00E-03	5.022778	0.000177
64	**1.00E-01**	1.00E-02	8.766485	0.000309
		5.00E-03	9.357569	0.00033
	1.00E-02	1.00E-02	5.611828	0.000198
		5.00E-03	5.87677	0.000207
128	**1.00E-01**	1.00E-02	10.71022	0.000378
		5.00E-03	11.72726	0.000414
	1.00E-02	1.00E-02	7.392733	0.000261
		5.00E-03	7.250718	0.000256
256	**1.00E-01**	1.00E-02	12.75674	0.00045
		5.00E-03	13.47559	0.000476
	1.00E-02	1.00E-02	13.67016	0.000483
		5.00E-03	13.90889	0.000491

Variation of dense layers

In Figure 5.43 the computational prediction time of the full maneuver is shown for varying numbes of dense layers. From the figure, it can be seen that the prediction time is increasing slightly for an increasing number of dense layers. The overall prediction time results, including the results per time step, are shown in Table 5.16.

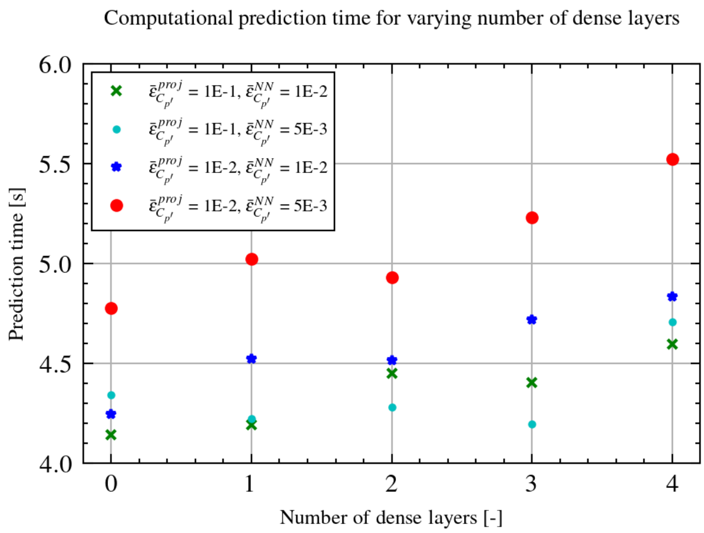


Figure 5.43: Computational prediction time for varying number of dense layers.

Table 5.16: Prediction results for a varying number of dense layers

Dense layers	$\bar{\epsilon}^{proj,training}_{C_{p'}}$ [-]	$\bar{\epsilon}^{NN}_{C_{p'}}$ [-]	Total prediction time [s]	Prediction time per time step [s]
0	**1.00E-01**	1.00E-02	4.140147	0.000146
		5.00E-03	4.3416	0.000153
	1.00E-02	1.00E-02	4.2474	0.00015
		5.00E-03	4.777703	0.000169
1	**1.00E-01**	1.00E-02	4.191268	0.000148
		5.00E-03	4.224438	0.000149
	1.00E-02	1.00E-02	4.52364	0.00016
		5.00E-03	5.022778	0.000177
2	**1.00E-01**	1.00E-02	4.450955	0.000157
		5.00E-03	4.279335	0.000151
	1.00E-02	1.00E-02	4.514106	0.000159
		5.00E-03	4.929297	0.000174
3	**1.00E-01**	1.00E-02	4.403034	0.000155
		5.00E-03	4.194825	0.000148
	1.00E-02	1.00E-02	4.718713	0.000167
		5.00E-03	5.229906	0.000185
4	**1.00E-01**	1.00E-02	4.597823	0.000162
		5.00E-03	4.708853	0.000166
	1.00E-02	1.00E-02	4.833574	0.000171
		5.00E-03	5.522656	0.000195

Variation of dense units

In Figure 5.44 the computational prediction time of the full maneuver is shown for varying number of dense units. From the figure it can be seen that the prediction time is increasing slightly for an increasing number of dense units, however the differences are significantly small that the effect of varying the dense units is negligible for computational prediction time . The overall prediction time results, including the results per time step are shown in Table 5.17.

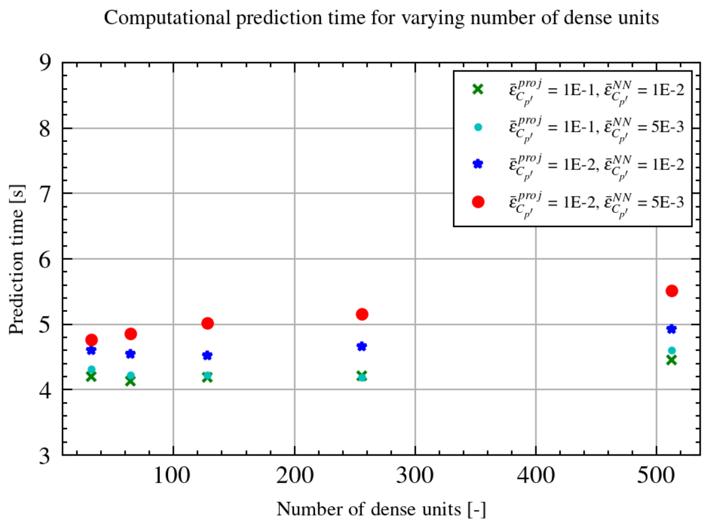


Figure 5.44: Computational prediction time for varying number of dense units.

Table 5.17: Prediction results for a varying number of dense units

Dense units	$\bar{\epsilon}_{C_{p'}}^{proj,training}$ [-]	$\bar{\epsilon}_{C_{p'}}^{NN}$ [-]	**Total prediction time [s]**	**Prediction time per time step [s]**
32	**1.00E-01**	1.00E-02	4.19401	0.000148
		5.00E-03	4.318153	0.000152
	1.00E-02	1.00E-02	4.598414	0.000162
		5.00E-03	4.76999	0.000168
64	**1.00E-01**	1.00E-02	4.130597	0.000146
		5.00E-03	4.218606	0.000149
	1.00E-02	1.00E-02	4.542324	0.00016
		5.00E-03	4.856077	0.000171
128	**1.00E-01**	1.00E-02	4.191268	0.000148
		5.00E-03	4.224438	0.000149
	1.00E-02	1.00E-02	4.52364	0.00016
		5.00E-03	5.022778	0.000177
256	**1.00E-01**	1.00E-02	4.212829	0.000149
		5.00E-03	4.18315	0.000148
	1.00E-02	1.00E-02	4.660517	0.000165
		5.00E-03	5.16251	0.000182
512	**1.00E-01**	1.00E-02	4.448591	0.000157
		5.00E-03	4.600008	0.000162
	1.00E-02	1.00E-02	4.925752	0.000174
		5.00E-03	5.510708	0.000195

Variation of time steps used

In Figure 5.45 the computational prediction time of the full maneuver is shown for varying number of time steps used. From the figure it can be seen that the prediction time shows a small linear increase for an increasing number of time steps used. The overall prediction time results, including the results per time step, are shown in Table 5.18.

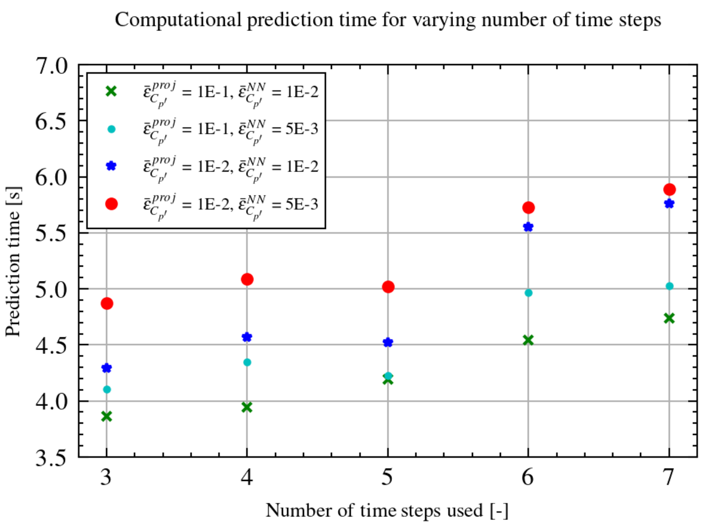


Figure 5.45: Computational prediction time for a varying number of time steps used.

Table 5.18: Prediction results for varying time steps used

Time steps	$\bar{\epsilon}_{C_{p'}}^{proj,training}$ [-]	$\bar{\epsilon}_{C_{p'}}^{NN}$ [-]	Total prediction time [s]	Prediction time per time step [s]
3	**1.00E-01**	1.00E-02	3.864684	0.000136
		5.00E-03	4.104628	0.000145
	1.00E-02	1.00E-02	4.293692	0.000152
		5.00E-03	4.873513	0.000172
4	**1.00E-01**	1.00E-02	3.940593	0.000139
		5.00E-03	4.345948	0.000153
	1.00E-02	1.00E-02	4.567063	0.000161
		5.00E-03	5.086767	0.00018
5	**1.00E-01**	1.00E-02	4.191268	0.000148
		5.00E-03	4.224438	0.000149
	1.00E-02	1.00E-02	4.52364	0.00016
		5.00E-03	5.022778	0.000177
6	**1.00E-01**	1.00E-02	4.545966	0.00016
		5.00E-03	4.965896	0.000175
	1.00E-02	1.00E-02	5.552729	0.000196
		5.00E-03	5.72535	0.000202
7	**1.00E-01**	1.00E-02	4.737366	0.000167
		5.00E-03	5.026316	0.000177
	1.00E-02	1.00E-02	5.761844	0.000203
		5.00E-03	5.886796	0.000208

POD-LSTM model performance evaluation

The results as discussed above should provide the user of the POD-LSTM model insights in the variation of the given hyperparameters for the computational prediction time of the model. From the results, it can be concluded that the variation of the LSTM layers and units have the largest effect on the computational prediction time, whereas the differences between minimum and maximum prediction time are up to a factor of 3. For all the other hyperparameters differences are significantly small that it can be considered that they have no effect on the computational prediction time of the POD-LSTM model. In Section 5.2.3 an overview of the minimum and maximum computational prediction time is found for the variance of the hyperparameters as well as a comparison is provided with respect to the high-fidelity simulations and the previous constructed surrogate model as described by Papp [18]. It can be seen from the table that on average the prediction time is similar for all levels of accuracy. Furthermore, it can be seen that the prediction time of the constructed POD-LSTM models is significantly smaller than of the previous constructed CNN by Papp, by up to a factor of 60. This can be explained as the proposed model in this work, just as for training of the model, only makes use of the reduced-order basis and hence the complexity of the model can be significantly decreased with respect to previous studies.

Table 5.19: POD-LSTM computational prediction time comparison with respect to high-fidelity simulations and previous constructed surrogate model from D. Papp [18]

Model	$\bar{\epsilon}_{C_{p'}}^{proj,training}$ [-]	$\bar{\epsilon}_{C_{p'}}^{NN}$ [-]	Minimum prediction time per time step	Maximum prediction time per time step	Average prediction time per time step
POD-LSTM (@2.4 GHz)	**1.00E-01**	1.00E-02	0.14 ms	0.45 ms	0.21 ms
		5.00E-03	0.15 ms	0.48 ms	0.21 ms
	1.00E-02	1.00E-02	0.15 ms	0.58 ms	0.21 ms
		5.00E-03	0.15 ms	0.71 ms	0.24 ms
ENSOLV (Training samples)					6×10^6 ms / 76×10^3 ms (@80 threads)
CNN (Papp @2.6 GHz)		**7.00E-05**			15 ms

5.3. Summary of the POD-LSTM model results

This chapter discussed the simulation results of the proposed POD-LSTM model. The results were split up in two different sections: the offline stage results and the online stage results.

Overall, the offline stage results have shown that the high-fidelity samples that are used for the construction of the reduced-order basis provide good regressor space coverage and the projection error of most of the test samples is in the same order of magnitude as the training samples. Furthermore, variance on the reduced-order basis was investigated to determine the effects on the use of different spatial POD modes, that can either be constructed from new training samples or can be altered from the original reduced-order basis. Dependent on the application purpose of this POD-LSTM model, variation of the reduced-order basis has shown that it can be beneficial for the integral load projection of the surface pressure coefficient or the pitching moment coefficient. The training process of the LSTM surrogate model has been evaluated and it has shown that the number of LSTM units used in the model has the highest influence on the computational training time, whereas if the parameters of the model are optimal the computational training time can be reduced to an average of 4 minutes. The limitations of the reduced-order basis were investigated and it was shown that for lower projection errors the region of error is highlighted near the tip of the UCAV configuration. However, in general the projection error of the reduced-order basis is small for both constructed reduced-order basis and limitations of using the POD method are therefore negligible.

For the online stage results, two different simulations were investigated: steady and unsteady simulations. The steady simulations were performed for the angle of attack range between 0 - 20 degrees, to identify both linear and non-linear phenomena. The results of the prediction of the steady simulations have shown accurate results for the normal force coefficient and the axial force coefficient for all level of model accuracy settings. However, the pitching moment coefficient was predicted inaccurately due to the inaccurate surface pressure coefficient near the wing tip region. It must be noted that the model with a higher projection error, i.e., a reduced-order basis with fewer spatial POD modes showed a better accuracy for the prediction of the pitching moment coefficient. For the unsteady results, harmonic pitch and plunge motions were investigated. These cases are introduced in order to identify the deficiencies of the POD-LSTM model within the regressor space of the high-fidelity training samples. The results for both the harmonic pitch and plunge motions are similar, whereas the surface pressure coefficient, the normal force coefficient, and the axial force coefficient are predicted with high accuracy. However, just as with the steady simulation results, the results of the pitching moment coefficient are inaccurate due to inaccurate prediction of the surface pressure coefficient of the wing tip region. The results have shown that even small errors of the surface pressure coefficient near the wing tip could lead to high errors of the pitching moment coefficient. For both harmonic pitch and plunge motions the errors of the time coefficients are presented and it is shown that the prediction of the time coefficients for the first spatial POD mode shows the highest accuracy, whereas the accuracy decreases with increasing rank of the spatial POD mode. The model performance in terms of computational cost has been investigated and it is shown that the number of LSTM layers has the highest influence on the prediction time of the POD-LSTM model. However, it must be noted that the prediction time of the proposed POD-LSTM model is faster than previous studies, which is mainly due to the reduced complexity of the surrogate model.

As an addition to the unsteady harmonic motions, also a special maneuver has been investigated. The results of this maneuver are shown in Appendix A. This maneuver was investigated to determine the performance of the POD-LSTM model near the bounds of the regressor space. From the results it is shown that the accuracy of the prediction of the POD-LSTM model near the bounds of the state variables tends to increase massively. Therefore, the POD-LSTM model can only be used within the bounds of the regressor space coverage and does not serve as extrapolation model.

6

Conclusion and recommendations

In this thesis, a CFD data-driven Reduced-Order Model (ROM) is constructed for the prediction of the surface pressure and the aerodynamic load coefficients of the MULDICON UCAV configuration. The motivation for this thesis work was to identify possible solutions for the drawbacks of previous studies that were performed to predict the surface pressure distributions of the UCAV configuration. These solutions include the incorporation of time-history effects using Recurrent Neural Network (RNN) architectures and the possibility to impose boundary conditions on the flow using Proper Orthogonal Decomposition (POD).

This work proposed a POD-LSTM model that makes use of a reduced-order basis that is constructed from high-fidelity CFD samples and has been used to predict the surface pressure coefficient distribution of the MULDICON UCAV configuration. A systematic approach was used in order to evaluate the model performance of the POD-LSTM model in terms of accuracy and computational training time and computational prediction time. Both evaluations of the offline and online stage were done and the results of this evaluation were presented in this thesis report. This chapter will conclude the results that were found and will propose further recommendations for the POD-LSTM model that can be used in future studies.

6.1. Conclusions

This section will discuss the conclusions that followed from the results for the prediction of the surface pressure coefficients using a POD-LSTM model. The conclusions will be presented in two different parts that will discuss the difference between the offline stage of the model and the online stage of the model.

The conclusions that can be drawn from the offline stage results are as follows:

- The high-fidelity samples that are obtained from CFD simulations, which were defined using the Schroeder sweep, provide good regressor coverage for the construction of a reduced-order basis using the POD method. The projection of the full-order solutions on the constructed reduced-order basis of the test samples has been investigated and has shown an accuracy that is in the same order of magnitude as the accuracy of the training samples. Furthermore, the limitations of the reduced-order basis have been investigated and it was shown that the projection errors only cover a small region of the leading edge near the wingtip. Although these errors could lead to a high contribution of the error of the pitching moment coefficient, the region of error is small and the limitations on the use of a reduced-order basis are minimal.
- Variations of the reduced-order basis have been investigated in order to determine the effect of alternative spatial POD modes on the projection error. It was shown that the alternative reduced-order basis, where the spatial POD modes were altered to provide more emphasis on the tip region, provided lower projection errors for both the surface pressure coefficient and the pitching moment coefficient, while the number of spatial POD modes used for this reduced-order basis was constant.
- A systematic approach was used in order to be able to evaluate the performance of the POD-LSTM model. The benefit of using the POD method for the POD-LSTM model is that errors between the offline and online stage can be identified separately, which can give the user of the model insight into

possible improvements. Furthermore, a sensitivity analysis has been performed to determine the effect of varying the model parameters for the computational training time of the POD-LSTM model. It was shown that the number of LSTM units in the model has a significant effect on the computational training time and should be considered as a parameter of interest in future modeling approaches. Furthermore, to achieve an accuracy of 5E-3 in terms of MSE for the prediction of surface pressure coefficient for the training samples, a minimum of 4 minutes of computational training time was found. This time is significantly lower than the time that was needed to produce the high-fidelity training samples. Due to time constraints, a fair comparison between previous studies was not possible. However, it was shown that the POD-LSTM model shows potential to be a good competitor with respect to previous studies in terms of computational expense.

The following conclusions can be drawn from the online stage results:

- The full-order steady CFD simulation results are compared with the POD-LSTM predictions. Here, it was shown that the normal force coefficient and the axial force coefficient are both well captured by the POD-LSTM model for all angles of attack. However, the pitching moment coefficient shows inaccurate predictions due to an inaccurate prediction of the surface pressure coefficients near the wing tip region. It was shown that the models with a higher projection error, i.e., the reduced-order basis with less spatial POD modes, showed a better accuracy for the pitching moment coefficient. Although it is expected that a reduced-order basis with a lower projection error shows better results for the prediction of the surface pressure coefficient, this was not the case for the steady results. As more time coefficients need to be predicted for a model with a lower projection error, the accuracy of the time coefficients of the first spatial POD modes is less accurate in comparison to a model with a higher projection error. It was shown that the frequency of the time coefficients increases with an increasing rank of the spatial POD modes. This makes it harder to predict the range of frequencies for the time coefficients if the number of POD modes used is increased.

- Unsteady harmonic oscillations were investigated and compared to the full-order solution. From the results, it was shown that most of the maneuvers are covered well by the POD-LSTM model, where the projection error for most maneuvers is below the threshold that was set for the training samples. Furthermore, it was shown that the time-averaged prediction errors for the surface pressure coefficients are in the same order of magnitude as the projection errors. Hence, it can be concluded that the POD-LSTM model is able to predict the test samples well. Similar to the steady results, it was shown that the pitching moment coefficient shows the highest prediction errors for all cases due to an inaccurate prediction of the surface pressure coefficient near the wing tip region. The highest errors of the prediction of the surface pressure coefficient are shown where the regressor coverage is minimal. Hence, more training samples are needed in this region to improve the reduced-order basis for the surface pressure coefficient.

- The computational prediction time of the POD-LSTM model was evaluated for varying model parameters. It was shown that the number of LSTM units and LSTM layers have the highest influence on the computational prediction time. Although these model parameters have a relatively high contribution towards the computational prediction time, the overall prediction time of the POD-LSTM model in comparison to previous studies is significantly lower, by up to a factor of 60.

Overall, the POD-LSTM model shows promising results for the prediction of the surface pressure coefficients and the integral aerodynamic load coefficients. In particular, the normal force coefficient and the axial force coefficient of the UCAV MULDICON configuration for both steady and unsteady simulations show promise. The drawback of this model in comparison to previous studies is the inaccurate prediction of the surface pressure coefficient near the wing tip region, which causes high errors of the pitching moment coefficient. As a reduced-order basis is used for the construction of the model, errors between the offline and online stage of the model can be identified easily and model complexity can be reduced significantly. This results in lower computational expense for both the training process and the prediction of the model. This research serves as a baseline for the construction of a POD-LSTM model that will provide knowledge on the influence of different model parameters. The full capabilities of the POD-LSTM model still need to be exploited in future studies, in which longer training time, higher prediction accuracy, and different training samples are suggested.

6.2. Recommendations: Improvement of the POD-LSTM model

As this thesis work only serves as a baseline for the construction of the POD-LSTM model, further research is necessary in order to exploit the full potential of this proposed reduced-order modeling approach. From the current work, the following recommendations are proposed for incorporation in future studies:

- To improve the surface pressure coefficient prediction near the wing tip region, a variance of the current reduced-order basis can be used. As can be seen from the offline stage results, a variance of the reduced-order basis that incorporated a linear weighted relationship towards the tip region for each spatial POD mode showed a decrease in the projection error for both the surface pressure coefficient and the pitching moment coefficient. Although this reduced-order basis was already investigated in the offline stage, they have not been used in the online stage of the POD-LSTM model yet. In addition, a second POD-LSTM model can be used to predict the region of the wingtip that makes use of a reduced-order basis with a larger number of spatial POD modes used. This method is known as domain decomposition [27] and, in combination with the original POD-LSTM model, can provide solutions for the inaccurate pressure prediction near the wingtip.
- To improve the prediction of the integral load coefficients, it is suggested to consider the integral load coefficients in the prediction of the POD-LSTM model. Currently, the integral load coefficients are calculated from the predicted time coefficients that are based on the surface pressure distribution of each individual mode. As the shear stress of the UCAV is not included, the prediction of the integral load coefficients can be inaccurate. Furthermore, if the time coefficients are predicted inaccurately, they can have a large influence on the integral load coefficients. As the surrogate model only makes use of time-sequential data, the integral load coefficients can easily be implemented in the prediction of the model to avoid errors that are imposed due to inaccurate predictions of the time coefficients.
- As the prediction of the time coefficients are directly linked to the error of the surface pressure coefficients, the error of the time coefficient prediction could be incorporated in the loss function of the POD-LSTM model. Currently, the loss function is only based on the projection of the predicted time coefficients on a reduced-order basis. A weighted relationship can be used to increase the accuracy of prediction of the time coefficients of the higher-ranked spatial POD modes, resulting in better prediction of the surface pressure coefficient and the integral aerodynamic loads.
- The scope of the POD-LSTM model is limited due to the simplified MULDICON UCAV configuration. As the proposed POD-LSTM model only makes use of the state variables as input, the model can easily incorporate more features such as the deflection of ailerons and elevators. To extend the scope of the current POD-LSTM model, a more detailed model of the MULDICON UCAV can be used.

A

Special simulations: Sharp pitch-up maneuver

Next to the harmonic pitch and plunge motions, another type of maneuver is investigated: the sharp pitch-up maneuver. This maneuver will be discussed as it will provide the user of the model information about the performance of the POD-LSTM model towards the bounds of the regressor space. Furthermore, this particular maneuver has been used in previous studies and provides good comparison to the POD-LSTM model. The input state variables as well as the error of the surface pressure coefficient for the given sharp pitch-up maneuver in terms of MSE are shown in Figure A.1. It can be seen from the figure that for both the models with a high projection error, $\bar{\boldsymbol{\epsilon}}^{\boldsymbol{proj,training}}$ = 1E-1, and low projection error, $\bar{\boldsymbol{\epsilon}}^{\boldsymbol{proj,training}}$ = 1E-2, the error of the surface pressure coefficient prediction is relatively high at the time instantaneous moments where the pitch rate changes. Furthermore, it can be seen that the differences between the surface pressure coefficient prediction for both projection errors, i.e., $\bar{\boldsymbol{\epsilon}}^{\boldsymbol{proj,training}}$ = 1E-1 and $\bar{\boldsymbol{\epsilon}}^{\boldsymbol{proj,training}}$ = 1E-2, are relatively small, whereas the model with the highest accuracy settings shows the best results.

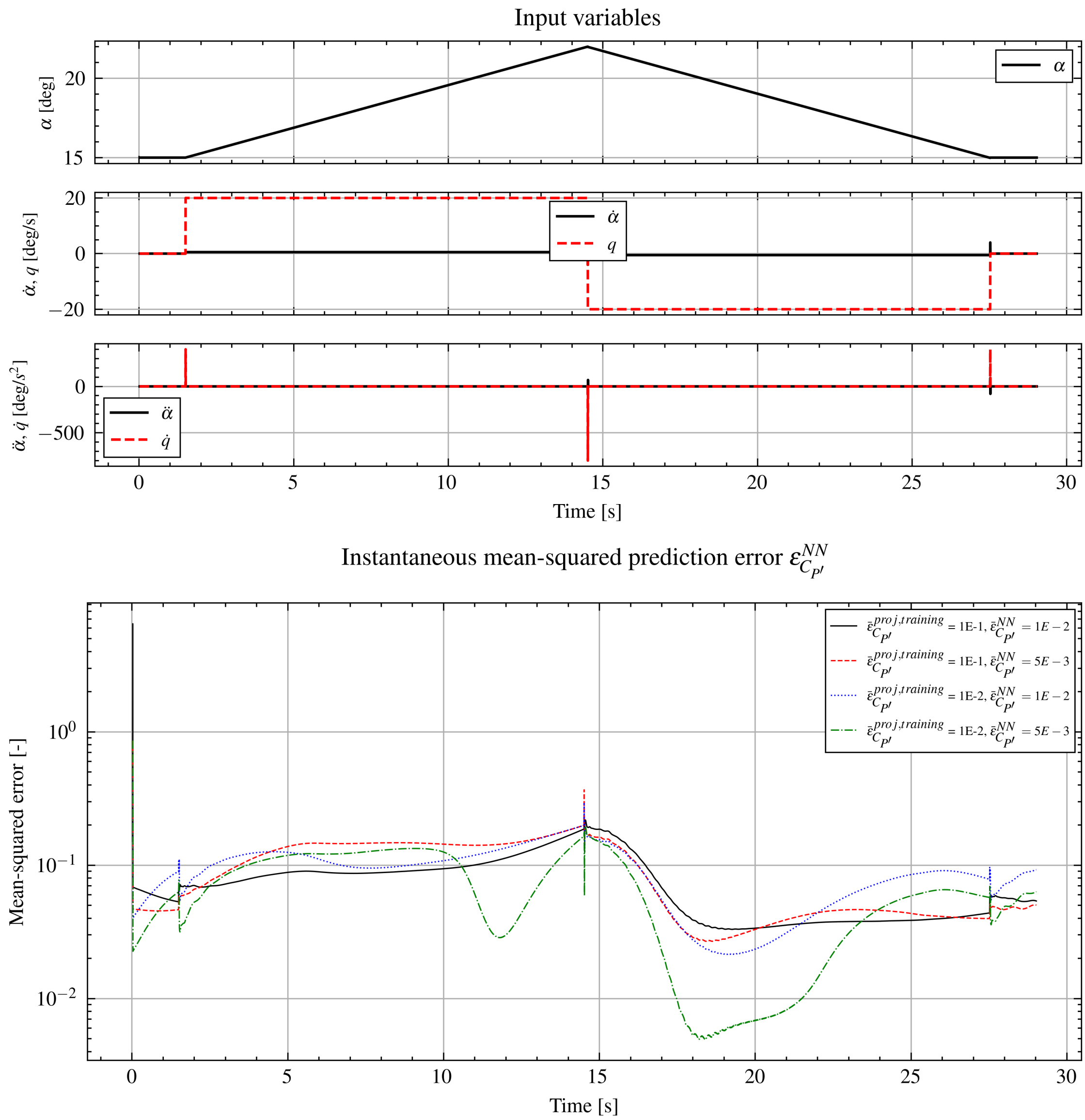


Figure A.1: POD-LSTM input state variables for sharp pitch-up maneuver.

The prediction of the POD-LSTM model for the normal force coefficient, axial force coefficient and the pitching moment coefficient is shown in Figure A.2, Figure A.3 and Figure A.4, respectively. It can be seen that both the normal force coefficient and the pitching moment coefficient are covered well by the POD-LSTM model with errors that are in the same orders of magnitude, whereas the axial force coefficient has an offset in its prediction. The transient that occurs after a change in pitch rate is not covered by the POD-LSTM model, although the average of the transient is predicted well. Furthermore, it can be noted that the peaks of the integral load coefficients that occur in the predictions of the model are due to the instantaneous change of the pitch rate, where its values are outside of the regressor space and therefore cause the error of prediction to increase significantly. The differences between the model prediction for both projection errors, i.e., $\bar{\boldsymbol{\epsilon}}^{\boldsymbol{proj,training}}$ = 1E-1 and $\bar{\boldsymbol{\epsilon}}^{\boldsymbol{proj,training}}$ = 1E-2, can vary up to a factor 100.

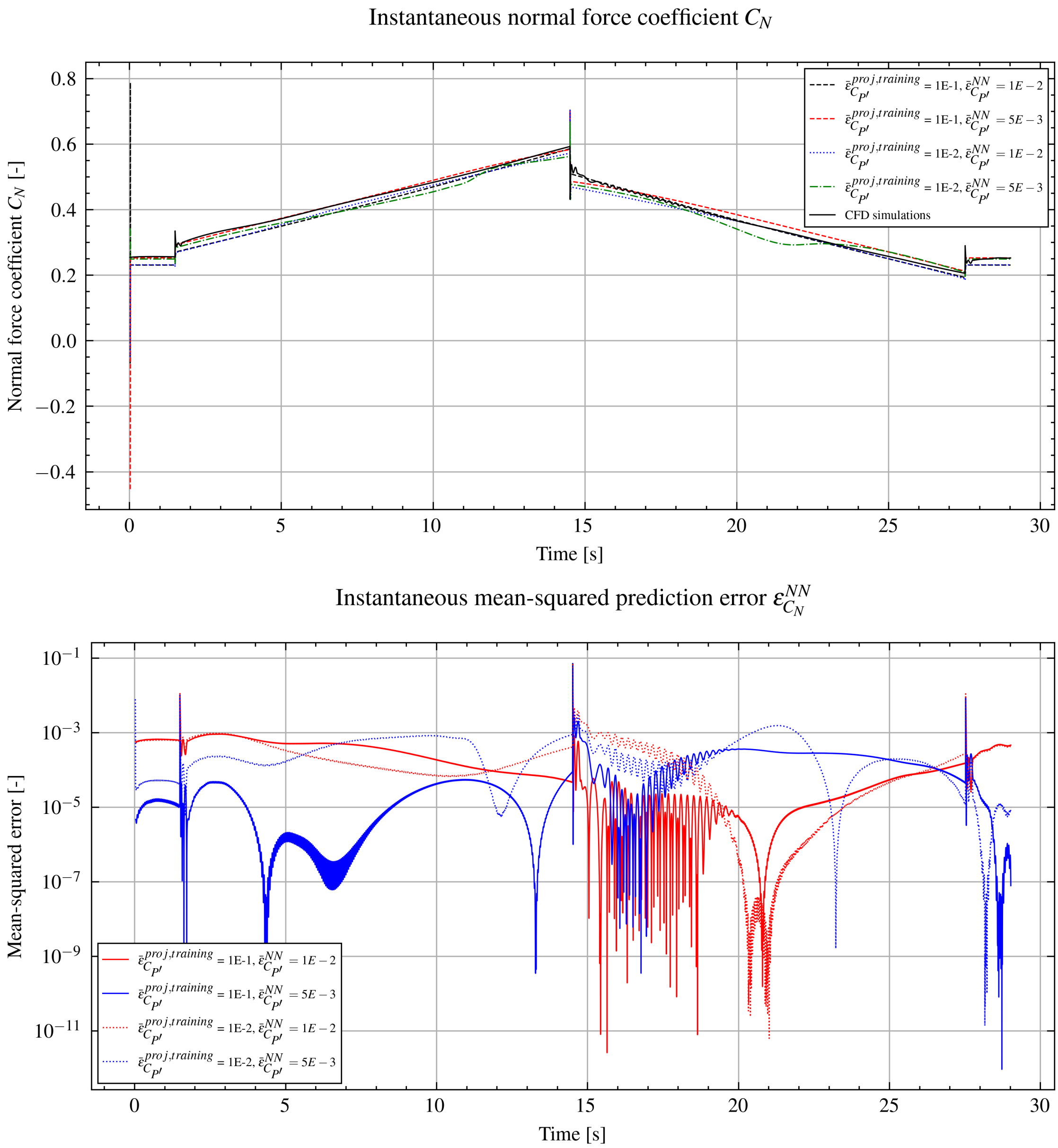


Figure A.2: POD-LSTM prediction of the normal force coefficient for sharp pitch-up maneuver.

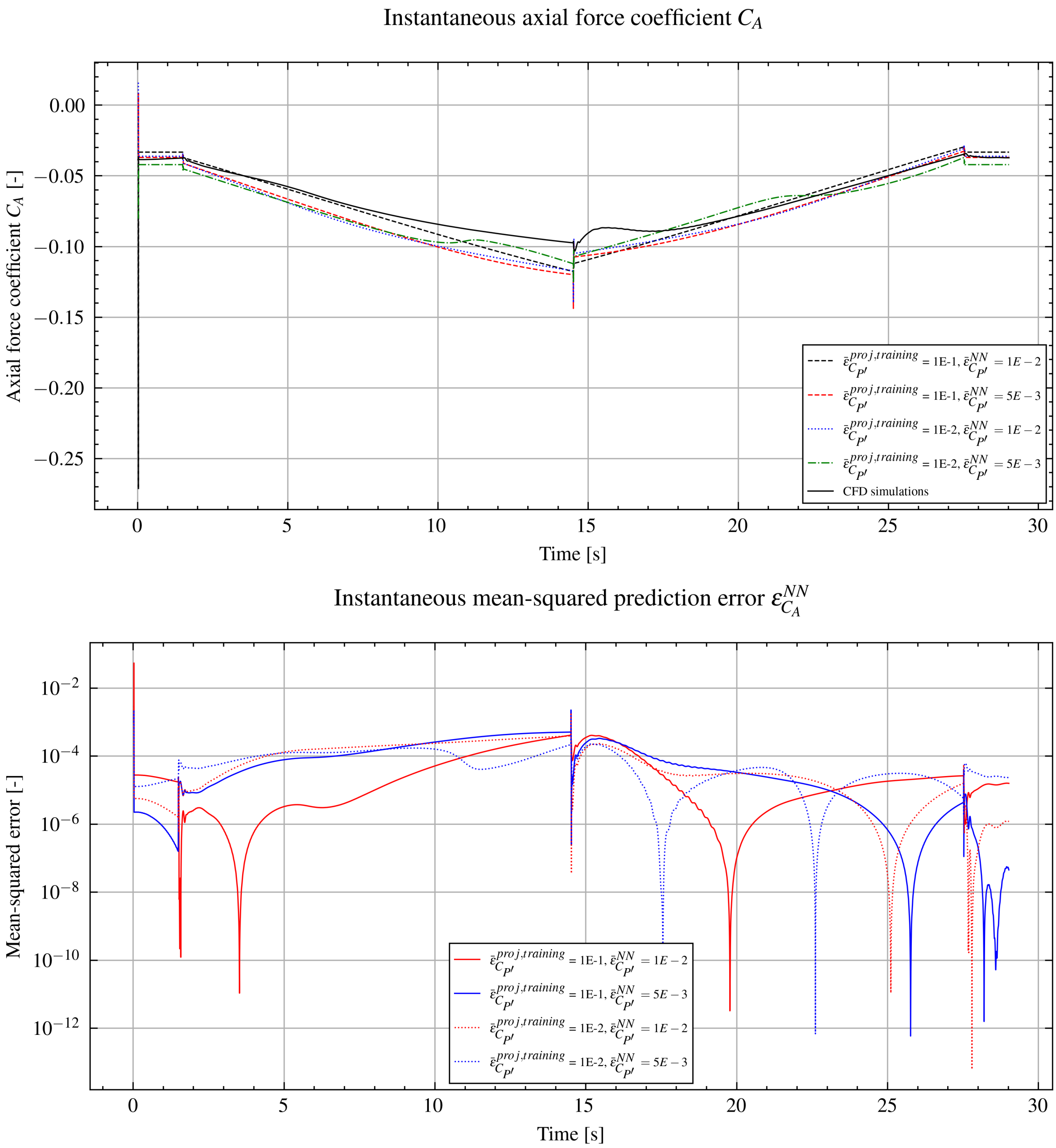


Figure A.3: POD-LSTM prediction of the axial force coefficient for sharp pitch-up maneuver.

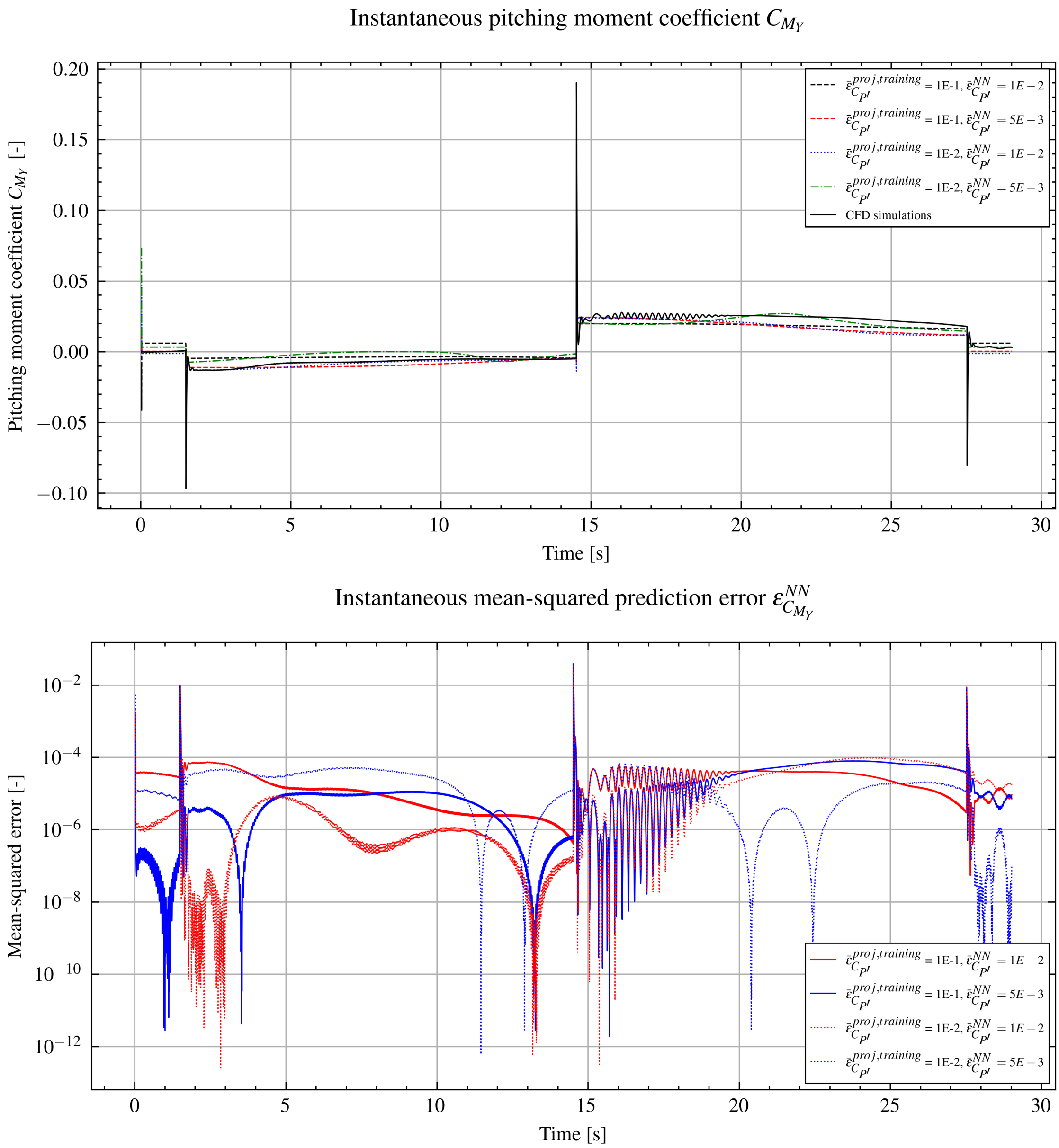
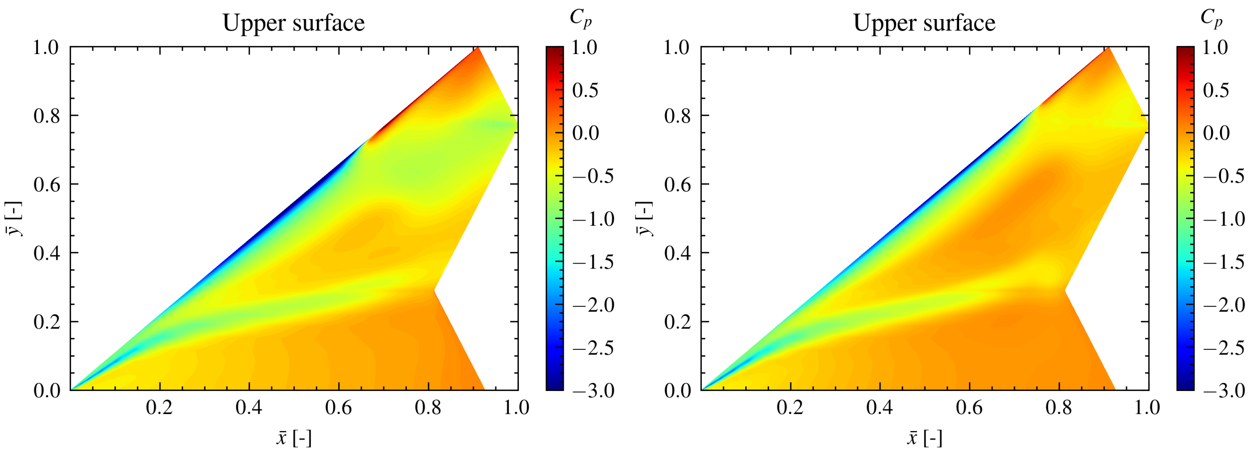


Figure A.4: POD-LSTM prediction of the pitching moment coefficient for sharp pitch-up maneuver.

The instantaneous surface pressure coefficient distribution for two notable time steps is investigated. In Figure A.5 the instantaneous surface pressure coefficients of the upper surface are shown for the CFD simulation at t = 14.5 s, which represents the abrupt changes in the pitch rate of the maneuver, and t = 18 s, where the differences between the different levels of model accuracy are large. As seen from the figures, both a leading edge wing vortex and flow separation near the wing tip region occur. It must be noted that the flow separation of the wing tip pressure coefficient covers a larger area of the upper surface than for the previously discussed pitch and plunge motions. Furthermore, it can be seen that the leading edge wing vortex is extended towards a larger part of the span of the wing.

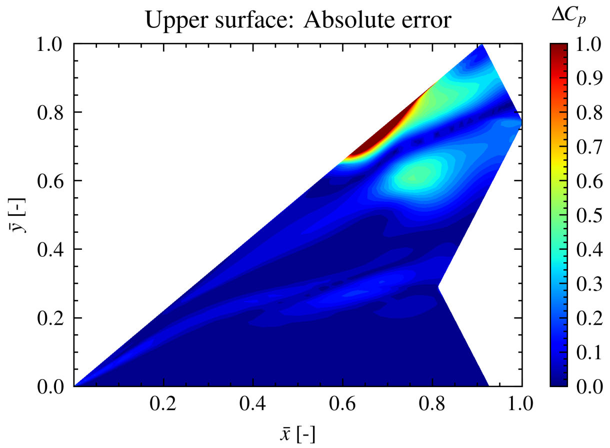


(a) Upper surface pressure distribution at t = 14.5 s.

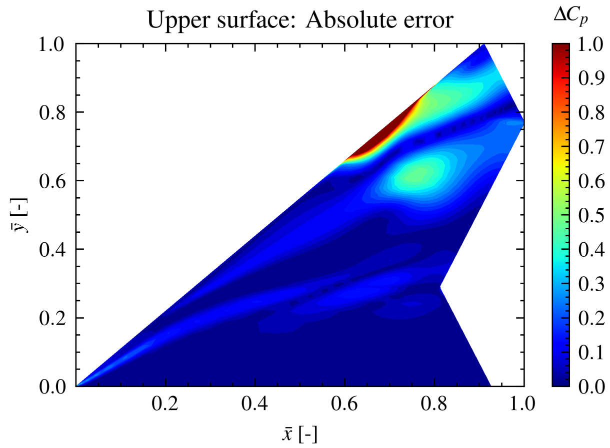
(b) Upper surface pressure distribution at t = 18 s.

Figure A.5: Upper surface pressure distribution for the sharp pitch-up maneuver, at t = 14.5 s and t = 18 s obtained from CFD simulations.

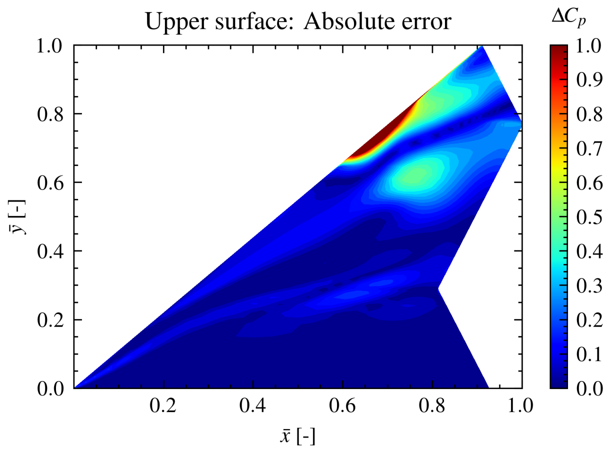
In Figure A.6 and Figure A.7 the absolute error of the surface pressure coefficient prediction is shown for the different levels of model accuracy for t = 14.5 s and t = 18 s, respectively. Similar results as for the pitch and plunge motions are obtained from the figures, where it can be seen that the largest error is found near the tip region of the upper surface of the wing. In Figure A.7 it can be seen that error of the prediction of the flow separation near the wing tip region decreases with increasing model accuracy used for training.



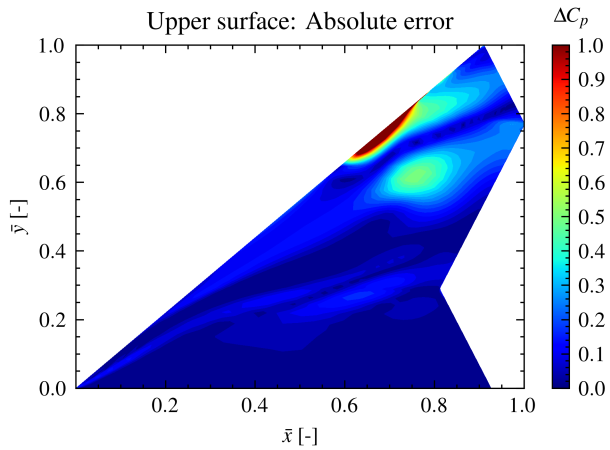
(a) Absolute prediction error at t = 14.5 s for surrogate model $\bar{\epsilon}_{C_{p'}}^{proj,training}$ = 1E-1, $\bar{\epsilon}_{C_{p'}}^{NN,training}$ = 1E-2.



(b) Absolute prediction error at t = 14.5 s for surrogate model $\bar{\epsilon}_{C_{p'}}^{proj,training}$ = 1E-1, $\bar{\epsilon}_{C_{p'}}^{NN,training}$ = 5E-3.

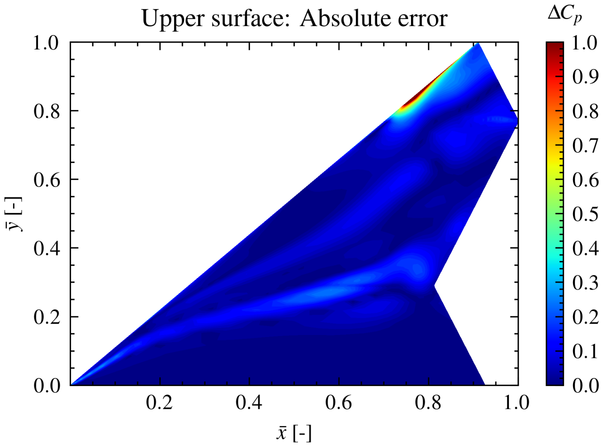


(c) Absolute prediction error at t = 14.5 s for surrogate model $\bar{\epsilon}_{C_{p'}}^{proj,training}$ = 1E-2, $\bar{\epsilon}_{C_{p'}}^{NN,training}$ = 1E-2.

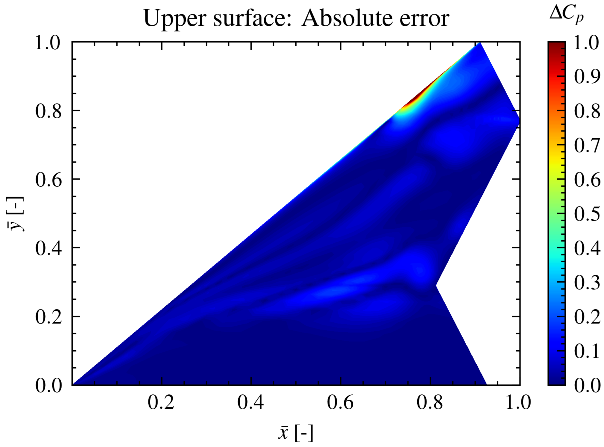


(d) Absolute prediction error at t = 14.5 s for surrogate model $\bar{\epsilon}_{C_{p'}}^{proj,training}$ = 1E-2, $\bar{\epsilon}_{C_{p'}}^{NN,training}$ = 5E-3.

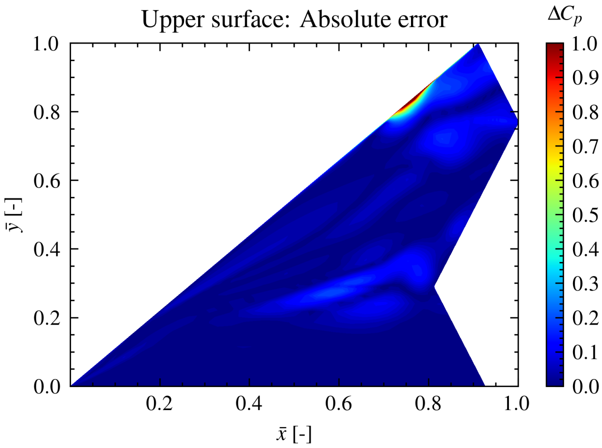
Figure A.6: POD-LSTM absolute prediction error of the upper surface pressure distribution for different surrogate model settings for the sharp pitch-up maneuver at t = 14.5 s.



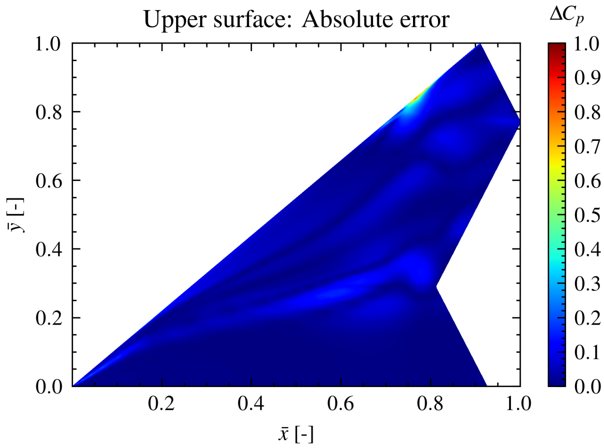
(a) Absolute prediction error at $t = 18$ s for surrogate model $\bar{\epsilon}_{C_{p'}}^{proj,training} = 1\text{E-}1$, $\bar{\epsilon}_{C_{p'}}^{NN,training} = 1\text{E-}2$.



(b) Absolute prediction error at $t = 18$ s for surrogate model $\bar{\epsilon}_{C_{p'}}^{proj,training} = 1\text{E-}1$, $\bar{\epsilon}_{C_{p'}}^{NN,training} = 5\text{E-}3$.



(c) Absolute prediction error at $t = 18$ s for surrogate model $\bar{\epsilon}_{C_{p'}}^{proj,training} = 1\text{E-}2$, $\bar{\epsilon}_{C_{p'}}^{NN,training} = 1\text{E-}2$.



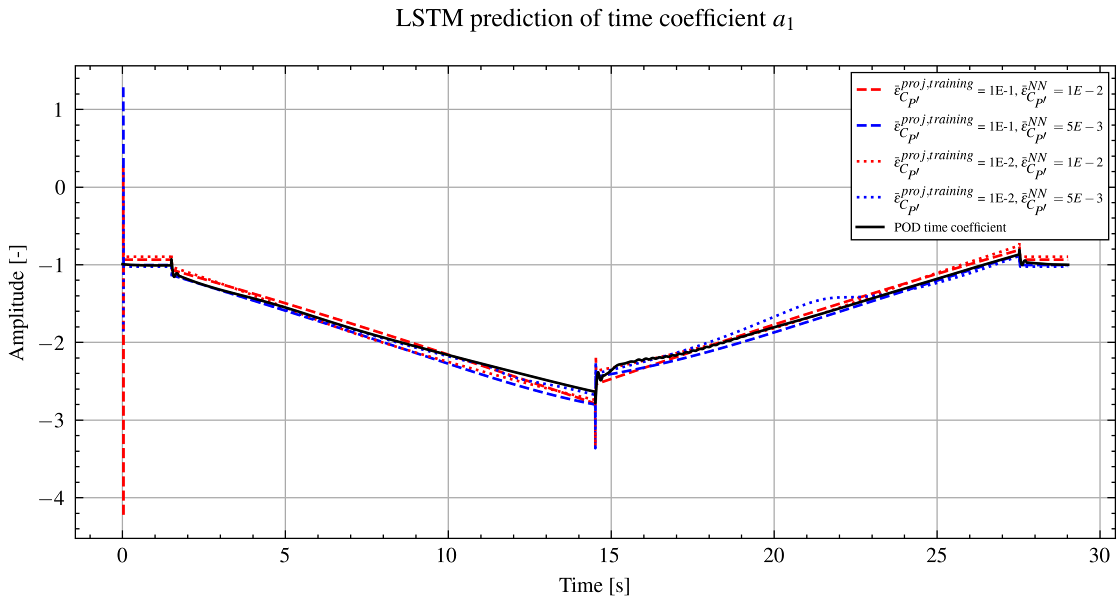
(d) Absolute prediction error at $t = 18$ s for surrogate model $\bar{\epsilon}_{C_{p'}}^{proj,training} = 1\text{E-}2$, $\bar{\epsilon}_{C_{p'}}^{NN,training} = 5\text{E-}3$.

Figure A.7: POD-LSTM absolute prediction error of the upper surface pressure distribution for different surrogate model settings for the sharp pitch-up maneuver at $t = 18$ s.

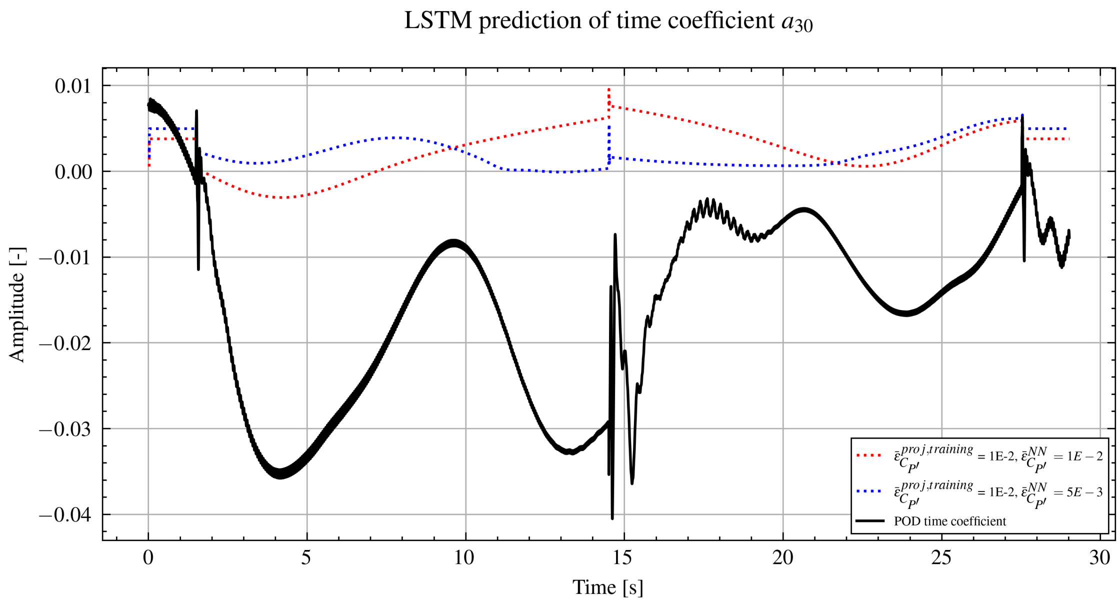
In Table A.1 the prediction error of the time coefficients for the spatial POD modes is shown in terms of NMSE. It can be seen that, similar to previous discussed results, the time coefficients for the first spatial POD mode is predicted with the highest accuracy and the accuracy of the prediction for higher ranked spatial POD modes decreases significantly. Furthermore, it must be noted that the time coefficient prediction for all model settings on average shows better results than for the pitch and plunge motions, especially for the models with a lower projection error, i.e., $\boldsymbol{\bar{\epsilon}_{C_{p'}}^{proj,training}} = 1\text{E-}2$. In Figure A.8 the time coefficient prediction for the first and the thirtieth spatial POD mode is shown, where it can be seen that the amplitudes and frequencies of the signal for the thirtieth spatial POD mode are significantly off.

Table A.1: NMSE evaluation of the POD-LSTM ROM time coefficient prediction for the sharp pitch-up maneuver.

Mode [-]	**NMSE per time coefficient** [-]			
	$\bar{\epsilon}^{proj}_{C_{p'}}$ = 1E-1		$\bar{\epsilon}^{proj}_{C_{p'}}$ = 1E-2	
	$\bar{\epsilon}^{NN}_{C_{p'}}$ = 1E-2	$\bar{\epsilon}^{NN}_{C_{p'}}$ = 5E-3	$\bar{\epsilon}^{NN}_{C_{p'}}$ = 1E-2	$\bar{\epsilon}^{NN}_{C_{p'}}$ = 5E-3
1	0.975982	0.970835	0.978669	0.983779
2	0.88617	0.971469	0.936745	0.939503
3	0.5838	0.545515	0.513483	0.722386
4	0.421363	0.491892	0.39815	0.355895
5	-0.02688	-0.34459	-0.76799	-0.15573
6			-1.32789	-1.59634
7			-0.31109	-0.00837
8			-0.14389	-0.41464
9			-0.4771	-0.32743
10			-1.39854	-1.73915
11			-0.10648	-0.23223
12			-0.76915	-0.79732
13			0.056474	0.056778
14			0.195286	0.150318
15			-0.1427	-0.20642
16			-0.19127	-0.14399
17			-0.46469	-0.61035
18			-0.1566	0.248758
19			-0.14707	-0.25291
20			0.084457	0.08769
21			-0.02132	0.052453
22			-0.09551	-0.21644
23			-0.21305	-0.33801
24			-0.28409	-0.18003
25			-0.63021	-0.64983
26			-0.02589	-0.03738
27			-0.37748	-0.20954
28			0.137783	-0.05389
29			-1.30334	-1.09934
30			-2.57686	-2.46191
31			-0.24817	-0.35453
32			-0.5724	-0.49688
33			-0.71864	-0.38454
34			-1.22691	-1.13637
35			-0.17347	-0.00947
36			-0.19504	-0.109
37			-1.42399	-1.088
38			-0.55018	-0.48896
39			-0.17979	-0.10453
40			-0.28565	-0.1568
Average [-]	0.57	0.53	-0.36	-0.31



(a) POD-LSTM prediction of time coefficient a_1 for the sharp pitch-up maneuver.



(b) POD-LSTM prediction of time coefficient a_30 for the sharp pitch-up maneuver.

Figure A.8: POD-LSTM prediction of time coefficient for different spatial modes for the sharp pitch-up maneuver. **Top:** POD-LSTM prediction of time coefficient with high accuracy. **Bottom:** POD-LSTM prediction of time coefficients with low accuracy.

Bibliography

[1] CS231n Convolutional Neural Networks for Visual Recognition. URL https://cs231n.github.io/neural-networks-1/.

[2] Frederico A. C. Azevedo, Ludmila R. B. Carvalho, Lea T. Grinberg, José Marcelo Farfel, Renata E. L. Ferretti, Renata E. P. Leite, Wilson Jacob Filho, Roberto Lent, and Suzana Herculano-Houzel. Equal numbers of neuronal and nonneuronal cells make the human brain an isometrically scaled-up primate brain. *Journal of Comparative Neurology*, 513(5):532–541, 2009. ISSN 1096-9861. doi: 10.1002/cne.21974. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/cne.21974. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/cne.21974.

[3] Y. Bengio, P. Simard, and P. Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, 5(2):157–166, March 1994. ISSN 1045-9227, 1941-0093. doi: 10.1109/72.279181. URL https://ieeexplore.ieee.org/document/279181/.

[4] Russell M. Cummings, Carsten Liersch, and Andreas Schuette. Multi-Disciplinary Design and Performance Assessment of Effective, Agile NATO Air Vehicles. In *2018 Applied Aerodynamics Conference*, Atlanta, Georgia, June 2018. American Institute of Aeronautics and Astronautics. ISBN 978-1-62410-559-3. doi: 10.2514/6.2018-2838. URL https://arc.aiaa.org/doi/10.2514/6.2018-2838.

[5] Imad Dabbura. Gradient Descent Algorithm and Its Variants, September 2019. URL https://towardsdatascience.com/gradient-descent-algorithm-and-its-variants-10f652806a3.

[6] A. R. Davari and M. R. Soltani. Effects of Plunging Motion on Unsteady Aerodynamic Behavior of an Aircraft Model in Compressible Flow*.

[7] Sepp Hochreiter and Jürgen Schmidhuber. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780, November 1997. ISSN 0899-7667, 1530-888X. doi: 10.1162/neco.1997.9.8.1735. URL https://www.mitpressjournals.org/doi/abs/10.1162/neco.1997.9.8.1735.

[8] Adam Jirasek, Tiger Jeans, Matthew Martenson, Russell Cummings, and Keith Bergeron. Improved Methodologies for Maneuver Design of Aircraft Stability and Control Simulations. In *48th AIAA Aerospace Sciences Meeting Including the New Horizons Forum and Aerospace Exposition*, Orlando, Florida, January 2010. American Institute of Aeronautics and Astronautics. ISBN 978-1-60086-959-4. doi: 10.2514/6.2010-515. URL http://arc.aiaa.org/doi/10.2514/6.2010-515.

[9] Andrej Karpathy. The Unreasonable Effectiveness of Recurrent Neural Networks. URL http://karpathy.github.io/2015/05/21/rnn-effectiveness/.

[10] Andy J. Keane and Prasanth B. Nair. *Computational Approaches for Aerospace Design: The Pursuit of Excellence*. Wiley, 1 edition, June 2005. ISBN 978-0-470-85540-9 978-0-470-85548-5. doi: 10.1002/0470855487. URL https://onlinelibrary.wiley.com/doi/book/10.1002/0470855487.

[11] Slawomir Koziel and Anna Pietrenko-Dabrowska. Basics of Data-Driven Surrogate Modeling. In *Performance-Driven Surrogate Modeling of High-Frequency Structures*, pages 23–58. Springer International Publishing, Cham, 2020. ISBN 978-3-030-38925-3 978-3-030-38926-0. doi: 10.1007/978-3-030-38926-0_2. URL http://link.springer.com/10.1007/978-3-030-38926-0_2.

[12] Carsten M. Liersch and Gavin Bishop. Conceptual Design of a 53deg Swept Flying Wing UCAV Configuration. In *2018 Applied Aerodynamics Conference*, Atlanta, Georgia, June 2018. American Institute of Aeronautics and Astronautics. ISBN 978-1-62410-559-3. doi: 10.2514/6.2018-2839. URL https://arc.aiaa.org/doi/10.2514/6.2018-2839.

[13] J. L. LUMLEY. The structure of inhomogeneous turbulent flows. *Atmospheric Turbulence and Radio Wave Propagation*, 1967. URL https://ci.nii.ac.jp/naid/10012381873. Publisher: Nauka.

[14] Eugene A. Morelli. Multiple input design for real-time parameter estimation in the frequency domain. *IFAC Proceedings Volumes*, 36(16):639–644, September 2003. ISSN 14746670. doi: 10.1016/S1474-6670(17)34833-4. URL https://linkinghub.elsevier.com/retrieve/pii/S1474667017348334.

[15] Anuja Nagpal. L1 and L2 Regularization Methods, October 2017. URL https://towardsdatascience.com/l1-and-l2-regularization-methods-ce25e7fc831c.

[16] Raj Nangia, Mehdi Ghoreyshi, Michel P.C. van Rooij, and Russell M. Cummings. Aerodynamic design assessment and comparisons of the MULDICON UCAV concept. *Aerospace Science and Technology*, 93:105321, October 2019. ISSN 12709638. doi: 10.1016/j.ast.2019.105321. URL https://linkinghub.elsevier.com/retrieve/pii/S1270963819303487.

[17] Michael A. Nielsen. Neural Networks and Deep Learning. 2015. URL http://neuralnetworksanddeeplearning.com. Publisher: Determination Press.

[18] David Papp. Prediction of unsteady nonlinear aerodynamic loads using deep convolutional neural networks: Investigating the dynamic response of agile combat aircraft. 2018. URL https://repository.tudelft.nl/islandora/object/uuid%3Aeda27197-0f43-4b83-a2ac-d0bb564a7ef4.

[19] Michael Phi. Illustrated Guide to LSTM's and GRU's: A step by step explanation, June 2020. URL https://towardsdatascience.com/illustrated-guide-to-lstms-and-gru-s-a-step-by-step-explanation-44e9eb85bf21.

[20] Walter Pitts and Warren S. McCulloch. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4):115–133, December 1943. ISSN 1522-9602. doi: 10.1007/BF02478259. URL https://doi.org/10.1007/BF02478259.

[21] Sebastian Raschka and Vahid Mirjalili. *Python Machine Learning: Machine Learning and Deep Learning with Python, scikit-learn, and TensorFlow, 2nd Edition*. Packt Publishing, 2nd edition, 2017. ISBN 978-1-78712-593-3.

[22] M. Ripepi, M. J. Verveld, N. W. Karcher, T. Franz, M. Abu-Zurayk, S. Görtz, and T. M. Kier. Reduced-order models for aerodynamic applications, loads and MDO. *CEAS Aeronautical Journal*, 9(1):171–193, March 2018. ISSN 1869-5582, 1869-5590. doi: 10.1007/s13272-018-0283-6. URL http://link.springer.com/10.1007/s13272-018-0283-6.

[23] F. Rosenblatt. *The Perceptron, a Perceiving and Recognizing Automaton Project Para*. Cornell Aeronautical Laboratory, 1957. Google-Books-ID: P_XGPgAACAAJ.

[24] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, October 1986. ISSN 1476-4687. doi: 10.1038/323533a0. URL https://www.nature.com/articles/323533a0. Number: 6088 Publisher: Nature Publishing Group.

[25] M. Schroeder. Synthesis of low-peak-factor signals and binary sequences with low autocorrelation (Corresp.). *IEEE Transactions on Information Theory*, 16(1):85–89, January 1970. ISSN 0018-9448. doi: 10.1109/TIT.1970.1054411. URL http://ieeexplore.ieee.org/document/1054411/.

[26] Kunihiko Taira, Steven L. Brunton, Scott T. M. Dawson, Clarence W. Rowley, Tim Colonius, Beverley J. McKeon, Oliver T. Schmidt, Stanislav Gordeyev, Vassilios Theofilis, and Lawrence S. Ukeiley. Modal Analysis of Fluid Flows: An Overview. *AIAA Journal*, 55(12):4013–4041, December 2017. ISSN 0001-1452, 1533-385X. doi: 10.2514/1.J056060. URL https://arc.aiaa.org/doi/10.2514/1.J056060.

[27] Andrea Toselli and Olof B. Widlund. *Domain Decomposition Methods — Algorithms and Theory*, volume 34 of *Springer Series in Computational Mathematics*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2005. ISBN 978-3-540-20696-5 978-3-540-26662-4. doi: 10.1007/b137868. URL http://link.springer.com/10.1007/b137868.

[28] Michel P.C. van Rooij, Neal T. Frink, Brett R. Hiller, Mehdi Ghoreyshi, and Mark Voskuijl. Generation of a reduced-order model of an Unmanned Combat Air Vehicle using indicial response functions. *Aerospace Science and Technology*, 95:105510, December 2019. ISSN 12709638. doi: 10.1016/j.ast.2019.105510. URL `https://linkinghub.elsevier.com/retrieve/pii/S1270963819303104`.

[29] Julien Weiss. A Tutorial on the Proper Orthogonal Decomposition. 2019. doi: 10.14279/depositonce-8512. URL `https://depositonce.tu-berlin.de/handle/11303/9456`. Accepted: 2019-06-20T11:33:10Z.

[30] P.J. Werbos. Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, 78 (10):1550–1560, October 1990. ISSN 00189219. doi: 10.1109/5.58337. URL `http://ieeexplore.ieee.org/document/58337/`.

[31] Jian Yu, Chao Yan, and Mengwu Guo. Non-intrusive reduced-order modeling for fluid problems: A brief review. *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering*, 233(16):5896–5912, December 2019. ISSN 0954-4100, 2041-3025. doi: 10.1177/0954410019890721. URL `http://journals.sagepub.com/doi/10.1177/0954410019890721`.