University of Southampton

Faculty of Engineering and Physical Sciences
School of Engineering

Data-Driven Modelling of Nonlinear Aerodynamics in High-Speed Aircraft Using Machine Learning

by

Gabriele Immordino
ORCiD: 0000-0003-2718-0120

*Thesis for the
degree of Doctor of Philosophy*

September 2025

University of Southampton

Abstract

Faculty of Engineering and Physical Sciences
School of Engineering

Doctor of Philosophy

Data-Driven Modelling of Nonlinear Aerodynamics in High-Speed Aircraft Using Machine Learning

by Gabriele Immordino

This thesis explores how advanced data-driven techniques can improve the modelling of nonlinear aerodynamics in high-speed aircraft, particularly in the transonic regime. As modern workflows increasingly rely on high-fidelity simulations, computational costs escalate when analysing large design spaces or unsteady phenomena. In response, this research develops and validates reduced-order modelling frameworks that integrate machine learning algorithms to provide accurate aerodynamic predictions at significantly reduced cost.

Initial studies on two-dimensional unsteady transonic loads demonstrate that flutter boundaries and other nonlinear aeroelastic features can be efficiently modelled. Building on these findings, more complex strategies are introduced, including parametric Volterra series combined with neural network interpolation, which allow rapid reconstruction of unsteady aerodynamic responses over a design space. Uncertainties in both high- and low-fidelity data are incorporated through a multi-fidelity Bayesian framework, providing confidence intervals for aerodynamic predictions. For large-scale three-dimensional test cases, advanced deep learning architectures, such as graph neural networks and spatio-temporal convolutional models, are proposed. These methods excel at handling unstructured meshes and flow features such as shock waves and separated boundary layers.

Applications include canonical transonic airfoils, wings, and wing-fuselage configurations, with emphasis on reducing simulation time while maintaining high-fidelity in predicting aerodynamic forces and flow characteristics. The proposed machine learning-based surrogates can significantly accelerate and enhance nonlinear aerodynamic analysis under both steady and unsteady conditions. By handling complex, unstructured geometries, integrating uncertainty quantification, and facilitating design space exploration, this work establishes the foundation for efficient optimisation and robust analysis in the design of high-speed aircraft and beyond.

Publications

Refereed Journals

- G. Immordino, A. Vaiuso, A. Da Ronch, and M. Righi. Spatio-temporal graph convolutional autoencoder for transonic wing pressure distribution forecasting. *Aerospace Science and Technology*, page 110516, 2025
- A. Vaiuso, G. Immordino, M. Righi, and A. Da Ronch. Multi-fidelity transonic aerodynamic loads estimation using bayesian neural networks with transfer learning. *Aerospace Science and Technology*, 163:110301, 2025
- G. Immordino, A. Da Ronch, and M. Righi. Parametric nonlinear volterra series via machine learning: Transonic aerodynamics. *Journal of Aircraft*, 0(0):1–18, 2025
- G. Immordino, A. Vaiuso, A. Da Ronch, and M. Righi. Predicting transonic flowfields in non–homogeneous unstructured grids using autoencoder graph convolutional networks. *Journal of Computational Physics*, page 113708, 2025
- G. Immordino, A. Da Ronch, and M. Righi. Steady-state transonic flowfield prediction via deep-learning framework. *AIAA Journal*, pages 1–17, 2024

Papers in Conference Proceedings

- D. Massegur Sampietro, G. Immordino, A. Vaiuso, A. Da Ronch, and M. Righi. Graph-convolutional autoencoder frameworks for aerodynamic shape predictions of the agard wing. In *AIAA SCITECH 2025 Forum*, page 0885, 2025
- G. Coppotelli, R. G. Sbarra, L. Onofri, M. Righi, G. Immordino, and A. Da Ronch. Experimental characterization of the flutter behavior of a very flexible wing. In *AIAA SCITECH 2025 Forum*, page 1019, 2025
- M. Righi, G. Coppotelli, G. Immordino, A. Da Ronch, L. Onofri, R. Sbarra, and G. Romano. Experimental characterization of flutter and lco of a very flexible wing. In *AIAA SCITECH 2024 Forum*, page 0831, 2024

- G. Immordino, A. Da Ronch, and M. Righi. Deep–learning framework for aircraft aerodynamics prediction. In *AIAA AVIATION 2023 Forum*, page 3846, 2023
- A. Da Ronch, G. Immordino, and J. W. Kim. A preliminary investigation into icing accretion around a wavy leading-edge wing. In *AIAA SCITECH 2023 Forum*, page 2459, 2023
- M. Righi, S. Düzel, D. Anderegg, A. Da Ronch, D. Massegur Sampietro, I. Soukhmane, and G. Immordino. Rom-based uncertainties quantification of flutter speed prediction of the bscw wing. In *AIAA SCITECH 2022 Forum*, page 0179, 2022

Contents

List of Figures

List of Tables

Declaration of Authorship

I declare that this thesis and the work presented in it is my own and has been generated by me as the result of my own original research.

I confirm that:

1. This work was done wholly or mainly while in candidature for a research degree at this University;
2. Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated;
3. Where I have consulted the published work of others, this is always clearly attributed;
4. Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work;
5. I have acknowledged all main sources of help;
6. Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself;
7. Parts of this work have been published as: Gabriele Immordino

Signed: .... Date: 10/09/2025

Acknowledgements

I wish to express my deepest gratitude to my supervisors, Prof. Andrea Da Ronch at the University of Southampton and Prof. Marcello Righi at the Zurich University of Applied Sciences (ZHAW), for their invaluable support, expert guidance, and insightful feedback throughout my doctoral research. Their encouragement and dedication have been instrumental in shaping both the direction and depth of this thesis.

I also acknowledge the use of the IRIDIS High Performance Computing Facility, and associated support services at the University of Southampton, which provided the computational resources essential for completing this work.

The project has been supported by Digitalization Initiative of the Zurich Higher Education Institutions (DIZH) grant 9710.Z.12.P.0003.05 from ZHAW.

To my family, for their infinite love and encouragement that has fuelled my ambition.

To Margherita, whose unfailing support and understanding have been a constant source of strength.

To my grandparents, for their unconditional affection and life lessons that have shaped who I am.

To my friend and colleague, Andro, for standing by my side in and out of the office.

To all my colleagues, for the friendship and inspiration that motivated me every day.

And to my doctoral fathers, Prof. Andrea Da Ronch and Prof. Marcello Righi, whose mentorship, guidance, and faith in my abilities have been the foundation of this journey.

Definitions and Abbreviations

Acronyms	
AoA	Freestream Angle of Attack
AePW	Aeroelastic Prediction Workshop
ARMA	Auto-Regressive Moving Average
BNN	Bayesian Neural Network
BO	Bayesian Optimisation
BSCW	Benchmark Supercritical Wing
CFD	Computational Fluid Dynamics
CNN	Convolutional Neural Network
CRM	Common Research Model
DMDc	Dynamic Mode Decomposition with control
DNN	Dense Neural Network
DOE	Design of Experiments
FCNN	Fully-Connected Neural Network
GCN	Graph Convolutional Network
GDL	Geometric Deep Learning
GNN	Graph Neural Network
GPR	Gaussian Process Regression
GRU	Gated Recurrent Unit
LHS	Latin Hypercube Sampling
LSTM	Long Short-Term Memory
ML	Machine Learning
MAE	Mean Absolute Error
MAPE	Mean Absolute Percentage Error
MSE	Mean Squared Error
MWLSI	Moving Weighted Least Squares Interpolation
NN	Neural Network
PReLU	Parametric Rectified Linear Unit
POD+I	Proper Orthogonal Decomposition with Interpolation
RANS	Reynolds-Averaged Navier Stokes
RMSE	Root Mean Square Error
ROM	Reduced-Order Model

SVD	Singular Value Decomposition
STGCN	Spatio-Temporal Graph Convolutional Network
TL	Transfer Learning

Roman Symbols

A^+	Pseudo-inverse of the matrix A
a_h	Location of the elastic axis from the mid chord in semichords
b	Semichord
c	Chord
C_D	Drag coefficient
C_F	Skin friction coefficient
C_L	Lift coefficient
C_M	Pitching moment coefficient
C_P	Pressure coefficient
C_P/deg	Pressure coefficient normalized by oscillation amplitude
$(C_P)_{Im}/deg$	Imaginary component of C_P/deg
$(C_P)_{mean}$	Mean pressure coefficient
$(C_P)_{Re}/deg$	Real component of C_P/deg
$h()$	Convolution kernels
$h_0...h_n$	Volterra kernels
H_n	Volterra operator of order n
k	Reduced frequency
$L_r()$	Laguerre polynomials of order r
m	Structural mass
$m_1...m_M$	Memory lag terms
M	Mach number
Re	Reynolds number
S	Reference surface
q_∞	Freestream dynamic pressure, $\frac{1}{2}\rho U^2$
q_{F}	Flutter dynamic pressure
U	Freestream velocity
U^*	Reduced velocity, $U/b\sqrt{K_\alpha/I_\alpha}$
x_α	Offset of the centre of gravity from the elastic axis in semichords

Greek Symbols

α	Pitch angle
$\dot{\alpha}$	Pitch rate
$\ddot{\alpha}$	Pitch acceleration
$\bar{\alpha}$	Mean pitch angle
α_A	Angular amplitude
ρ	Density
τ	Reduced time

$\Delta\tau$	Nondimensional timestep
ξ	Plunge
$\dot{\xi}$	Plunge rate
$\ddot{\xi}$	Plunge acceleration
$\boldsymbol{Y}$	Model output fields
$\boldsymbol{x}$	Grid point coordinates
$\mathcal{S}$	Surface mesh

Chapter 1

Introduction

This work explores how modern machine learning (ML) approaches can accelerate and improve the study of high-speed aerodynamic flows, a key area of modern aircraft design and optimisation. In recent decades, the aerospace industry has increasingly relied on high-fidelity computational fluid dynamics (CFD) simulations to capture complex transonic and supersonic phenomena, ranging from shock-boundary layer interactions to viscous flow separation. While such simulations provide valuable physical insights, their high computational cost can make them impractical for large-scale design iterations or real-time decision making. Consequently, there is a strong interest in combining the accuracy of high-fidelity simulations with the computational efficiency of reduced-order models (ROMs), leading to the emergence of data-driven frameworks.

In this context, this thesis aims to integrate advanced ML models with traditional aerodynamic simulation paradigms to enable reliable and time-efficient predictions of complex flow physics. Focusing on typical aerodynamic problems in the transonic regime, it specifically investigates how nonlinear phenomena, such as shock formation, boundary layer separation, and three-dimensional flow interactions, can be accurately modelled using ML techniques. The central hypothesis is that appropriately trained data-driven models can achieve the predictive accuracy of high-fidelity CFD at a fraction of the computational cost. By exploring this hypothesis, the thesis aims to broaden the scope of ML in aerospace research and thus provide a path to faster and more cost-effective analysis in aircraft design and optimisation.

1.1 Challenges in High-Fidelity Aerodynamic Simulations

Certain aerodynamic problems, including turbulent flows, transonic shock-wave interactions and fluid-structure coupling, require very fine spatial and temporal resolution to resolve multi-scale features [195, 200]. Conventional high-fidelity approaches, such as Reynolds-Averaged Navier-Stokes (RANS) and Large-Eddy Simulation (LES), become prohibitively expensive when applied to unsteady problems or to the repeated analyses required for multiple flight conditions [69], structural variants [24] or control scenarios [32, 70]. Flutter prediction is a typical example: capturing the unsteady loading over several structural modes and across a range of dynamic pressures can take several days per run with a fully time-accurate CFD solver [75], making such calculations infeasible within an iterative design loop.

To obtain rapid load estimates during the concept phase, designers therefore rely on inexpensive potential flow models [97, 237]. Although relatively inexpensive, these methods neglect the nonlinearities that dominate the transonic and supersonic regimes [26], so higher-fidelity CFD is reintroduced at later stages to ensure reliable performance and stability predictions [151]. However, the computational burden of full-order solvers clashes with industrial needs for real-time flow control [62], large-scale design optimisation [4], and rigorous uncertainty quantification [341]. Bridging this gap requires models that balance accuracy with speed-hence the growing interest in reduced-order and data-driven techniques, which the remainder of this thesis explores in the context of high-speed aerodynamics.

1.2 The Role of Reduced Order Models in Aerodynamics

ROMs have emerged as a powerful alternative to traditional high-fidelity CFD simulations, offering a balance between computational efficiency and predictive accuracy [289]. By capturing the essential flow physics in a lower-dimensional representation, these models are particularly beneficial in transonic [115] and transient aerodynamic applications [376] where full-scale CFD analysis is prohibitively expensive for iterative design processes [4] or real-time control [62].

The fundamental principle of model order reduction is to project a high-dimensional system onto a reduced subspace that retains the most significant dynamic behaviour of the flow [219]. There are usually two stages in this process, one offline and one online. In the offline phase, full-scale CFD simulations are performed at selected design points, generating a dataset of high-fidelity solutions. The most representative flow structures are extracted to form a reduced basis. The online phase then leverages this reduced basis to make rapid aerodynamic predictions with significantly lower computational cost.

Various ROM strategies have been developed to address different aerodynamic challenges. Projection-based techniques, such as Proper Orthogonal Decomposition (POD) and Dynamic Mode Decomposition (DMD), have been widely used for flowfield reconstruction and transient aerodynamics [30, 84, 168, 362, 365, 306, 322, 336, 337]. POD ranks snapshot data by energy to build an optimal linear subspace [30, 153], whereas

DMD extracts modes with single frequency temporal behaviour [306, 351]. Their linear nature makes them ideal for weakly nonlinear dynamics, yet it also explains their limitations when strong nonlinearities, such as shock waves, moving separation bubble or limit cycle oscillation, dominate the solution [220, 336, 337]. Sharp gradients or discontinuities cannot be efficiently represented by a small number of globally supported smooth basis functions, so the modal expansion must grow rapidly to maintain accuracy [153, 254]. In addition, both POD and DMD are inherently interpolation tools, once the system leaves the convex hull of the training snapshots, for example due to shock induced changes in topology, the reduced model lacks a consistent representation and its prediction quality deteriorates [7, 9, 167, 247, 370].

To address these limitations, ML-based ROMs have gained prominence by leveraging advanced architectures [112, 392]. Unlike traditional methods that explicitly rely on governing equations, these data-driven models learn complex aerodynamic relationships directly from CFD-generated datasets [205, 388]. By mapping input parameters to aerodynamic quantities of interest, these models provide highly flexible and efficient surrogates that generalize well across different flow conditions [133]. ROMs have demonstrated significant potential in various aerodynamic applications, including transonic flowfield prediction, vortex shedding analysis, and aeroelastic modelling [112, 215].

Despite their advantages, ROMs are not without drawbacks. Their accuracy heavily depends on the representativeness of the training dataset, making it extremely important to sample the parameter space effectively [205]. Additionally, their extrapolation capability remains limited: once predictions extend beyond the sampled region, reliability decreases [168]. To address these issues, recent research has focused on hybrid and multi-fidelity modelling techniques [110, 239, 349, 291, 212], where ROMs are fed with additional data sources or embedded within physics-informed frameworks to enhance robustness [112, 114].

1.3 Project motivation

Despite significant advances in both high-fidelity CFD and data-driven modelling approaches, the gap between the accuracy offered by large-scale CFD analyses and the agility required for practical engineering workflows remains substantial [364, 418]. This problem is particularly evident in the transonic regime, where strong nonlinearities, ranging from shock wave formation and vortex interactions to local flow separation, lead to high computational cost and increased numerical complexity [167]. Even small uncertainties in these flow features can cause large deviations in the predicted aerodynamic loads, making conventional ROMs less reliable. Traditional linear projection techniques often require extensive modes and complex interpolation schemes to handle the pronounced nonlinearities characteristic of transonic flows. Consequently, while these methods are well established for low-speed or weakly nonlinear regimes, they may be ill-suited for the high-speed, shock-dominated contexts essential to modern aerodynamic applications [84, 246].

Unlike previous approaches [56, 400, 401] that first distil certain physical attributes and then feed these distilled features into the network, the methods herein proposed delegate most of the feature extraction directly to the neural architecture itself. Current ML surrogates for aerodynamics span three main families: feature-engineered pipelines that rely on hand-crafted descriptors or precomputed physical quantities before learning [56, 401], physics-informed models that enforce constraints through residual penalties or soft priors [279, 337], and operator-learning and graph-based simulators that learn solution maps on grids or meshes [47, 206, 267]. In transonic, shock-dominated regimes, feature-engineered surrogates can degrade when flow topology changes invalidate the chosen descriptors, physics-informed objectives may struggle near discontinuities, and operator learners often require careful retraining to remain robust across operating conditions and geometries [167, 206, 246, 337]. We therefore pursue end-to-end learning on signals and unstructured meshes so that the network selects flow relevant features and representations, with the aim of reducing assumptions about the physics and improving robustness to shocks, moving separation, and geometry changes [207, 371, 385, 394].

Consequently, the core research objective of this thesis is to enable near real-time aerodynamic predictions in the transonic regime without sacrificing accuracy. Achieving this requires more than mere improvements to existing ROM frameworks. Instead, it requires a paradigm shift toward methods capable of capturing localized physical processes, abrupt flow features, and complex shock-boundary layer interactions. ML models, especially deep neural networks and novel graph-based architectures, offer a path for capturing complex nonlinear flow behaviours at reduced computational cost. By embedding nonlinear activation functions, these methods can incorporate a richer, more flexible representation of flow physics, mapping high-dimensional data into meaningful latent spaces in a way that conventional linear or polynomial-based ROMs cannot [371]. Indeed, recent studies have demonstrated the effectiveness of neural network surrogates in reproducing turbulence statistics [394], accurately predicting shock locations even in the presence of strong discontinuities [207], and modelling transient aerodynamic responses with high accuracy [385] faster than full-scale CFD, which motivates their use for iterative design, optimisation, and control.

Bridging the fidelity-speed gap also has broader implications. In the context of multi-fidelity design and optimisation loops, efficient surrogates are essential for coupling aerodynamic, structural, and control analyses. A transonic aircraft wing, for example, may require hundreds of evaluations under varying angles of attack, Mach numbers, and structural deformations. Relying solely on direct CFD for all these cases may not be feasible, especially under time or budget constraints. By integrating robust ML strategies into the model reduction workflow, designers can significantly reduce computational effort while retaining the essential transonic flow physics.

In addition, the ability to quantify uncertainty and provide reliable confidence bounds supports a more informed and systematic design process [12]. Unlike deterministic methods that yield single-point estimates, probabilistic approaches produce predictive distributions that capture both model inaccuracies and operational uncertainties. This enables the evaluation of worst-case scenarios and sensitivities in aerodynamic responses, thereby reducing the risk of unforeseen instabilities or performance shortfalls under off-design conditions. Moreover, explicit uncertainty estimates guide the allocation of computational resources to the most critical regions of the design space, where high-fidelity analyses are most needed, resulting in notable savings without sacrificing reliability [382, 410]. Consequently, design margins can be calibrated with greater precision, ensuring that safety and performance criteria are met while avoiding excessive conservatism [296].

Overall, this thesis is motivated by the pressing need for models that combine the reliability of high-fidelity CFD with the efficiency demanded by industrial and real-time applications. The research therefore aims to (i) identify the key challenges posed by strong transonic nonlinearities, (ii) propose and refine ML-driven ROM frameworks suitable for these conditions, and (iii) rigorously test their performance on a range of aerodynamic problems. By advancing the fidelity of data-driven surrogates, this project aims to open up new opportunities for rapid aerodynamic analysis, uncertainty quantification, and integrated aero-structure optimisation, ultimately contributing to faster, more robust design cycles in the aerospace industry.

1.4 Research objectives

The overarching goal of this doctoral research is to propose, implement, and validate advanced ML-based reduced-order modelling strategies for capturing nonlinear aerodynamic phenomena in high-speed aircraft applications. Particular emphasis is placed on two complementary capabilities: time–resolved prediction of integrated aerodynamic loads and accurate modelling of surface vector fields. Leveraging high-fidelity CFD data, the proposed models aim to reconstruct unsteady flow physics with reduced computational cost, provide precise estimates of aerodynamic loads, and quantify confidence intervals. These capabilities are intended to accelerate the iterative design and analysis of aircraft.

The thesis is therefore structured around five specific objectives, each mapped to one of the two capabilities listed above and contributing to a broader data–driven modelling framework for nonlinear transonic aerodynamics.

O1. Dynamics Prediction: Model the temporal evolution of global aerodynamic loads. Accurate prediction of unsteady aerodynamic loads is essential for aeroelastic analysis and control design, yet full-order CFD simulations are computationally prohibitive for time-domain applications. Classical ROMs accuracy tend to deteriorate in strongly nonlinear regimes, such as those involving shock-induced separation or large-amplitude motion, leading to rapid error accumulation over time, reduced robustness during extrapolation, and numerical instability when coupled with structural solvers. To address these limitations, two complementary methodologies are proposed: a linear-quadratic autoregressive ML-based ROM for a two-dimensional (2D) pitching airfoil, selected for tractability and integration into an aeroelastic solver for flutter boundary detection, and a nonlinear Volterra-series-based ROM for three-dimensional (3D) configurations, in which deep neural networks learn to generalise Volterra kernels across varying flight conditions. These approaches mitigate long-term error propagation, preserve stability under fluid-structure coupling, and maintain predictive accuracy when extrapolating beyond the training region. The resulting frameworks deliver accurate and efficient unsteady force predictions at a fraction of the cost of full-order CFD, enabling their integration into aeroelastic analyses performed during the design process.

O2. Uncertainty Quantification: Incorporate predictive confidence in integrated aerodynamic forces predictions.
In safety-critical aerospace applications, surrogate models must provide not only accurate point predictions but also credible uncertainty estimates to ensure reliability under the many approximations introduced by reduced-order modelling, sparse training data, numerical discretisation variability, and diverse fidelity data sources. Addressing this need is particularly challenging due to the difficulty in specifying appropriate noise models, as well as the risk of overconfident extrapolation in under-sampled regions of the design space. This thesis adopts different strategies to tackle these issues. Classical uncertainty propagation techniques, such as Monte Carlo sampling and polynomial chaos expansions, are employed to benchmark the robustness of deterministic ROMs under parametric variability. Subsequently, a multi-fidelity Bayesian neural network framework is introduced, leveraging the complementary strengths of low- and high-fidelity simulation data to produce probabilistic predictions of integrated aerodynamic loads. The resulting models deliver reliable confidence intervals alongside predictions, thereby enhancing trust in surrogate-based analyses and enabling risk-aware decision-making in design and certification.

O3. Spatial Modelling: Predict steady aerodynamic vector fields on complex unstructured grids.
High-fidelity modelling of 3D unsteady aerodynamics requires capturing rich

spatial structures on unstructured meshes, which are not compatible with classical convolutional neural network architectures. The transition from 2D to 3D geometries introduces both geometric complexity and increased sensitivity to mesh resolution and topology. A deep learning framework is first established that performs well on structured domains but degrades on irregular, unstructured grids. To overcome its limitations, a graph-based surrogate modelling framework is introduced, combining a graph convolutional network with an autoencoder to extract and reconstruct low-dimensional representations of steady aerodynamic fields while preserving mesh connectivity and geometric fidelity. The method enables mesh-consistent, accurate, and efficient prediction of vector fields in complex aerodynamic configurations.

O4. Spatio–temporal Modelling: Capture unsteady flow dynamics on complex unstructured grids.
The spatial framework of O3 is extended to unsteady flows by integrating temporal recurrence into a spatio-temporal graph convolutional network. This framework learns to predict time-resolved surface pressure fields by capturing both spatial mesh structure and temporal evolution of the flow. Difficulties lie in capturing long-range dependencies on non-Cartesian grids, balancing model performance against training data scarcity, and controlling error accumulation during multi-step autoregression. The proposed framework allows for accurate, mesh-consistent prediction of transient pressure fields on complex lifting surfaces, supporting applications such as aeroelastic coupling and flow state reconstruction for feedback control.

O5. Modal Space Analysis: Estimate aerodynamic responses for wing geometries deformed by prescribed structural modes.
Early-stage aircraft design demands rapid evaluation of aerodynamic performance over large modal deformation spaces, yet high-fidelity CFD remains far too expensive for such exploratory analysis. A major difficulty lies in generalising aerodynamic responses to shape-induced flow variations while preserving both geometric fidelity and numerical consistency across different configurations. This becomes particularly challenging when mesh topology changes with geometry, risking the loss of structural correspondence, or when the surrogate is required to extrapolate in poorly sampled regions of the modal space. To meet this need, this objective extends the graph-based learning framework of O3 to enable fast aerodynamic prediction across a range of wing configurations deformed by prescribed structural modes. The proposed method provides fast and accurate aerodynamic evaluation suitable for integration into optimisation pipelines and preliminary aircraft design analyses.

1.5 Thesis Outline

This thesis is organised into nine chapters that progressively develop data-driven models for predicting nonlinear aerodynamics in high-speed aircraft and address the research objectives outlined in Section 1.4. Chapter 2 establishes the theoretical foundations for reduced-order modelling, with a particular focus on both classical projection-based techniques and relevant ML architectures, highlighting their respective capabilities and limitations in capturing complex aerodynamic phenomena.

Building on these foundations, Chapter 3 addresses part of Research Objective O1 by presenting data-driven ROMs for predicting unsteady transonic loads on a 2D supercritical airfoil. Here, several ROM architectures are tested to identify the flutter boundary, with particular emphasis on how dataset selection and turbulence modelling can affect predictive reliability and computational costs. These investigations also contribute to Research Objective O2, by assessing the sensitivity of ROM performance to input uncertainties and evaluating their robustness under different modelling assumptions.

Chapter 4 achieves Research Objective O1, extending nonlinear aerodynamic modelling to more complex, 3D flows. Through an adaptation of parametric Volterra series, ML is used to interpolate Volterra kernels over a parameter space spanning Mach number and angle of attack. This framework is validated first on two-dimensional airfoils and then on 3D wing applications to demonstrate its adaptability and robustness.

Chapter 5 focuses on Research Objective O2, introducing a multi-fidelity Bayesian neural network that fuses both high- and low-fidelity simulation data to improve predictions of steady transonic aerodynamic loads while quantifying prediction confidence. By using multi-level data, this method reduces computational burden and improves uncertainty estimates, highlighting the benefits of combining sparse high-fidelity data with faster, lower-accuracy simulations.

In Chapter 6, a deep learning-based surrogate model designed for 3D steady, transonic vector fields is proposed. Benchmarked against a POD-based ROM, the nonlinear network architecture reproduces shock-induced discontinuities and separation zones with markedly higher fidelity across diverse flight conditions. Beyond advancing spatial generalisation, this chapter contributes to Research Objective O2 by analysing the model sensitivity to variations in freestream parameters. However, the model limitations on complex unstructured grids motivate the development of a mesh-consistent alternative, leading into Research Objective O3. Chapter 7 fulfills the remainder of Research Objective O3 by proposing a graph-based surrogate that operates directly on unstructured CFD meshes. An autoencoder combined with a Graph Convolutional Network encodes spatial flowfields into a low-dimensional latent space and reconstructs them, enabling efficient and geometry-consistent predictions across non-Cartesian domains.

Chapter 8 advances Research Objective O4, extending the graph-based approach of Chapter 7 to transient problems using a Spatio-Temporal Graph Convolutional Network that, by combining autoregressive temporal layers with graph convolutions, encapsulates both the spatial mesh connectivity and the temporal evolution of transonic vector fields. The result is a more accurate prediction of time-dependent aerodynamic responses.

Chapter 9 further adapts the framework of Chapter 7 to modal space analysis, in order to address Research Objective O5. By embedding geometric deformations and their associated vector fields in a shared latent space, the graph-convolutional autoencoder model accurately generalises across mode-induced wing shape variations and delivers fast steady aerodynamic evaluations for preliminary design studies.

Finally, Chapter 10 synthesises the main research findings and critically evaluates the contribution of the developed ML-based ROM to the research objectives. It also discusses the limitations inherent in the proposed modelling strategies and outlines promising directions for further research.

Chapter 2

Foundations

In this chapter, we establish the theoretical foundations and computational strategies that we will use throughout our research. We begin by introducing ROMs as computationally efficient surrogates for approximating high-fidelity aerodynamic analysis and discuss both their classical projection-based formulations and modern data-driven variants. We then delve into the fundamentals of ML, explaining how both Gaussian processes and neural networks can handle highly nonlinear aerodynamic phenomena, from shock waves to flow separation. This chapter serves as a precursor to the following ones, in which these methodologies are applied to specific test cases.

2.1 Data-driven Reduced-order Models

In many aerospace and mechanical engineering applications, hierarchies of numerical models are employed to balance accuracy and computational cost. High-fidelity methods, often based on discretizing nonlinear partial differential equations, can reliably predict complex flow physics but are expensive to run, especially for unsteady or parametric studies requiring evaluations at many points in the design space. Lower-fidelity methods alleviate the computational burden but may struggle to capture complex nonlinear phenomena, such as shock waves, boundary-layer separation, or large-amplitude oscillations in transonic flows. ROMs offer a middle ground by retaining, in principle, much of the predictive power of high-fidelity approaches while reducing the number of degrees of freedom [96, 320]. Their fundamental idea is to project a high-dimensional flow state (for instance, from a finely resolved CFD mesh) onto a significantly lower-dimensional subspace that still captures the essential flow physics. Classic projection-based methods, such as POD [39, 60, 154] and DMD [307], have been effective in uncovering dominant flow structures in linear or weakly nonlinear regimes [194, 219]. However, traditional linear dimensionality reduction techniques can struggle with strong nonlinearities unless an impractically large number of modes is used. To better handle nonlinearities, various extensions have been proposed. Parametric ROMs interpolate coefficients or modes at different points in the parameter space [6, 8, 10, 398], often assuming that the response lies on a smooth manifold [9]. Other methods build explicit nonlinear representations, such as Volterra series [361] for unsteady and nonlinear aerodynamic responses [211, 218, 227, 319, 321], or combine lower-fidelity models with correction terms learned from sparse high-fidelity data. Volterra series expand the idea of an impulse response from linear theory into multi-dimensional kernels that capture successive orders of nonlinearity. The problem is that identifying these kernels for high-dimensional applications can be data-intensive [19, 94, 185]. Beyond these classic ROM formulations, ML approaches have rapidly gained traction. Indeed, they can be viewed as powerful nonlinear function approximators that map high-dimensional input data (e.g., a flowfield) to a compact latent representation. Various regression- or classification-based ML methods have already shown promising results in fluid mechanics: from Kang et al.'s [175] flow-regime classification framework, to Duraisamy et al.'s [100] turbulence-model corrections, and Dominique et al.'s [95] regression scheme for wall-pressure spectra and noise predictions. All these examples underscore the importance of carefully balancing model complexity with the quantity and quality of available data, as well as embedding the correct physical constraints to ensure meaningful predictions.

Whether via classical projection-based methods, Volterra expansions, or modern ML, reduced-order modelling has become a key tool in tackling the computational challenges associated with high-dimensional, nonlinear fluid dynamics. Despite their differing formalisms, all ROM frameworks share a common philosophy: extract the essential degrees of freedom from the system response, then leverage those to achieve accurate yet computationally feasible predictions.

Proper Orthogonal Decomposition

POD is a widely used linear method for extracting low-dimensional representations of high-dimensional flow data, particularly useful for the analysis of nonlinear aerodynamic phenomena. Despite the linear nature of its basis functions, POD has demonstrated considerable success in transonic and high-speed flow regimes, primarily due to its ability to preserve the dominant flow structures while significantly reducing computational cost [14, 135, 243]. As a result, it is often used as a benchmark for ROM in aerodynamics [108].

The key insight behind POD is to represent high-dimensional flow snapshots using a smaller set of orthonormal modes that capture the most significant part of the flow energy. Mathematically, these modes are computed by singular value decomposition (SVD) of the snapshot matrix $\mathbf{X}$, which can correspond, for instance, to either the pressure coefficient or the skin friction coefficient on the aerodynamic surface:

$$\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^* = \sum_{i=1}^{n} \sigma_i u_i v_i^* \tag{2.1}$$

Here, $\mathbf{U}$ contains the spatial modes, $\mathbf{\Sigma}$ holds the singular values, and $\mathbf{V}$ represents the temporal modes. To reduce dimensionality, the basis is truncated to a reduced rank $k < n$, retaining only the first k modes that account for 95% of the total energy.

To enhance the versatility of this method, an hybrid approach often referred to as POD with Interpolation (POD-I) can be employed. In this framework, the POD coefficients are interpolated across a parameter space (e.g., Mach number, angle of attack) using Radial Basis Functions (RBF), or alternative interpolation schemes [194, 219, 223, 340, 415]. By interpolating the modal coefficients, the methodology extends POD's applicability to parametric analyses and enables rapid evaluations at untested flow conditions [38].

Although POD-I can effectively capture dominant flow features with only a handful of modes, its inherent dependence on linear modal expansions can be restrictive in regimes where highly nonlinear phenomena play a decisive role [168, 358]. In such cases, more advanced methods, for instance those integrating ML [243, 283] or nonlinear system identification frameworks [53], can be necessary for enhanced predictive accuracy. Nevertheless, POD remains a valuable baseline tool for dimensionality reduction, providing both robust performance and low computational overhead in numerous aerodynamic applications [108].

Dynamic Mode Decomposition

DMD [350] offers efficient approximations of high-dimensional flowfields through low-rank representations. In particular, it excels at extracting coherent spatio-temporal patterns from unsteady flows. Its extension, Dynamic Mode Decomposition with control (DMDc) [271], explicitly incorporates motion inputs, enabling effective reconstruction and forecasting of flow responses under dynamic excitation. DMDc has been successfully applied in various fluid dynamic contexts, including aerodynamic [16, 112, 333] and aeroelastic [111, 278] applications.

Originally introduced by Proctor et al. [271], DMDc models the system evolution via the discrete-time linear dynamical system:

$$\mathbf{x}_{t+1} = \mathbf{A}\mathbf{x}_t + \mathbf{B}\mathbf{u}_t, \tag{2.2}$$

where $\mathbf{x}_t \in \mathbb{R}^m$ is the state vector, $\mathbf{u}_t \in \mathbb{R}^p$ is the control input, and $\mathbf{A} \in \mathbb{R}^{m \times m}$, $\mathbf{B} \in \mathbb{R}^{m \times p}$ are the system and input matrices to be identified.

To estimate the operators **A** and **B**, one collects time-series snapshots from simulations or experiments:

$$\mathbf{X} = \begin{bmatrix} \mathbf{x}_1 & \dots & \mathbf{x}_{n-1} \end{bmatrix}, \tag{2.3}$$

$$\mathbf{X}' = \begin{bmatrix} \mathbf{x}_2 & \dots & \mathbf{x}_n \end{bmatrix}, \tag{2.4}$$

$$\mathbf{U} = \begin{bmatrix} \mathbf{u}_1 & \dots & \mathbf{u}_{n-1} \end{bmatrix}. \tag{2.5}$$

These snapshots are stacked into an augmented data matrix:

$$\mathbf{\Psi} = \begin{bmatrix} \mathbf{X} \\ \mathbf{U} \end{bmatrix}. \tag{2.6}$$

The system matrices are then obtained by solving a least-squares problem via the Moore-Penrose pseudoinverse:

$$\begin{bmatrix} \mathbf{A} & \mathbf{B} \end{bmatrix} = \mathbf{X}'\mathbf{\Psi}^{\dagger}. \tag{2.7}$$

To reduce the model order, truncated SVD is applied to the state matrix:

$$\mathbf{X} \approx \mathbf{U}_r \mathbf{\Sigma}_r \mathbf{V}_r^*, \tag{2.8}$$

Truncation rank r is based on the Frobenius norm of $\mathbf{\Sigma}$, with the number of retained modes selected to minimize below a threshold the root mean square error between the original and the reconstructed state matrix.

The reduced operators are projected onto the subspace spanned by the retained modes:

$$\tilde{\mathbf{A}} = \mathbf{U}_r^T \mathbf{A} \mathbf{U}_r, \quad \tilde{\mathbf{B}} = \mathbf{U}_r^T \mathbf{B}, \tag{2.9}$$

resulting in the reduced-order dynamical system:

$$\tilde{\mathbf{x}}_{t+1} = \tilde{\mathbf{A}}\tilde{\mathbf{x}}_t + \tilde{\mathbf{B}}\mathbf{u}_t, \quad \mathbf{x}_t \approx \mathbf{U}_r \tilde{\mathbf{x}}_t. \tag{2.10}$$

A stabilization procedure is applied to improve the conditioning of the reduced system, especially when unstable eigenvalues appear due to truncation artifact. This procedure modifies the eigenvalues of matrix $\tilde{\mathbf{A}}$, making them all lie inside the unit circle in the complex plane. When an unstable eigenvalue is obtained, a new eigenvalue is calculated using a flip method, which modifies the eigenvalue outside the unit circle and reflect it back inside while preserving phase and mode orientation (refer to [112]).

Although DMD [190, 306] and its controlled extension DMDc [272] offer an interpretable and computationally efficient approach to reduced-order modelling, they share several limitations with POD-based methods that motivate the adoption of ML-based surrogates [293, 308]. First, standard DMD assumes linear dynamics in the chosen observables, which may not adequately capture strongly nonlinear behaviours typical of transonic and turbulent flows, leading to oversimplified reconstructions [190, 351]. Nonlinear extensions, such as extended DMD, kernel DMD, and time-delay variants help address this limitation by enriching the observable space [13, 374]. Second, DMD is sensitive to noise and measurement errors and often requires denoising, truncation, or noise-aware formulations to yield stable and interpretable modes [87, 148, 338]. Lastly, standard DMD lacks robustness when extrapolating beyond the training data regime, hence generalisation typically requires augmentation through nonlinear lifting, control-aware formulations, or additional constraints [272, 308, 374].

2.2 Machine Learning

ML is a branch of Artificial Intelligence (AI) and computer science that leverages data and algorithms to emulate human learning processes, progressively improving predictive accuracy over time. Samuel [301] introduced the term ML by defining it as the study of giving computers the ability to learn without explicit programming. Later, Mitchell [240] formalized this concept in his well-known definition: *A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E.*

In recent years, the synergy between ML and aerodynamics has received increasing attention [187], driven by rapid improvements in algorithms, data availability, and computational resources. These advances have not only accelerated knowledge discovery, but also facilitated the development of more efficient numerical simulations and robust multidisciplinary design frameworks for aerodynamic applications. The central goal of ML is to learn a mapping from input to output variables (i.e., a target function) that effectively generalizes to new data. In many engineering applications, this mapping must capture highly nonlinear relationships within high-dimensional datasets, such as those encountered in CFD aerodynamic analysis. In this work, ML techniques serve two important purposes. First, they help create surrogate models that quickly approximate complex flow physics in the transonic and supersonic regimes, providing significant computational savings [124, 381, 395]. Second, they provide UQ frameworks essential for robust design space exploration [15, 363] and for handling model form uncertainties in high-speed aircraft configurations [258].

ML is mainly divided into supervised and unsupervised paradigms. In supervised learning, algorithms train on labeled data, learning the underlying mapping from inputs (e.g., freestream conditions, geometric descriptors) to known outputs (e.g., aerodynamic coefficients). In this thesis, primarily supervised methods are used in regression tasks, such as predicting pressure coefficients or aerodynamic force distributions from geometric and flow-related parameters. In unsupervised learning, algorithms discover hidden patterns in unlabeled datasets, often used for dimensionality reduction or clustering of flow features to reveal physically relevant structures (e.g., regions of shock-induced separation).

Several classes of ML models are used in aerodynamic analysis. Gaussian processes (GP), for example, provide a probabilistic framework that handles small to medium datasets well by providing both mean predictions and confidence intervals. This capability is particularly useful in the early stages of aerodynamic design, where data availability is limited and uncertainty must be carefully managed [395, 15]. Neural networks (NNs), on the other hand, learn complex patterns in data using layered processing units (neurons). When these networks contain many layers, they are referred to as deep learning methods. Deep networks effectively capture highly nonlinear relationships such as those encountered in transonic flows with strong shocks, boundary layer separation, and vortex shedding. In later chapters, we illustrate how fully–connected, convolutional, and graph-based deep learning models can each be tailored to address different geometric complexities and flow regimes in the aerodynamic test cases of this thesis.

Regardless of the ML algorithm chosen, the learning procedure typically involves (i) specifying the model architecture or kernel function (in the case of GPs), (ii) selecting an appropriate loss function that quantifies the prediction error, and (iii) optimizing the model parameters (weights, biases, hyperparameters) to minimize this loss. Appropriate validation and UQ procedures are essential to ensure that the final model remains reliable under changing flight conditions, geometric variations, or data limitations [363, 258].

Role of Machine Learning in this Thesis

In this research project, ML combines fundamental fluid dynamics with advanced data-driven surrogate modelling. Specifically, the frameworks presented in the following chapters rely heavily on these methods to achieve three core goals:

1. **Efficient Surrogate Modelling for Nonlinear Aerodynamics:** Traditional CFD approaches for transonic or supersonic flows are computationally expensive, especially when integrated into iterative design cycles. By learning from a set of high-fidelity simulations, ML surrogate models enable rapid prediction of aerodynamic loads for new conditions, significantly accelerating shape optimisation and flight performance analysis.

2. **Reduced-Order Modelling and Dimensionality Reduction:** High-dimensional mesh-based simulations generate large datasets of aerodynamics fields. Using ML, we reduce these dimensions while preserving essential physical features. Graph-based neural networks, for example, are used to handle unstructured computational grids in later chapters.

3. **Uncertainty Quantification and Robust Design:** In aerospace design, tolerances for risk are minimal. ML models allow uncertainties in flight conditions or geometry to be propagated and confidence bounds to be estimated for quantities of interest such as lift, drag, or flutter boundaries. This helps to make more robust design decisions.

The upcoming sections build upon these foundational concepts.

2.3 Gaussian Process Regression

Gaussian Process Regression (GPR) is a non-parametric Bayesian approach that provides both predictions and an estimation of uncertainty. Unlike deterministic models, which assume a fixed functional form for the relationship between input and output variables, GPR models the function as a probability distribution over possible functions, enabling it to generalize well even with limited data. Its application spans multiple fields, including fluid dynamics, aerospace engineering, and UQ, where accurate predictions with confidence intervals are essential.

The use of GPR in ML dates back to the work of Seeger [312], who demonstrated their probabilistic nature as an alternative to Multi-Layer Perceptrons (MLP). Later, Neal [250] showed that, under certain conditions, a GP could be interpreted as an MLP with an infinite number of hidden layer units, highlighting its powerful nature in function approximation. Over time, advancements in covariance functions have allowed GPs to capture more complex relationships in the data. Research by Durrande et al. [101] and Gonen et al. [127] introduced more sophisticated covariance functions to improve the representation of complex datasets. Additionally, efforts by Lyu et al. [221] and Wilson et al. [375] incorporated GPs into structured probabilistic frameworks to enhance their expressiveness and scalability.

A key strength of GPR is its ability to incorporate prior knowledge into the modelling process due to its Bayesian inference foundation. Compared to deep learning models, GPR requires fewer training samples to achieve comparable accuracy, making it a practical choice when data collection is expensive or time-consuming. However, a primary challenge is its computational cost, as the inversion of the covariance matrix scales as $O(n^3)$ (with n the size of the training samples), making it computationally prohibitive for large datasets. To address this issue, various sparse approximations and variational inference methods have been proposed, allowing for faster predictions in large-scale applications. Alternatively, dimensionality reduction techniques or multi-fidelity modelling may be required to maintain computational efficiency. Another limitation is the sensitivity to kernel selection, as the choice of the covariance function significantly affects model performance. Unlike deep neural networks, which can automatically learn hierarchical feature representations, GPR heavily relies on kernel engineering to capture data dependencies effectively.

Despite their limitations, GPRs have also been applied in nonlinear system identification. Ross et al. [292] introduced a nonparametric Volterra Kernels Model (NVKM) based on GPs, capable of modelling complex nonlinear dynamical systems. Further applications of GPR in aerospace engineering have highlighted its potential in solving problems with multi-level accuracy and hierarchical modelling frameworks, as shown in the studies by Snyder et al. [324] and Nakamura et al. [248]. These studies emphasize GPR's critical role in UQ, where it is used to propagate uncertainties in both input conditions and model parameters, improving the robustness of aerodynamic predictions.

More recently, Deep Gaussian Processes (DGPs), introduced by Damianou et al. [81], have extended the capabilities of traditional GPRs by introducing hierarchical layers that enable the modelling of highly nonlinear and multi-scale functions. Unlike standard GPs, which assume a single-layer latent function, DGPs propagate uncertainty across multiple layers, making them suitable for problems with strongly nonlinear responses and limited data availability. Their application in aerospace engineering has been particularly promising, as demonstrated by Hebbal et al. [141], who employed DGPs for rocket booster design optimisation, and Rajaram et al. (2020)[280], who applied them as surrogate models for high-dimensional aerodynamic design problems. Studies by Damianou et al. [80] and Vafa et al. [353] have shown that DGPs effectively handle non-stationary aerodynamic responses, often outperforming both traditional GPs and BNNs in regression tasks.

Salimbeni and Deisenroth [300] demonstrated that increasing the depth of DGP models does not necessarily lead to overfitting, even for datasets of small to medium size. Their effectiveness in aerospace engineering has been further validated by their successful implementation in Bayesian optimisation techniques and uncertainty-aware aerodynamic modelling. Additional research by Tsilifis et al. [349] and Scoggins et al. [311]

has explored the use of multi-fidelity GPs for hierarchical UQ, demonstrating their ability to incorporate multiple levels of accuracy into aerodynamic modelling.

Hybrid models that combine GPs with neural networks have been explored as an alternative to purely deterministic methods. Studies by Snyder et al. [324] and Scoggins et al. [311] have demonstrated that combining GPR with deep learning significantly improves prediction accuracy and uncertainty estimation in high-speed aerodynamic flows. These hybrid approaches leverage the strengths of both methodologies: the interpretability and UQ of GPR with the scalability and feature extraction capability of deep learning.

2.3.1 Mathematical Formulation

A Gaussian process is defined as a collection of random variables, any finite subset of which follows a joint Gaussian distribution. It is fully characterized by a mean function $\mu(x)$ and a covariance function $k(x, x')$, which define the expected value and the similarity between outputs for given inputs. The mean function $\mu(x)$ is often set to zero without loss of generality, as it only shifts the predictions by a constant. The covariance function, also referred to as the kernel, defines the structure and smoothness of the function approximation.

Given a training dataset of n input-output pairs $\{X, y\}$, where $X = \{x_1, x_2, ..., x_n\}$ represents the inputs and $y = \{y_1, y_2, ..., y_n\}$ represents the corresponding outputs, Bayesian inference is used to obtain the posterior distribution of the Gaussian process:

$$p(f|X, y) = \frac{p(y|f, X)p(f|X)}{p(y|X)} \tag{2.11}$$

where $p(y|f, X)$ is the likelihood function representing the probability of observing the data given the Gaussian process, $p(f|X)$ is the prior distribution, and $p(y|X)$ is the marginal likelihood acting as a normalizing constant. The posterior distribution enables probabilistic forecasting and allows for the estimation of prediction uncertainties.

For a new input x^*, the predictive distribution follows a Gaussian distribution:

$$p(f(x^*)|X, y, x^*) = \mathcal{N}(\mu(x^*), \sigma^2(x^*)) \tag{2.12}$$

where the mean prediction is given by:

$$\mu(x^*) = k(x^*, X)[K(X, X) + \sigma^2 I]^{-1} y \tag{2.13}$$

and the variance, which quantifies prediction uncertainty, is:

$$\sigma^2(x^*) = k(x^*, x^*) - k(x^*, X)[K(X, X) + \sigma^2 I]^{-1} k(x^*, X)^T \tag{2.14}$$

where $K(X, X)$ is the covariance matrix for the training data, $k(x^*, X)$ represents the covariance between the new input x^* and training inputs X, and σ^2 is the measurement noise variance, assumed to be independent and identically distributed Gaussian noise with mean zero.

The choice of kernel function is fundamental to the performance of the GPR model, as it defines how input data points influence one another. In preliminary studies, we benchmarked several covariance functions and compositions, including RBF, Matérn classes with $\nu = 1/2, 3/2, 5/2$, rational quadratic, linear, and periodic kernels, as well as additive and product combinations. Hyperparameters were learned by maximising the marginal likelihood, and models were compared using negative log marginal likelihood and held-out predictive error. Across the datasets considered, an additive kernel combining RBF and Matérn 5/2 provided the best trade-off between accuracy and calibrated uncertainty, and is therefore adopted in the present work.

The RBF kernel assumes a smooth function and models correlations as an exponentially decaying function of the squared Euclidean distance between inputs:

$$k_{\text{RBF}}(x, x') = \sigma_f^2 \exp\left(-\frac{1}{2l^2}\|x - x'\|^2\right) \tag{2.15}$$

where σ_f^2 is the signal variance, l is the characteristic length scale controlling the influence of distant points, and $\|x - x'\|^2$ represents the squared Euclidean distance between inputs.

The Matérn 5/2 kernel provides a more flexible alternative that allows for less smooth functions compared to the RBF kernel. It is particularly useful when modelling physical systems with localized variations:

$$k_{\text{Matérn } 5/2}(x, x') = \sigma_f^2 \left(1 + \frac{\nu\|x - x'\|}{l} + \frac{(\nu\|x - x'\|)^2}{3l^2}\right) \exp\left(-\frac{\nu\|x - x'\|}{l}\right) \tag{2.16}$$

where $\nu = \sqrt{5}$ is the smoothness parameter, controlling the degree of function differentiability.

The linear combination of the RBF and Matérn 5/2 kernels exploits the complementary strengths of the two functions. The RBF kernel ensures global smoothness, while the

Matérn 5/2 kernel captures localized variations that may be present in transonic flow regimes and other aerodynamic applications:

$$k_{\text{combined}}(x, x') = k_{\text{RBF}}(x, x') + k_{\text{Matérn } 5/2}(x, x') \tag{2.17}$$

By optimizing the kernel hyperparameters using Maximum Likelihood Estimation (MLE), the Gaussian Process model can be fine-tuned to achieve higher accuracy. This combined kernel formulation ensures that the model retains both smooth long-range correlations and short-range variations, making it particularly well-suited for multi-fidelity aerodynamic modelling and UQ.

2.4 Neural Network

The increasing complexity of modern fluid dynamics problems, particularly in transonic and turbulent regimes, often makes traditional computational approaches prohibitively expensive for design optimisation or real-time applications. Within this context, Neural Networks (NNs) have emerged as a powerful family of nonlinear ML methods capable of mapping complex, high-dimensional inputs (e.g., flow variables, geometric parameters) to outputs of engineering interest (e.g., aerodynamic loads, flow-field snapshots).

NNs draw conceptual inspiration from biological neurons. In an NN, inputs propagate through layers of interconnected units, known as neurons. Each neuron performs an affine transformation followed by a nonlinear activation function. These local transformations propagate layer by layer, enabling the model to capture complex, non-intuitive relationships in the data.

Through training, the weights and biases are optimised to accurately approximate the desired function. To ensure that the model learns effectively while maintaining the ability to generalize, the available data is typically divided into a training set for parameter updates, a validation set for monitoring generalizability, and a test set for final performance evaluation.

Two major challenges must be addressed during this learning process. One is overfitting, where the model memorizes particular training examples instead of uncovering broader patterns, leading to poor performance on unseen data. Another is underfitting, where the model remains too simple to capture the complexity of the underlying relationships, resulting in systematically large errors even on the training set. Striking a balance requires careful selection of network architecture and hyperparameters.

At the algorithmic level, NNs rely on forward and backward passes. In the forward pass, the input data propagate through the network to produce predictions. The difference between these predictions and the true output is then quantified using a chosen loss function, such as the Mean Squared Error (`MSE`) or Mean Absolute Error (`MAE`) in regression tasks. While `MSE` is useful when large errors should be strongly penalized, `MAE` is preferable when a balanced approach is needed. These loss measures guide the backward pass, where gradients are computed and parameters updated so that the network progressively reduces the validation error. Modern optimizers like the Adaptive Moment Estimation (Adam) algorithm [181] apply adaptive learning rates to each parameter and track moving averages of first and second gradient moments, helping stabilize and accelerate this iterative refinement.

A further consideration in NN design is the activation function, which introduces non-linearity into each neuron output. The hyperbolic tangent (`Tanh`) function ensures outputs remain within a bounded range, though it can cause gradients to diminish in deeper networks. The Rectified Linear Unit (`ReLU`), which outputs zero for negative inputs and passes positive values unchanged, is computationally simple and mitigates vanishing gradients. However, it can still result in inactive neurons for negative inputs, leading to variants such as Leaky `ReLU` and Parametric `ReLU` (`PReLU`), which retain small but non-zero gradients in negative regions.

By combining thoughtful data partitioning, model parameter regularisation, careful selection of activation functions, and efficient optimisation algorithms, NNs can learn complex aerodynamic or physical patterns and effectively generalize to new scenarios.

2.5 Deep Learning Architectures

As these networks become deeper - leading to the concept of deep learning - there is a need for specialized architectures capable of capturing multi-scale features and complex interactions. This highlights the importance of carefully selecting or designing different network architectures to ensure robust performance when modelling complex aerodynamic patterns.

2.5.1 Fully–connected Neural Network

The fully–connected neural network (FCNN) is the most fundamental neural network architecture. It consists of multiple layers of neurons, with every neuron in one layer connected to every neuron in the next layer. This dense connectivity allows the network to learn complex relationships between inputs and outputs, making FCNNs effective for function approximation and pattern recognition.

FCNNs have been widely used in aerodynamic applications, including predicting the dynamic response of nonlinear systems in subsonic [347] and transonic regimes [330], estimating aerodynamic loads during dynamic stall [367], and serving as surrogate models for efficient aircraft design and optimisation [402]. By accurately capturing complex unsteady aerodynamic behaviour, FCNN-based models reduce computational complexity compared to traditional CFD-based methods.

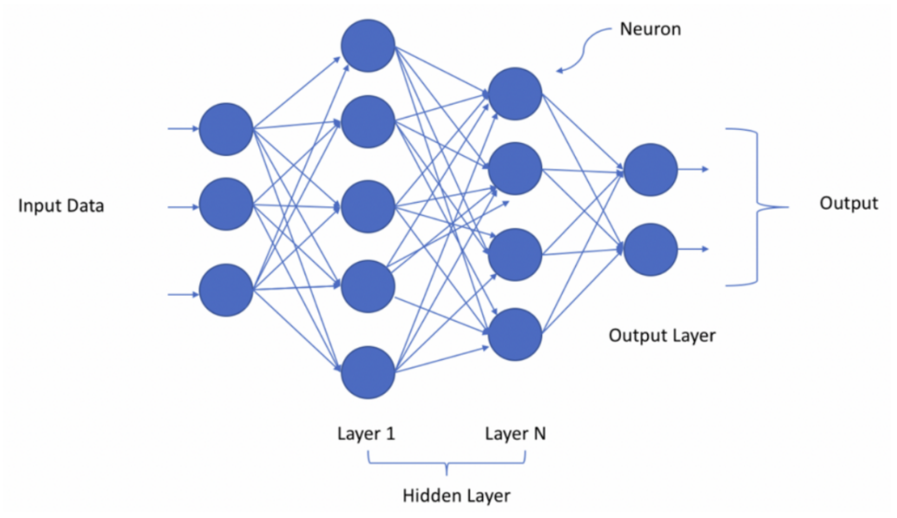


FIGURE 2.1: Fully–connected network architecture.

In a FCNN, each neuron in the ℓ-th layer processes its inputs by applying a weighted sum followed by a nonlinear activation function. Formally, for the n-th neuron in layer ℓ, the output $y_{\ell,n}$ is given by:

$$y_{\ell,n} = \sigma\Big(\mathbf{w}_{\ell,n} \cdot \mathbf{x}_\ell + b_{\ell,n}\Big), \tag{2.18}$$

where $\mathbf{w}_{\ell,n}$ is the weight vector, $b_{\ell,n}$ is the bias term, and σ denotes the nonlinear activation function in layer ℓ.

One of the primary advantages of FCNNs is their universal approximation capability, which allows them to model highly nonlinear functions with sufficient depth and width. They are easy to implement and can generalize well with sufficient training data. However, the full connectivity of the neurons increases the number of parameters, potentially leading to computational inefficiency. Without regularisation techniques, they may suffer from overfitting, where the model memorizes training data instead of generalizing to new samples. In addition, they do not exploit spatial hierarchies in structured data, making them less suitable than convolutional neural networks for certain flowfield predictions. In time-dependent simulations, they can accumulate errors sequentially, which can lead to instability during time marching.

2.5.2 Convolutional Neural Network

Originally proposed by LeCun [196], Convolutional Neural Networks (CNNs) constitute a class of deep learning architectures designed to extract hierarchical patterns from structured data through the convolution operation. By exploiting local connectivity and weight sharing, CNNs preserve spatial relationships while drastically reducing the number of trainable parameters compared with fully connected networks, thus mitigating the curse of dimensionality. This allows the network to capture translation-invariant features and efficiently recognise spatial hierarchies, as illustrated in Figure 2.2.

CNNs rely on three key principles: sparse connectivity, parameter sharing, and translation invariance. Filters operate on small receptive fields to detect local patterns, their weights are reused across the domain, and the resulting features remain robust to position shifts. These properties make CNNs particularly effective for high-dimensional domains such as aerodynamic flowfields, where they can identify coherent structures like vortices and shock waves. For instance, while a $256 \times 256 \times 3$ input would require nearly two million parameters per layer in an FCNN, a CNN achieves comparable representation reusing the same filters throughout the domain, reducing overfitting risk and memory requirements.

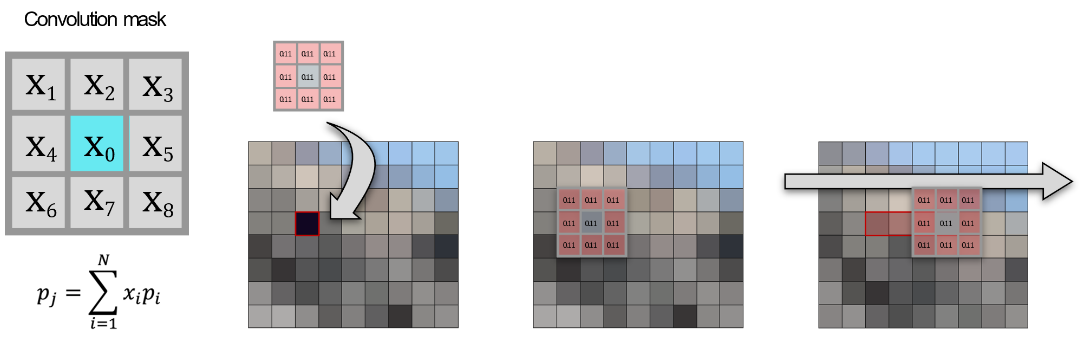


FIGURE 2.2: Example of 2D convolution operation with a 3x3 kernel

While CNNs achieved breakthrough performance in computer vision tasks, most notably with AlexNet [188], their applicability extends beyond image processing to any domain with spatially or temporally structured data. In image analysis, CNNs can process raw pixel matrices directly, eliminating the need for manual feature engineering that characterized earlier computer vision approaches.

Mathematically, the convolution operation in a CNN applies a filter K of size $m \times n$ over an input feature map X, producing an output feature map Y. This operation is defined as:

$$Y(i,j) = \sum_m \sum_n K(m,n)X(i-m, j-n) \tag{2.19}$$

where i, j are spatial indices of the output feature map, and m, n are the dimensions of the filter kernel. The learned filters capture relevant spatial features at different levels of abstraction, enabling the network to detect patterns such as edges, shapes, and textures.

The architectural design of CNNs follows a hierarchical feature extraction paradigm, typically comprising alternating convolutional and pooling layers followed by fully connected layers for final decision-making. Convolutional layers apply multiple filters of varying sizes to extract local features, while pooling operations (max pooling, average pooling, or more sophisticated variants like spatial pyramid pooling) perform dimensionality reduction and provide spatial invariance. This hierarchical structure enables CNNs to learn increasingly complex representations, from low-level features in early layers to high-level semantic concepts in deeper layers.

CNNs have been successfully applied to various aerodynamic studies. For example, Jin et al. [169] developed a data-driven CNN model for predicting velocity fields around a circular cylinder based on pressure measurements. Omata and Shirayama [255] proposed a convolutional autoencoder for capturing temporal variations in unsteady flow structures around an airfoil. Additionally, Fukami et al. [118] applied CNN-based super-resolution techniques to enhance the fidelity of coarse aerodynamic simulations, demonstrating significant improvements over traditional interpolation methods. Rozov and Breitsamter [294] introduced a deep CNN able to capture the nonlinear behaviour of the LANN wing in transonic regime.

Nevertheless, CNNs exhibit some limitations in aerodynamic modelling. One primary challenge is the requirement for structured data grids, as CNNs rely on fixed grid topologies for convolution operations. This constraint necessitates interpolating unstructured CFD meshes onto Cartesian grids, which may introduce interpolation errors and computational overhead. Additionally, CNNs may struggle to generalize to complex, non-homogeneous flow conditions without extensive training data.

2.5.3 Graph Neural Network

Geometric deep learning extends the capabilities of deep learning models to data domains where geometry or structure plays a central role. This is in contrast to traditional approaches such as CNNs, which operate on regular grids (e.g., images or sequential data). Instead, geometric data can take the form of graphs, meshes, or point clouds, which are common in fields such as computer vision, social network analysis, and bioinformatics [45, 43, 383, 407, 413]. Figure 2.3 illustrates how graph convolution differs from standard pixel-wise convolution.

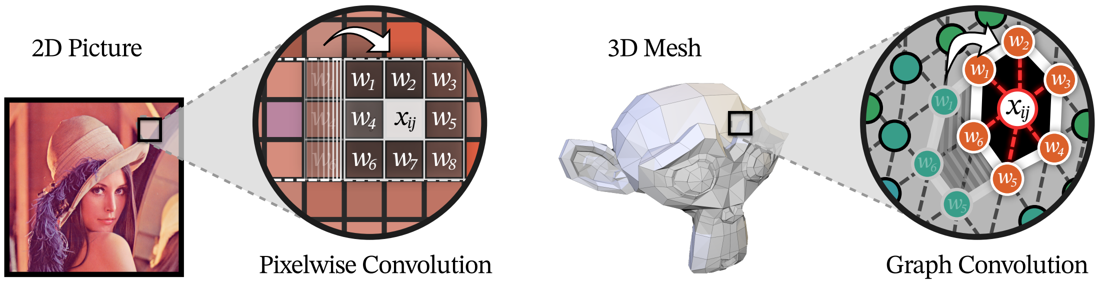


FIGURE 2.3: Visual comparison between pixelwise convolution on a 2D digital image and graph convolution on a 3D mesh.

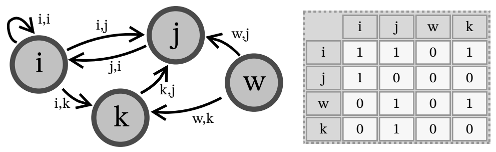
Within this geometric framework, Graph Neural Networks (GNNs) have emerged as a specialized class of models for processing and analysing graph-structured data [383, 407, 413]. In a graph, nodes represent entities, while edges capture relationships or connections between these entities. By efficiently encoding and exploiting the relational information in graphs, GNNs have gained considerable popularity.

Among GNN architectures, Graph Convolutional Networks (GCNs) are particularly suitable for learning both local and global dependencies [263]. First introduced by Kipf and Welling [182], GCNs generalize the convolutional operation to irregular domains by iteratively aggregating features from neighbouring nodes. Here, GCNs are used to extract meaningful features from the graph by applying a graph convolution operator and learning node embeddings that capture both local and global structural information. This message-passing framework is particularly useful for tasks that require an understanding of the relationships between nodes, as GCNs use a convolutional operation similar to classical CNNs to aggregate information from neighbours while also incorporating distance information from the local neighbourhood. The scalability of GCNs is facilitated by parameter sharing - parameters are shared uniformly across all nodes - allowing the model to handle large graphs. In addition, stacking multiple graph convolution layers can reveal hierarchical features and deeper relationships within the network.

Notably, it has been shown that GCNs outperform traditional approaches in handling local nonlinearities [149]. They have demonstrated precise predictions for aerodynamic performances [173] and flowfield properties [253]. Additionally, they result effectiveness in addressing complex time-dependent problems [233] and have proven successful in diverse aerospace applications, including data fusion tasks [203], UQ [204] and multi-objective optimisation [208]. Moreover, GCNs show great potential in dealing with highly deformed meshes and complex geometric regions, such as wing-fuselage interfaces, where standard grid-based convolutional methods may be less effective.

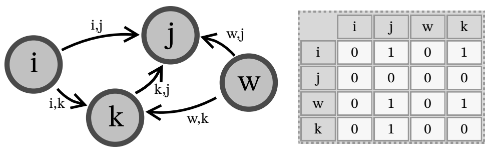
Mathematical Formulation

A graph G consists of nodes N and edges E. An edge (i, j) denotes a directional connection from node i to node j, differing from (j, i) when $i \neq j$. Self-loops are possible if $(i, i) \in E$. Graphs are often illustrated with circles for nodes and arrows for connections. In the graph G shown in Figure 2.4, with nodes $N = \{i, j, k, w\}$, edges are represented by one-way arrows. These connections can be expressed using an adjacency matrix $\mathbf{A}$, where $\mathbf{A}_{ij} = 1$ if $(i, j) \in E$, and $\mathbf{A}_{ij} = 0$. Graphs can also carry edge costs, denoted as e_{ij}, representing distances or other values. In an adjacency matrix with costs, replace 1 with the cost and use ∞ for absent connections. A path $p(i \to j)$ in a graph is a finite series of steps $\langle n_k, n_{k+1} \rangle$ from i to j. A graph is called acyclic if it contains no path $p(i \to j)$ with $i = j$; otherwise, it is cyclic, as illustrated in Figures 2.4a and 2.4b.



	i	j	w	k
i	1	1	0	1
j	1	0	0	0
w	0	1	0	1
k	0	1	0	0

(A) Cyclic



	i	j	w	k
i	0	1	0	1
j	0	0	0	0
w	0	1	0	1
k	0	1	0	0

(B) Acyclic

FIGURE 2.4: Visual representation of a graph diagram and its connectivity matrix.

In the aerodynamic context, a CFD mesh can be viewed as a cyclic graph G wherein each grid point i in the surface mesh G is a node characterized by variables (features), which are for instance positional coordinates or pressure values. The connections between grid points form the edges of the graph, linking target node i with grid points $j \in S$. The nodes features are denoted as y_i, and the weights on edges are denoted as e_{ij}.

Graph connectivity is expressed through the adjacency matrix $\mathbf{A}$, where each entry e_{ij} represents the weight on the edge connecting node j to node i. Edge weights e_{ij} represent the Euclidean distance $\|\mathbf{x}_i - \mathbf{x}_j\|_2$ between grid points. To normalize the edge weights within the range $(0, 1]$, including self-loops with $e_{ii} = 1$, the adjacency matrix is augmented by the identity matrix: $\hat{\mathbf{A}} = \mathbf{A} + \mathbf{I}$. Additionally, since $\forall (i, j) \in E \, \exists (j, i) \in E$ and $e_{ij} = e_{ji}$, the adjacency matrix results symmetric: $\hat{\mathbf{A}} = \hat{\mathbf{A}}^T$. The adjacency matrix may become sparse as the number of nodes increases, and there are efficient techniques for storing such matrices.

Considering the sparsity of both the graph connectivity and the adjacency matrix, a more memory-efficient organisation in Coordinate List (COO) format is adopted. The edge-index matrix has dimensions $n_e \times 2$ (pairs of node indices), and the edge-weight matrix is $n_e \times 1$, where n_e represents the number of edges in the mesh.

Here, GCNs are used to extract meaningful graph features via a message-passing approach that captures both local and global structure. By convolving node embeddings with their neighbors, GCNs aggregate neighborhood features while incorporating distance metrics.

The GCN operator follows a layer-wise propagation rule:

$$H^{(l+1)} = \sigma(\tilde{\mathbf{D}}^{-\frac{1}{2}}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-\frac{1}{2}}H^{(l)}W^{(l)}) \tag{2.20}$$

where $H^{(l)}$ and $H^{(l+1)}$ represent the input and output graphs at layers l and $l+1$, respectively. The matrix $\tilde{\mathbf{A}} = \mathbf{A} + \mathbf{I}_N$ is the adjacency matrix with added self-loops, and $\tilde{\mathbf{D}}$ is a diagonal matrix called degree matrix of the graph, defined as $\tilde{\mathbf{D}}_{ii} = \sum_j \tilde{\mathbf{A}}_{ij}$. $W^{(l)}$ is the trainable matrix for layer l, and σ is the activation function. This equation allows the network to capture local graph structure and propagate information between nodes.

Spectral convolution extends this concept into the Fourier domain [182]. By expressing the filter in terms of Chebyshev polynomials, the convolution is simplified to a computationally efficient form. To mitigate overfitting and simplify modelling, the Chebyshev order K is typically set to 1 [136], yielding:

$$g_\theta * x \approx \theta(\mathbf{I}_N + \mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}})x \tag{2.21}$$

Repeated application of this operator can introduce exploding or vanishing gradients, so a renormalisation trick [182] is applied to stabilize training.

Successive pooling operations propagate information across distant regions of the graph. By stacking k_l GCN layers, the receptive field of each node expands into the k_l-th order neighborhood. Finally, the output of each GCN layer passes through a nonlinear activation σ, allowing deeper networks to learn more complex relationships.

2.5.4 AutoEncoder

High-fidelity simulations often become computationally expensive, making real-time or iterative design optimisation impractical. In particular, the simulation of flows at high Reynolds and Mach numbers requires fine-grid discretisation and high temporal resolution, resulting in large datasets. Autoencoders (AEs) address this problem by distilling high-dimensional inputs into lower-dimensional latent representations that still capture critical flow features. Unlike traditional ROMs that rely on linear reductions, AEs learn nonlinear transformations, making them particularly effective for aerodynamic problems characterized by complex features as shock waves, boundary layer separations, or vortex structures.

An AE consists of two primary components: the encoder and the decoder. The encoder compresses the high-dimensional input data $x \in \mathbb{R}^n$ into a lower-dimensional latent representation $z \in \mathbb{R}^m$, where $m \ll n$, using a transformation of the form:

$$z = f_\theta(x) = \sigma(W_e x + b_e) \tag{2.22}$$

where W_e and b_e are trainable parameters and σ is a nonlinear activation function. The decoder then reconstructs the original data from the latent space via:

$$\hat{x} = g_\phi(z) = \sigma(W_d z + b_d) \tag{2.23}$$

where W_d and b_d represent the decoder parameters. The network is optimised by minimizing the reconstruction loss,

$$L(x, \hat{x}) = ||x - \hat{x}||^2, \tag{2.24}$$

ensuring that the reconstructed output closely approximates the original input.

By learning these nonlinear mappings, AEs capture complex flow dynamics more accurately than linear techniques. They compress high-dimensional CFD data while preserving the underlying physics, significantly reducing memory consumption and computational overhead. As a result, AEs facilitate rapid aerodynamic predictions, making them valuable for iterative design processes and surrogate modelling.

2.5.5 Graph Recurrent Neural Network

Recurrent Neural Networks (RNNs) are specialized neural architectures designed to model sequential data by capturing temporal dependencies through a hidden state. Their weight-sharing mechanism reduces the number of trainable parameters compared to fully–connected networks, improving training efficiency for long sequences. As a result, they are particularly useful for time-series forecasting and predicting dynamic responses in unsteady fluid flow. An RNN maintains a hidden state vector $h_t \in \mathbb{R}^{n_h}$ that evolves over time as a function of the current input $x_t \in \mathbb{R}^{n_x}$ and the previous hidden state h_{t-1}. The update can be expressed as:

$$h_t = \phi(W_{xh} x_t + W_{hh} h_{t-1} + b_h)$$

where W_{xh} and W_{hh} are trainable weight matrices, b_h is a bias term, and ϕ is a nonlinear activation function.

However, classical RNNs often suffer from vanishing or exploding gradients when backpropagating over many time steps, which hinders long-term dependency learning [28]. Long Short-Term Memory (LSTM) architecture addresses this problem using gating mechanisms to selectively retain essential information and thus effectively capture long-term dependencies. RNNs, and their variants, have been widely applied in aerodynamic modelling for unsteady loads, turbulence, and aeroelastic systems [47, 205, 225, 231, 366]. They effectively capture shock-induced instabilities [397], reconstruct time-dependent velocity fields [91], and model turbulence via compressed convolutional LSTMs [242]. By accurately learning unsteady behaviours and dynamic responses, they are well-suited for modelling unsteady flow phenomena [67, 152].

A key limitation of standard RNNs arises when data is defined on non-Cartesian domains, such as the unstructured meshes common in CFD. Traditional recurrent architectures typically assume sequential inputs arranged in a fixed grid or temporal order, making them less suitable for irregular domains. To overcome this, graph recurrent neural networks (GRNNs) integrate GNN operators with recurrent layers, allowing the model to process data represented as graphs. Through this hybrid approach, GRNNs can effectively exploit both the temporal dimension (via RNN-based gating mechanisms) and the spatial relationships (via GNN-based feature aggregation) in non-Cartesian data. Following the mathematical formulation, let $G = (V, E)$ denote the CFD mesh graph with adjacency matrix A and degree matrix D. Given node features $x_t \in \mathbb{R}^{|V|\times n_x}$ at time step t, the hidden state $H_t \in \mathbb{R}^{|V|\times n_h}$ is updated through a combination of recurrent dynamics and graph convolution. With Chebyshev order $K = 1$, the graph convolution operator can be written in the renormalised form:

$$\tilde{A} = D^{-\frac{1}{2}}(A + I)D^{-\frac{1}{2}}, \quad g_\theta * x_t = \tilde{A} x_t \Theta,$$

where Θ is a trainable weight matrix. The GRNN update then is defined as:

$$H_t = \phi\big(\tilde{A} X_t W_x + \tilde{A} H_{t-1} W_h + b\big)$$

Several architectures have been proposed to incorporate graph convolutions into recurrent models. Among them, Graph Convolutional Recurrent Networks (GCRNs) leverage spectral graph convolutions to replace traditional grid-based convolutions, allowing them to handle irregular spatial domains. Similarly, attention-based models have been introduced to enhance the interpretability and adaptability of graph-based temporal learning. Attention mechanisms have revolutionized time-series forecasting by enabling models to focus on the most relevant parts of the input sequence [65, 357]. This capability leads to more accurate and robust predictions, as demonstrated by improvements in maintenance scheduling through estimating icing probabilities on wind turbine blades [66], stable long-term fluid dynamics predictions using transformer-style temporal attention [138], and enhanced design and control of hypersonic vehicles by capturing spatiotemporal turbulence characteristics [98]. Attention mechanisms allow models to dynamically weigh the importance of different time steps, thereby improving the ability to model complex temporal patterns. Furthermore, Spatio-Temporal Graph Convolutional Networks (STGCNs) have demonstrated strong performance in capturing these patterns by processing entire sequences in parallel and applying filters across both spatial and temporal dimensions. This enables STGCNs to effectively model both short- and long-term dependencies, making them well-suited for aerodynamic applications [18].

The following subsections detail the mathematical formulations of these four GRNN architectures, each designed to address specific needs in dynamic prediction on graph-structured data: GRU-based models for efficient short-term memory, LSTM extensions for longer-term dependencies, attention mechanisms for focusing on key time steps, and spatio-temporal graph convolutions for parallel processing of entire time windows.

Gated Recurrent Unit

The combination of GCNs with GRU [67] offers several key advantages when dealing with spatio-temporal data. GRUs are widely used and well-suited for modelling temporal dependencies, but they can not directly used with non-Cartesian domains like graphs, where spatial relationships are irregular. By incorporating graph convolution operators on GRUs, it is possible to improve the generalisation capability of the model by replacing traditional convolution with a graph convolution, which can handle arbitrary graph structures and effectively learn from unstructured data.

Based on the approach in [318] where recurrent networks for fixed grid-structured sequence are introduced, Seo et al. [314] proposed a GCRN architecture for building a generalised GRU that works with unstructured sequence. To generalize the model to graphs, the 2D convolution is replaced by the graph convolution operator, here $*_g$, introduced in (2.21). In particular, GRU cell in GCRN is defined by:

$$\begin{aligned} z_t &= \sigma(W_{xz} *_g x_t + U_{hz} *_g h_{t-1} + b_z) \\ r_t &= \sigma(W_{xr} *_g x_t + U_{hr} *_g h_{t-1} + b_r) \\ \hat{h}_t &= \phi(W_{xh} *_g x_t + U_{hh} *_g (r_t \odot h_{t-1}) + b_h) \\ h_t &= (1 - z_t) \odot h_{t-1} + z_t \odot \hat{h}_t \end{aligned}$$

Here, $W_{xz} *_g x_t$ refers to the graph convolution operation of x_t with spectral filters which are functions of the graph Laplacian L parametrized by K Chebyshev coefficients. z_t is the update gate vector, r_t is the reset gate vector, $\hat{h}_t$ is the candidate activation vector, h_t is the hidden state at time step t, W and U are the trainable weight matrices for the input and hidden states, respectively, σ is the logistic sigmoid function, and ϕ is the hyperbolic tangent function (or other possible activation functions). The operator $\odot$ denotes the Hadamard product, while b represents the biases.

Long Short-Term Memory

LSTM networks [152] are particularly useful when the data involve long-term dependencies, as they contain memory units that can store information over multiple time steps. This makes them potentially suitable for modelling unsteady aerodynamic flows, where past behaviour influences future predictions over long periods of time. While both LSTM and GRU address the vanishing gradient problem and are designed to capture temporal relationships, LSTM includes additional memory structures that allows it to retain information over longer time periods by sacrificing computational power and increasing the number of parameters. The implementation of a convolutional graph based LSTM follows a similar approach presented before with GRU [314], by creating a model that replaces the 2D convolution with the graph convolution operator $*_g$. In particular:

$$\begin{aligned}
i_t &= \sigma(W_{xi} *_g x_t + W_{hi} *_g h_{t-1} + w_{ci} \odot c_{t-1} + b_i) \\
f_t &= \sigma(W_{xf} *_g x_t + W_{hf} *_g h_{t-1} + w_{cf} \odot c_{t-1} + b_f) \\
c_t &= f_t \odot c_{t-1} + i_t \odot \phi(W_{xc} *_g x_t + W_{hc} *_g h_{t-1} + b_c) \\
o_t &= \sigma(W_{xo} *_g x_t + W_{ho} *_g h_{t-1} + w_{co} \odot c_t + b_o) \\
h_t &= o_t \odot \phi(c_t)
\end{aligned}$$

Where i_t is the input gate, f_t the forget gate, c_t the cell state, o_t the output gate and h_t the hidden state, which is the output of the LSTM at time step t. As before, σ is the logistic sigmoid function, ϕ is the hyperbolic tangent function, W represents the trainable weight matrix, and the support K of the graph convolutional kernels is defined by the Chebyshev coefficients. This extension of the standard LSTM architecture enables the model to learn temporal dependencies while also taking into account the spatial structure of the data.

Attention Mechanisms

Incorporating attention mechanisms allows for dynamically assigning importance to different time steps in a temporal sequence, which is especially useful in graph-based models where both complex spatial and temporal dependencies must be captured. Following the work of Bai et al. [18], attention mechanisms can be employed to re-weight the influence of hidden states of a GCRN across time, enabling the model to focus on the most relevant time points for prediction, rather than treating each equally. The model was constructed by combining GCN and GRU to compute both the spatial and temporal domains of the graph, by using the graph convolution operator $*_g$ introduced before. In addition, the attention mechanism is used to compute a context vector that selectively weighs the hidden states of the GCRN.

First, for each time step, the hidden states of the GRU h_t are passed through an attention layer, where attention scores α_t are computed using a softmax function. These scores are then used to weigh the hidden states, resulting in the context vector C, which captures the global variation information.

In particular, given a series of hidden states calculated by the recurrent network for T time steps: $H = \{h_1, h_2, ..., h_T\}$, the attention weights $\alpha_t, 1 < t < T$ are computed using a softmax function on the scores derived from a MLP:

$$e_t = w^{(2)}(w^{(1)}H + b^{(1)}) + b^{(2)}, \;\; \alpha_t = \frac{\exp(e_t)}{\sum_{i=1}^{T} \exp(e_i)}$$

where $w^{(1)}$, $w^{(2)}$, $b^{(1)}$, and $b^{(2)}$ are trainable weights and biases in the MLP. The context vector C is then calculated by the weighted sum and used for implementing the attention mechanism on the GCRN hidden states:

$$C = \sum_{t=1}^{T} \alpha_t h_t$$

By combining GCNs for spatial feature extraction with GRUs and attention mechanisms for temporal modelling, the model can capture both short-term and long-term dependencies in the data.

Spatio-temporal Graph Convolution

STGCN was introduced by Yu et al. [396] as a method to capture temporal dependencies in spatio-temporal data by applying convolutions across the time dimension. Unlike RNNs, which process inputs sequentially, temporal convolutions handle entire sequences at once, allowing for parallelisation and faster computation. The temporal convolution block consists of 1-D causal convolutions followed by a Gated Linear Unit (GLU) to introduce nonlinearity and control the flow of information.

For each node in the graph G, the temporal convolution layer explores K_t neighboring elements along the time axis. This approach does not require padding, and as a result, the length of the sequence decreases by $K_t - 1$ at each layer. Given an input sequence $Y \in \mathbb{R}^{M \times C_i}$ with M time steps and C_i channels, the convolution kernel $\Gamma \in \mathbb{R}^{K_t \times C_i \times 2C_o}$

maps the input to a single output element $[P\ Q] \in \mathbb{R}^{(M-K_t+1)\times(2C_o)}$. The GLU is then applied, splitting $[P\ Q]$ into two parts, and the output of the temporal convolution is given by:

$$\Gamma *_T Y = P \odot \sigma(Q) \in \mathbb{R}^{(M-K_t+1)\times C_o},$$

where P and Q are the inputs of the GLU, $\odot$ denotes the element-wise Hadamard product, and σ is the sigmoid function. The GLU selectively gates the information flow, determining which parts of the input sequence are relevant for capturing dynamic temporal dependencies. Stacking multiple layers of these temporal convolutions enables the model to capture both short- and long-term patterns effectively.

This approach can also be generalised to 3D tensors, where the same convolution kernel is applied to every node $Y_i \in \mathbb{R}^{M\times C_i}$ in the graph G, resulting in the operation $\Gamma *_T Y$ with $Y \in \mathbb{R}^{M\times n\times C_i}$.

2.5.6 Bayesian Neural Network

Bayesian Neural Networks (BNNs) overcome the limitations of traditional data fusion techniques in UQ by incorporating Bayesian inference to estimate distributions over network weights [348]. This probabilistic framework allows BNNs to capture both model and data uncertainties, thereby improving the reliability of predictions [157]. Recent studies, such as those by Meng et al. [239] and Sharma et al. [315], explore the role of BNNs in multi-fidelity models, emphasizing computational complexity and challenges in integrating heterogeneous data sources. To address these issues, Kerleguer et al. [179] propose a hybrid approach that combines GPR with BNNs, demonstrating an effective strategy for fusing diverse datasets while mitigating integration challenges.

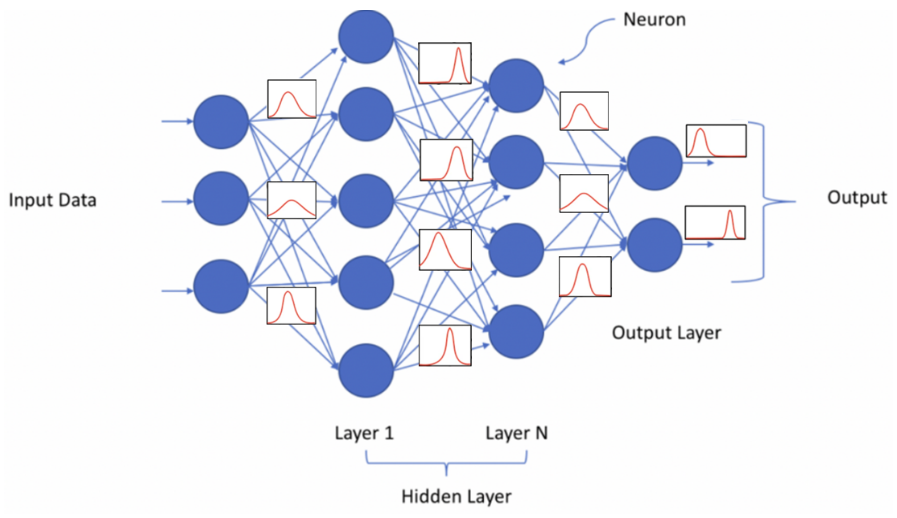


FIGURE 2.5: Bayesian neural network architecture.

BNNs improve standard NNs by incorporating a Bayesian inference to estimate uncertainty in model predictions (refer to Figure 2.5). Instead of learning a single set of weights, BNNs place a prior distribution over the network weights θ, denoted by $p(\theta)$, which is considered a model hyperparameter and is usually randomly initialized. Given a training dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N}$, the objective is to compute the posterior distribution of the weights:

$$p(\theta \mid \mathcal{D}) \;=\; \frac{p(\mathcal{D} \mid \theta)\, p(\theta)}{p(\mathcal{D})}, \tag{2.25}$$

where $p(\mathcal{D} \mid \theta) = \prod_{i=1}^{N} p(y_i \mid x_i, \theta)$ is the likelihood of the data under the weights θ, and $p(\mathcal{D})$ is the marginal likelihood (or evidence).

For a regression task, the predictive distribution at a new input x^* marginalizes out the uncertainty in the weights:

$$p(y^* \mid x^*, \mathcal{D}) \;=\; \int p(y^* \mid x^*, \theta)\; p(\theta \mid \mathcal{D})\, d\theta. \tag{2.26}$$

This integral is typically intractable due to the complexity of $p(\theta \mid \mathcal{D})$. Hence, approximate methods such as the Laplace approximation [222], Hamiltonian Monte Carlo [249], and other Markov Chain Monte Carlo techniques [61] are commonly employed.

Variational Inference (VI) [129] is another popular approach to approximate the true posterior $p(\theta \mid \mathcal{D})$ by introducing a simpler variational distribution $q(\theta)$ and then minimizing the Kullback–Leibler (KL) divergence between the two distributions:

$$\mathrm{KL}\big(q(\theta) \,\|\, p(\theta \mid \mathcal{D})\big) \;=\; \int q(\theta)\, \log\!\left(\frac{q(\theta)}{p(\theta \mid \mathcal{D})}\right) d\theta. \tag{2.27}$$

Because $p(\theta \mid \mathcal{D})$ is unknown in closed form, one typically derives an equivalent objective known as the Evidence Lower BOund (ELBO). Under a regression setting with mean squared error (MSE) as the negative log-likelihood (assuming Gaussian noise), the variational objective can be written (up to constants independent of θ) as:

$$\mathcal{L}(\theta) \;=\; \underbrace{\sum_{i=1}^{N} \mathbb{E}_{q(\theta)}\Big[\big(y_i - f_\theta(x_i)\big)^2\Big]}_{\mathcal{L}_{\text{MSE}}} \;+\; \underbrace{\mathrm{KL}\big(q(\theta) \,\|\, p(\theta)\big)}_{\text{Regularisation}}, \tag{2.28}$$

where $p(\theta)$ is the prior over the weights. This objective encourages $q(\theta)$ to produce accurate predictions for the training data while remaining close to the prior. In practice, the gradient of this loss is estimated via stochastic optimisation, making it suitable to large-scale problems.

2.5.7 Bayesian optimisation of the Hyperparameters

Bayesian optimisation provides a systematic approach to hyperparameter tuning that explores complex, high-dimensional design spaces [323, 35, 390]. Unlike manual tuning or traditional sensitivity studies, Bayesian optimisation treats hyperparameter search as a black-box function evaluation problem and uses a surrogate probabilistic model to identify hyperparameter configurations that maximize (or minimize) a given performance metric (e.g., validation loss) [107, 390]. The process begins by sampling an initial set of hyperparameters, typically via random selection or space-filling strategies like Latin Hypercube Sampling [35]. Each sampled configuration is evaluated by training the model on the available dataset and computing a loss function on a validation set. This evaluation provides initial data on the objective function landscape. With this initial dataset, a surrogate model is constructed (e.g., a Gaussian Process) that approximates the unknown mapping from hyperparameters to objective values [390]. It then balances two main strategies: Exploitation, where the algorithm refines the search around hyperparameter sets already known to perform well, and Exploration, where the algorithm deliberately samples unseen hyperparameter regions to avoid premature convergence to suboptimal points.

This trade-off is governed by an acquisition function that ranks potential hyperparameter configurations based on their expected improvement or other heuristic criteria. The next hyperparameter trial is selected by maximizing this acquisition function, after which the model is re-trained and a new objective value is obtained. The resulting information updates the surrogate model, generating a more accurate representation of the search space and guiding subsequent sampling decisions. This iterative refine/update cycle ends when it converges on a hyperparameter set that provides strong performance within computational constraints [35].

Bayesian optimisation is applicable to continuous, discrete, and categorical hyperparameters. Treating discrete or categorical variables as continuous during optimisation and then rounding or one-hot encoding for training is convenient but generally suboptimal, as it distorts neighbourhood structure, biases the surrogate and the acquisition function, and ignores conditional dependencies [85]. More principled alternatives exist, including sequential model-based optimisation with random forests [29, 158], multi-fidelity optimisation methods [107], and GP models with specialised kernels or latent embeddings for categorical variables [121, 273, 404]. These approaches avoid ad-hoc relaxations and typically improve sample efficiency in mixed-variable settings and can be adopted in future implementations.

After the optimisation phase, the best performing hyperparameters are selected and used for a final model training run, providing more robust and reliable performance than can often be achieved through manual experimentation or grid search methods [107, 390]. Pseudocode for the Bayesian optimisation strategy is presented in Algorithm 1.

Algorithm 1 Bayesian Approach for Hyperparameter optimisation

```
1: Input: Objective function $f$, search space $S$, hyperparameter tuner $T$, number of trials $N_{\text{trials}}$
2: Output: Best hyperparameter set $\theta^*$
3: Initialize hyperparameter tuner $T$ with search space $S$
4: Perform an initial random search to populate $T$
5: Initialize best hyperparameter set: $\theta^* \leftarrow$ None
6: Initialize best objective value: $y^* \leftarrow -\infty$
7: for $i = 1$ to $N_{\text{trials}}$ do
8:     Get the next hyperparameter set from $T$: $\theta_i \leftarrow \text{sample}(S)$
9:     Evaluate objective function: $y_i = f(\theta_i)$
10:    if $y_i > y^*$ then
11:        Update best hyperparameter set: $\theta^* \leftarrow \theta_i$
12:        Update best objective value: $y^* \leftarrow y_i$
13:    end if
14:    Update posterior distribution conditioned on observed data
15:    Set: $\theta_{\text{next}} \leftarrow \arg\max_\theta \text{AcquisitionFunction}(\theta; \text{Posterior})$
16:    Add $\theta_{\text{next}}$ to $T$
17: end for
18: Return: Best hyperparameter set $\theta^*$
```

Chapter 3

Data-Driven Aerodynamic Modelling of 2D Unsteady Transonic Loads

Predicting aeroelastic behaviour in transonic flow conditions is computationally challenging due to the complex interplay between aerodynamic forces and structural dynamics. The presence of shock waves, boundary layer separation, and their interactions with structural modes significantly increases the nonlinearity of the problem, making traditional CFD approaches very resource intensive. In response to this challenge, and in alignment with Research Objective O1 of this thesis, this chapter focuses on the development of data-driven aerodynamic ROMs coupled with a structural model for a supercritical airfoil operating in the transonic regime. The goal is to construct accurate yet computationally efficient models capable of predicting flutter boundaries across a range of flight conditions. The study examines the effectiveness of different ROM architectures, including linear and nonlinear system identification approaches, in capturing the dominant aeroelastic phenomena with minimal computational overhead. The influence of model complexity, training dataset selection, and uncertainty quantification is also discussed, thereby supporting Research Objective O2 and laying the basis for the more complex 3D analysis presented in subsequent chapters.

3.1 Introduction

The accurate prediction of transonic aeroelastic instabilities remains a critical aspect of modern aircraft design. Flutter, a dynamic instability arising from the coupling between aerodynamic forces and structural motion, can lead to catastrophic failure if not properly accounted for during the design phase. Traditional flutter analysis relies on high-fidelity unsteady CFD simulations coupled with computational structural dynamics (CSD) solvers, but this approach is prohibitively expensive, requiring extensive computational resources to resolve unsteady transonic flow structures accurately [184, 339].

A major challenge in transonic aerodynamic modelling is the presence of nonlinear flow phenomena, including shock wave-boundary layer interactions and shock-induced separation. These effects contribute to complex unsteady aerodynamic loads, which are difficult to capture using classical linearized potential-flow-based methods [274]. High-fidelity simulations provide accurate pressure distributions but demand substantial computational resources [112]. Additionally, extensive parametric studies are necessary to capture the full range of aerodynamic behaviours, further increasing the overall computational burden [109].

To address these challenges, data-driven ROMs have been developed as efficient alternatives to full-order CFD-CSD simulations. These models leverage ML techniques to learn the underlying physics from precomputed high-fidelity data, enabling rapid and accurate predictions of unsteady aerodynamic loads and flutter boundaries. Compared to conventional ROMs, ML models offer enhanced flexibility in capturing nonlinear dependencies and generalizing to unseen flow conditions [251, 392]. Furthermore, transfer learning techniques allow ML models to operate effectively even with limited high-fidelity data, reducing computational demands [377].

Recent studies have demonstrated the efficacy of ML approaches in transonic aeroelastic analysis. For instance, researchers have applied neural networks for aeroelastic system identification to predict flutter velocities accurately [25]. Additionally, methods such as DMD with control have been employed to construct ROMs from high-fidelity CFD data, facilitating efficient transonic aeroelastic analyses [112]. Recent advancements in parametric global linearisation models also offer efficient alternatives for predicting flutter boundaries with significantly reduced computational resources [325]. Moreover, advanced surrogate modelling techniques, such as nonlinear state-space identification and Koopman theory-based flutter analysis, have been proposed to improve prediction accuracy while reducing computational costs [214, 241]. The integration of deep learning approaches has also demonstrated success in processing aeroelastic flight test data, enabling near real-time flutter identification [2].

Building on these advances, this study explores the application of ML-based ROMs for predicting flutter onset in a transonic airfoil. The goal is to construct surrogate models that maintain accuracy while significantly reducing computational complexity, thus enabling near real-time flutter analysis. To this end, the methodology involves training ROMs on high-fidelity CFD data using three progressively richer system-identification techniques: linear Finite Impulse Response (FIR), quadratic FIR, and auto-regressive moving-average (ARMA) models [50, 217, 270].

The use of this modelling approach is motivated by both the sparse training regime, only a few damped free-motion signals are available per Mach number, which favours parsimonious, identifiable models and by regime-dependent flow physics observed in the transonic range, where shock-boundary layer interaction, intermittent separation and hysteresis emerge [50, 73, 125, 197, 270, 345]. The linear regression model provides a simple and interpretable baseline that is consistent with quasi-linear behaviour at lower Mach numbers, recovering dominant aerodynamic trends while remaining computationally inexpensive [217]. The quadratic regression model extends this by introducing nonlinear self- and cross-terms, allowing the capture of weak amplitude-dependent effects and mild hysteresis that emerge near $M \approx 0.80$, in line with established nonlinear identification frameworks [20, 34]. Finally, the ARMA formulation incorporates lagged outputs to represent aerodynamic memory effects associated with convective and boundary layer dynamics, thereby improving multi-step roll-out accuracy and ensuring stable behaviour when embedded in the coupled aeroelastic solver [176, 199, 217].

By adopting these three complementary approaches, the study enables a controlled ablation of complexity (linear $\rightarrow$ nonlinear static $\rightarrow$ nonlinear with memory), retains physical interpretability of coefficients, and ensures stable coupling with the structural model. The effectiveness of the resulting ROMs is evaluated by comparing their flutter predictions to full-order CFD-CSD simulations, consistent with prior demonstrations that system identification can accelerate aeroelastic prediction in transonic regimes while retaining fidelity [17, 72, 360].

In addition, this chapter examines the impact of training data selection and model complexity on prediction accuracy. A critical aspect of ROM development is to ensure that the surrogate model captures the essential aerodynamic and structural dynamics without introducing excessive computational overhead. To this end, an uncertainty quantification analysis is performed to assess the robustness of the ROM to variations in input parameters, turbulence modelling, and training dataset composition. By addressing these factors, this research aims to improve the reliability and computational efficiency of ML-based ROMs for transonic flutter prediction, thus bridging the gap between high-fidelity simulation and near real-time prediction capabilities.

3.2 Methodology

This section explains the ROM approach used for rapid prediction of transient aerodynamic loads in the transonic regime. The ROM is coupled with the structural model of the system to perform the aeroelastic analysis and determine the flutter boundary at a given Mach number.

A schematic of the solver workflow is shown in Figure 3.1. The ROM predicts the aerodynamic loads, which are then coupled with the pitch–plunge structural equations of motion. The solver operates in discrete time; at each time step, the aerodynamic model communicates lift and moment coefficients to the structural solver, which returns the updated pitch and plunge states to the aerodynamic model for the next step. This process is iterated until the time–marching solution is complete. Thus, the aeroelastic model is employed to find the flutter point at the Mach number at which the aerodynamic model was generated.

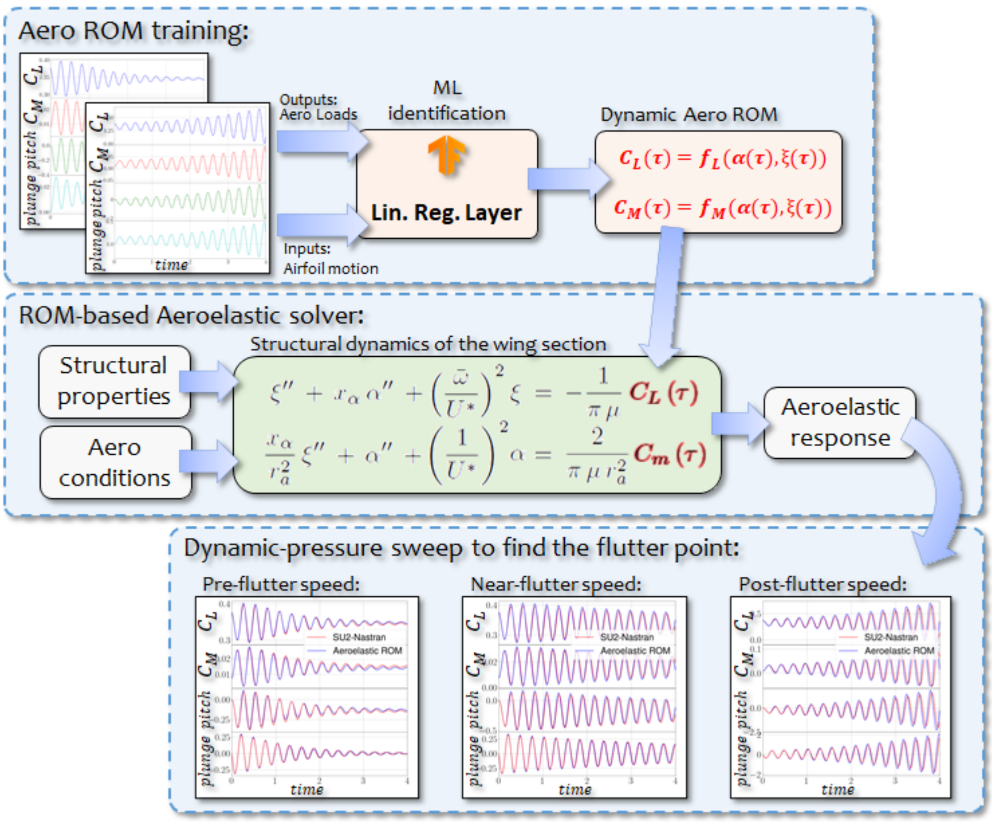

Aero ROM training:
Outputs: Aero Loads
ML identification
Lin. Reg. Layer
Inputs: Airfoil motion
Dynamic Aero ROM
$C_L(\tau) = f_L(\alpha(\tau),\xi(\tau))$
$C_M(\tau) = f_M(\alpha(\tau),\xi(\tau))$
ROM-based Aeroelastic solver:
Structural dynamics of the wing section
Structural properties
Aero conditions
$\xi'' + x_\alpha \alpha'' + \left(\frac{\bar\omega}{U^*}\right)^2 \xi = -\frac{1}{\pi\mu} C_L(\tau)$
$\frac{x_\alpha}{r_a^2}\xi'' + \alpha'' + \left(\frac{1}{U^*}\right)^2 \alpha = \frac{2}{\pi\mu r_a^2} C_m(\tau)$
Aeroelastic response
Dynamic-pressure sweep to find the flutter point:
Pre-flutter speed:
Near-flutter speed:
Post-flutter speed:
SU2-Nastran
Aeroelastic ROM
time

FIGURE 3.1: Schematic of the ML based aeroelastic solver workflow.

The ROM is trained and validated using high-fidelity CFD simulations at selected Mach numbers ranging from 0.74 to 0.84. For each Mach number, five damped free-motion signals are generated by varying the dynamic pressure. Two of these signals, all below the critical flutter limit, are used for training, reflecting realistic constraints in experimental testing. The other signals are reserved for validation.

The focus is now on the realisation of a smart identification system able to capture the physics of the problem. The ML regression model is used to map the structural states (inputs) to the aerodynamic loads (outputs). Following the Bryan concept for combining aerodynamic derivatives in a linear and differentiable manner [33], the lift coefficient $\widehat{C}_L$ is expressed as:

$$\widehat{C}_L(\tau) = C_{L,\text{static}} + \boldsymbol{W}_L^T\,\boldsymbol{x}(\tau) \tag{3.1}$$

where $\widehat{C}_L$ is the approximated lift coefficient and $\boldsymbol{x}$ is the basis of the input parameters (pitch α, plunge ξ and their combinations). The dynamic recovery is done around the static condition $C_{L,\text{static}}$. The matrix $\boldsymbol{W}_L$ contains the learnable weights of the identification.

The nonlinearity has been included in the model by using high–order polynomials, whereas the unsteady effects by adding the values of the target functions as inputs at previous time steps. The resulting basis function for the identification system becomes:

$$\boldsymbol{x} = [1, \alpha, \dot{\alpha}, \ddot{\alpha}, \xi, \dot{\xi}, \ddot{\xi}, \alpha^2, ..., \ddot{\xi}^2, ..., \alpha^p, ..., \ddot{\xi}^p, \alpha\xi, \dot{\alpha}\xi, ..., \alpha\ddot{\xi}, ..., C_L(\tau - \mathrm{d}\tau), C_M(\tau - \mathrm{d}\tau), ...]^T \tag{3.2}$$

Three different architectures have been chosen for solving the problem: linear FIR, quadratic FIR, ARMA. The process started with the simplest model, the linear FIR, which allows in theory to recover the fundamental physics of the dependence of the lift coefficient on the instantaneous angle of attack and pitch rate, and their dependence on Mach number. Then, a derived version of the model, the quadratic FIR, has been adopted in order to take into account nonlinear coefficients. Finally, a third model designed for unsteady phenomena, the ARMA, has been implemented. Here, the current state of the prediction is fed back to the model to predict the state at the next time step.

The structure of these three distinct architectures is reported in Table 3.1. It is important to observe the presence of the second time–derivatives at the previous time step in each model as it is required by the discrete–time method, as reported in Eq. (3.4).

Model	**Architecture**	**Basis function**
1	Linear FIR	$\boldsymbol{x} = [1, \alpha, \dot{\alpha}, \ddot{\alpha}(\tau - \mathrm{d}\tau), \xi, \dot{\xi}, \ddot{\xi}(\tau - \mathrm{d}\tau)]$
2	Quadratic FIR	$\boldsymbol{x} \odot \boldsymbol{x}$, for $\boldsymbol{x} = [1, \alpha, \dot{\alpha}, \ddot{\alpha}(\tau - \mathrm{d}\tau), \xi, \dot{\xi}, \ddot{\xi}(\tau - \mathrm{d}\tau), \alpha\dot{\xi}]$
3	ARMA	$\boldsymbol{x} = [1, \alpha, \dot{\alpha}, \ddot{\alpha}(\tau - \mathrm{d}\tau), \xi, \dot{\xi}, \ddot{\xi}(\tau - \mathrm{d}\tau), \alpha\dot{\xi}, C_L(\tau - \mathrm{d}\tau), C_M(\tau - \mathrm{d}\tau)]$

TABLE 3.1: The three ROM architectures adopted in the current work. $\odot$ denotes the Hadamard product.

The system-identification model was implemented using the deep learning Python library `TensorFlow/Keras` [1]. A single-layer linear regression architecture was selected, with Adam employed as the optimisation algorithm to minimize the `MSE` loss function. The network architecture adopted in this study, along with the selected hyperparameter values, can be found in Appendix A.1.

The accuracy of the generated ROM was evaluated through a multi-step validation process. In a pointwise test, the aerodynamic loads predicted at each time step of a validation signal are directly compared to the corresponding CFD outputs, confirming that the model can reproduce the reference data under identical input conditions. Next, a time-marching validation is performed in which the ROM is subjected to a prescribed motion (e.g., pitch-plunge motion) and generates aerodynamic loads that are evaluated against the CFD results. In the case of FIR model, this validation step yielded results identical to the pointwise validation. Finally, free-motion validation is performed by embedding the ROM in a structural solver that uses the pitch-plunge equations of motion. The system is then integrated in time with no external forced motion, allowing the dynamic response of the airfoil to emerge from the coupling of the aerodynamic and structural equations. The aerodynamic loads and structural states for this final case are compared to the fully coupled CFD–CSD reference (`SU2-Nastran`), providing confidence that the ROM approach can capture the essential physics of flutter.

3.3 2D Test Case

The test case adopted here is an elastically suspended NASA–SC(2)0414 airfoil with two degrees of freedom (Figure 3.2), pitch denoted by α and plunge displacement denoted by h, which define the motion around the elastic axis (e.a.). This airfoil represents the Benchmark Super Critical Wing (BSCW) section from the AIAA Aeroelastic Prediction Workshop [1].

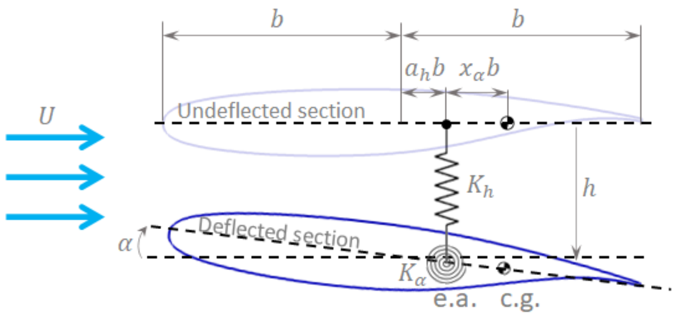


FIGURE 3.2: Schematic of an elastically suspended airfoil section.

The motion of the system can be described as:

$$\begin{cases} m\ddot{h} + S_\alpha\ddot{\alpha} + C_h\dot{h} + K_h h = -L \\ S_\alpha\ddot{\alpha} + I_\alpha\ddot{\alpha} + C_\alpha\dot{\alpha} + K_\alpha\alpha = M \end{cases} \tag{3.3}$$

The motion is restrained by a torsional and linear springs denoted by K_α and K_h, respectively. According to the convention, the lift L is defined positive upward, while

[1] `https://nescacademy.nasa.gov/workshops/AePW3/public/wg/highangle`

the plunge displacement h is positive downward. airfoil mass, first moment of inertia and second moment of inertia are defined by m, S_α and I_α, respectively. b indicates the airfoil semi–chord. $a_h\, b$ represents the position of the elastic axis from the airfoil mid–chord b and $x_a\, b$ is the distance between the centre of gravity and the elastic axis.

In non–dimensional form, the equations of motion become:

$$\begin{cases} \ddot{\xi} + x_\alpha \ddot{\alpha} + 2\bar{C}_h \dfrac{\bar{\omega}}{U^*}\dot{\xi} + \left(\dfrac{\bar{\omega}}{U^*}\right)^2 \xi = -\dfrac{1}{\pi\mu} C_L(\tau,\alpha,\xi) \\ \dfrac{x_\alpha}{r_a^2}\ddot{\xi} + \ddot{\alpha} + 2\bar{C}_\alpha \dfrac{1}{U^*}\dot{\xi} + \left(\dfrac{1}{U^*}\right)^2 \alpha = \dfrac{2}{\pi\mu r_a^2} C_M(\tau,\alpha,\xi) \end{cases} \tag{3.4}$$

where $\xi = h/b$ is the non–dimensional plunge and $\tau = t\, b/U$ is the non–dimensional time.

The last step consists of the transformation of this set of second order, ordinary differential equations into a set of first order, ordinary differential equations which have been solved in time with a Runge–Kutta time–integration solver.

$$\begin{cases} \dfrac{d\mathbf{y}}{dt} = \mathbf{f}(t,\mathbf{y}) \\ \mathbf{y}(t=0) = \mathbf{y}_0 \end{cases} \tag{3.5}$$

The aerodynamic loads were computed with high–fidelity CFD simulation for ensuring an high level accuracy in the model, as this test case is characterized by strong and nonlinear viscous effects.

Background physics

A background study of the NASA–SC(2)0414 supercritical airfoil is introduced in order to highlight the physics that the ML model has to predict.

The unsteady–state analysis were carried out employing a python–based fluid–structure–interaction solver in `SU2-Nastran` package [113]. Unsteady Reynolds–averaged Navier–Stokes (URANS) equations are discretized using `SU2` [104], a cell–centred finite volume method software that converts the partial differential equations into a set of ordinary differential equations. The JST central scheme with artificial dissipation is adopted for the discretisation of convective flows. The gradients of the flow variables are calculated using a Green–Gauss method. The time integration is based on the Euler Implicit Method, setting a fixed Courant number equals to 2. The linear solver biconjugate gradient stabilisation is chosen, with ILU preconditioner applied. The one equation Spalart–Allmaras turbulence model is used for closure of the RANS equations. URANS simulations were all computed by restarting the solution from the steady–state configuration. $AoA_0 = 0$ [deg] was set as initial condition which drives the motion of the not symmetric airfoil section. Also initial displacements of the modes and artificial modal damping were set to zero at $t = 0$.

The mesh used for the NASA–SC(2)0414 airfoil is an O–type structured mesh with $6.2 \cdot 10^4$ elements. A $y^+ = 0.5$ is adopted, after a preliminary mesh convergence study that ensured an adequate resolution of the boundary layer and shock wave. The trailing edge of the airfoil is cut off at 99% of the chord. The farfield extends for a length equals to 100 times the airfoil chord. An impression of the grid can be obtained from Figure 3.3.

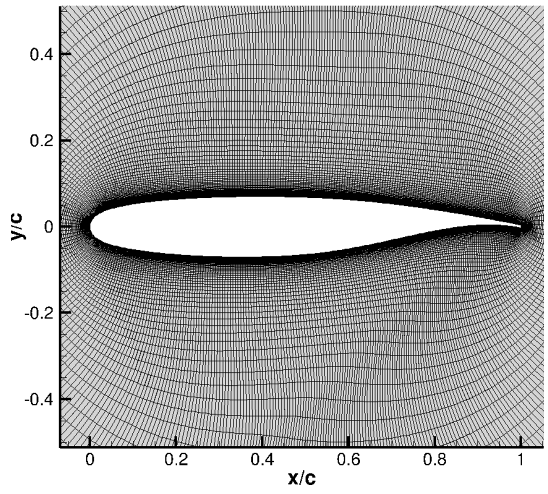
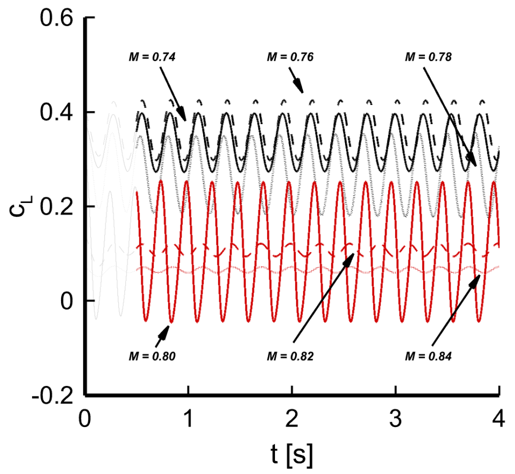


FIGURE 3.3: O–type structured mesh for the NASA–SC(2)0414 airfoil.

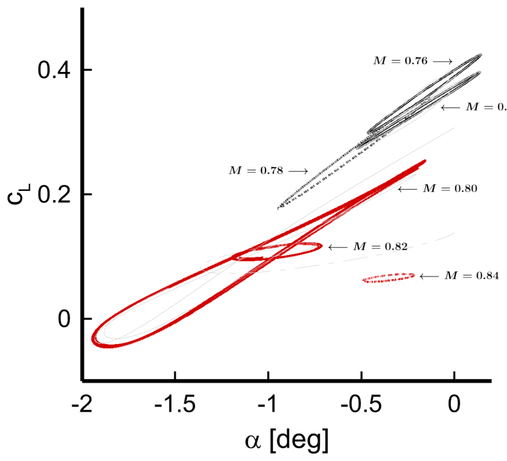
Figure 3.4a reports the aeroelastic C_L response at the flutter dynamic pressure q_F for different freestream Mach numbers. It appears that, since the AoA is the same for each case, there is a nonlinearity related to the Mach number which can already be seen at $t = 0$ where the c_L static is different for each M. This nonlinearity may interact with the one of the dynamic part, i.e. movement of the shock wave and its interaction with the boundary layer, and lead to a more complex prediction of the dynamical system response. Moreover, looking at Figure 3.4b-3.4c, an hysteresis curve may be seen on the c_L response at $M = 0.80$, this change in the pattern may be related to the complex physics of this aeroelastic test case or to the turbulence model, and so to the RANS solver which is not able to predict these complex phenomena.

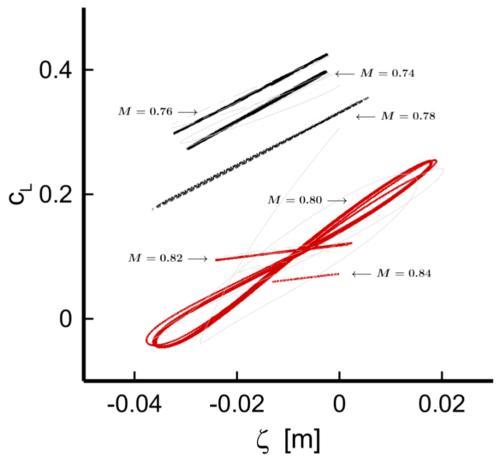
Figure 3.5 shows the influence of the Mach number on the airfoil flutter. The instantaneous Mach number contour is plotted at q_F for the minimum C_L value after three seconds of aeroelastic response in order to allow the complete development of the flowfield (Figure 3.4). The black solid line represents the critical Mach number.

The shock motion is locked–in with the airfoil response. A weak shock is present on the upper surface at $M = 0.74$. The increase in the Mach number leads to a strengthening of the upper shock wave and a formation of a smaller shock wave on the lower surface



(A) Aeroelastic C_L response over time.





(B) Pitch influence on the aeroelastic C_L response.

(C) Plunge influence on the aeroelastic C_L response.

FIGURE 3.4: Aeroelastic C_L response at q_F for the six Mach numbers considered. Shadow lines denote the transitory phase. ($Re = 4.49 \cdot 10^6$ and $AoA_0 = 0$ [deg]).

of the airfoil. At $M = 0.76$, according to the classification of Tijdeman [344], the lower shock wave is of Type B, associated with a periodic appearance and disappearance during the airfoil motion. Above $M = 0.80$, the system of shock waves is always present.

For Mach numbers up to $M = 0.78$, the flow is characterized by a linear behaviour. In fact, the boundary layer remains attached to the airfoil and the shock wave remains in an almost fixed position. At $M = 0.80$, the flow structure becomes nonlinear as there are large fluctuations of the shock wave size and an intermittent boundary layer separation and re-attachment. After $M = 0.82$, the shock wave behaviour results unaffected by the movement of the airfoil as the variation in size as well as in position becomes negligible.

The increase of the Mach number causes a shock–induced separation bubble, which strongly influences the shock wave position by pushing it towards the leading edge.

This strongly coupled interaction of the boundary layer separation with the shock wave motion led to a switch in the ROM architecture, such that its complexity increases in order to capture critical nonlinearities.

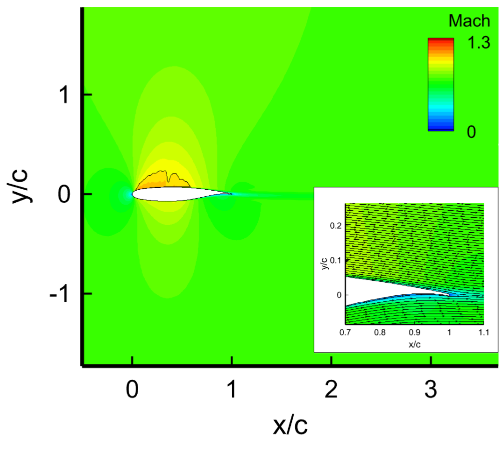
For locating the flutter point, where the damping of the aeroelastic system vanishes and the system exhibits undamped vibrations, time–based flutter–prediction simulations were used. They consist of discrete–time integration of the motion equations at a given speed and observe the damping of the response. The stability is assessed by post–processing the damping and the frequencies of the aeroelastic response. In particular, the bisection method was employed for locating the flutter point. The problem of this method is that using unsteady–state CFD simulations to compute the aerodynamic forces (C_L and C_M) at each time step is impractical as several responses of multiple time steps are required to locate the flutter. As a result, here it is possible to understand the importance of an alternative, more efficient approach for predicting the flutter point, such the ML model here implemented. Figure 3.6 depicts the approach used to find the flutter point. The illustration, in particular, shows the case run at $M = 0.80$ and $Re = 4.49 \cdot 10^6$.

The flutter analysis is performed at several Mach numbers in order to obtain the dynamic pressure q_F at each flow regime, known as flutter boundary. The dynamic pressure is a measure of the coupling between the surrounding fluid and the structure. At low values of dynamic pressure, there is a weak coupling and the structure vibrates predominantly unaffected by the fluid. As the dynamic pressure increases, the aeroelastic system becomes strongly coupled and energy is then exchanged between the fluid and the structure sub–components. To maintain the flow similarity throughout the q_∞ sweep, Mach and Reynolds numbers are kept constant. Hence, the dynamic pressure is varied by changing temperature, which is equivalent to changing flight speed and altitude at the same time.

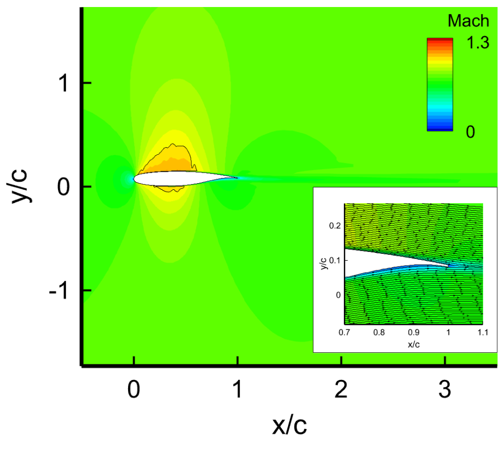
3.4 Results

In this section has been assessed the capability of the ML aeroelastic model to predict the flutter boundary of the BSCW wing section for the range of Mach numbers between 0.74 and 0.84. The structural proprieties adopted for the NASA–SC(2)0414 Supercritical airfoil are taken from the BSCW wing test case [143, 145, 282]. Here, the structural damping is neglected and the positions of the elastic axis and the centre of gravity are coincident at the mid–chord.

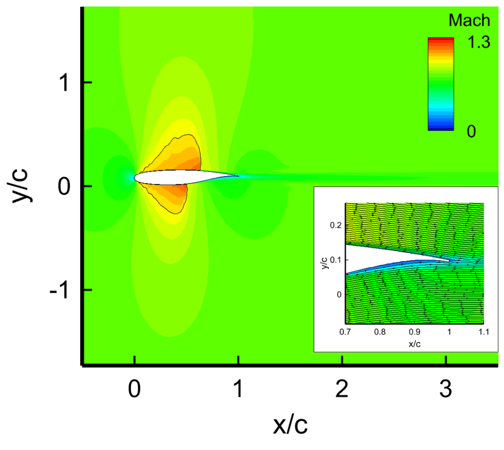
The flutter dynamic pressure q_F results at the different Mach cases for the three model architectures are reported in Table 3.3. The linear FIR is in good agreement with the `SU2-Nastran` results up to $M = 0.80$, with error consistently below $\sim 3\%$. By contrast,



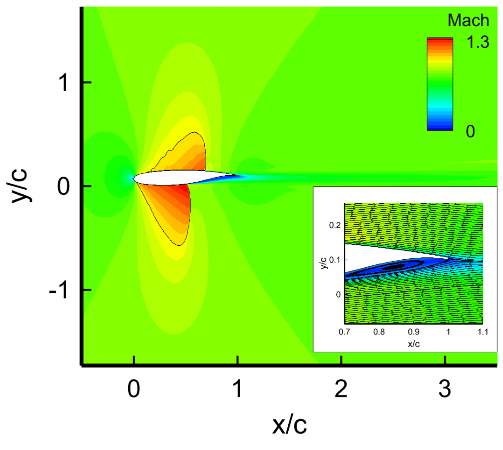
(A) $M = 0.74$, $q_F = 4574$ Pa



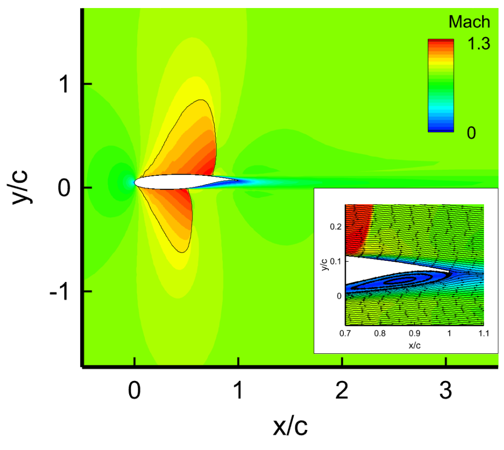
(B) $M = 0.76$, $q_F = 4580$ Pa



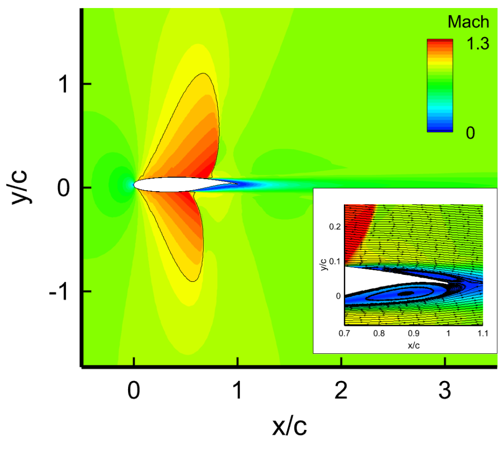
(C) $M = 0.78$, $q_F = 5437$ Pa



(D) $M = 0.80$, $q_F = 8383$ Pa



(E) $M = 0.82$, $q_F = 9562$ Pa



(F) $M = 0.84$, $q_F = 9582$ Pa

FIGURE 3.5: Effect of the Mach number on the instantaneous Mach number contours at q_F in the unsteady–state case. Black solid line marks the critical Mach number. $Re = 4.49 \cdot 10^6$, $AoA = 0$ [deg].

above this Mach value, this simple linear model showed numerical divergence, suggesting that it fails to replicate the more complex phenomena, such as the nonlinear

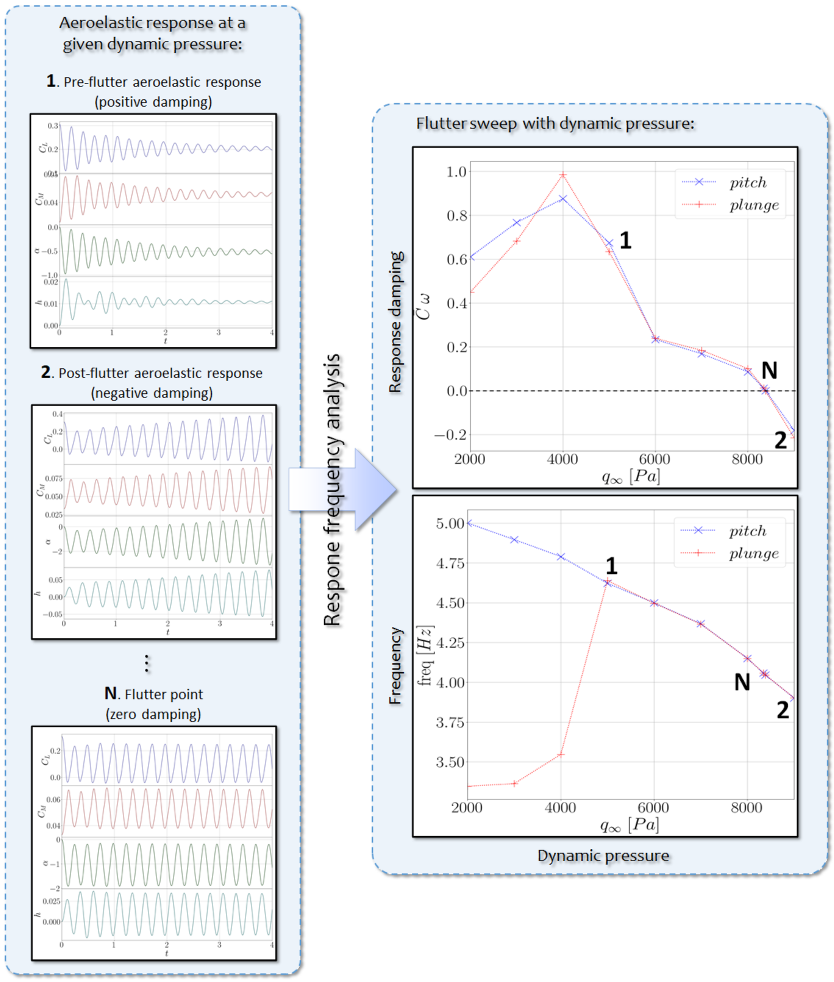


FIGURE 3.6: Flutter analysis procedure in time-domain. The dynamic pressure sweep corresponds to the case at $M = 0.80$ and $Re = 4.49 \cdot 10^6$.

effects from the separated boundary layer.

Thus, the same sweep has been performed with the quadratic FIR, as it includes nonlinear terms. In this case, the model is in good agreement along the Mach range. However, this nonlinear model failed to find the flutter at $M = 0.80$ due to excessive damping.

To complete the analysis, the ARMA model was used with the aim of capturing the unsteady phenomena in different regimes by introducing the model dependence on the previous state. The q_F is also in good agreement at lower transonic regimes but the error gradually increases with the Mach number.

Overall, all ROM architectures are suited to flow conditions at $M < 0.80$. Therefore, a linear model is adequate to represent fluctuating shock waves with attached boundary layer. On the other hand, at $M > 0.80$, the flutter–prediction error becomes larger and the robustness of the model decays. The nonlinear flow effects from the boundary layer separation past the shock wave are therefore more difficult to represent. In conclusion, $M = 0.80$ represents a ”switch” point in the model architecture and this is trivial as

BSCW properties	
K_α	$4018.64\,\mathrm{Nm\,rad^{-1}}$
K_h	$38484.12\,\mathrm{N\,m^{-1}}$
C_α	0
C_h	0
m	$87.91\,\mathrm{kg}$
I	$3.765\,\mathrm{kg\,m^2}$
c	$0.4064\,\mathrm{m}$
a_h	0
x_α	0
M	0.74, 0.76, 0.78 0.80, 0.82, 0.84
AoA	0 deg
Re	$4.49 \cdot 10^6$
q_∞	changed with T_∞

TABLE 3.2: Structural and fluid-condition properties of the BSCW model.

this flow condition is characterized by intermittent boundary layer separation and re–attachment.

Model	$M = 0.74$	$M = 0.76$	$M = 0.78$	$M = 0.80$	$M = 0.82$	$M = 0.84$
`SU2-Nastran`	4574.7	4580.0	5437.1	8382.8	9562.4	10508.6
#1: Linear FIR	4441.8	4450.5	**5423.6**	8497.2	FNF	FNF
#2: Quadratic FIR	4534.0	**4532.9**	FNF	**8067.7**	**10000.5**	**10000.8**
#3: ARMA	**4569.2**	4522.0	5637.2	8855.1	10386.8	FNF

a) Flutter dynamic pressure [Pa].

Model	$M = 0.74$	$M = 0.76$	$M = 0.78$	$M = 0.80$	$M = 0.82$	$M = 0.84$
#1: Linear FIR	-2.90%	-2.83%	**-0.25%**	1.37%	FNF	FNF
#2: Quadratic FIR	-0.89%	**-1.03%**	FNF	**-3.76%**	**4.58%**	**4.37%**
#3: ARMA	**-0.12%**	-1.27%	3.68%	5.63%	8.62%	FNF

b) Flutter dynamic pressure error to benchmark.

TABLE 3.3: Flutter results for the different Mach cases. The results from the three different ROM architectures are compared to the reference benchmark in `SU2-Nastran`. FNF stands for 'flutter not found'. The best ROM is highlighted in bold.

The flutter boundaries for each ROM architecture are plotted in Figure 3.7. In general, the boundaries are well captured by any model but the discrepancies are more obvious at the higher Mach numbers. It is interesting to note the increase in the flutter dynamic pressure as the Mach number gets bigger, with localised large changes in the range $0.78 < M < 0.82$.

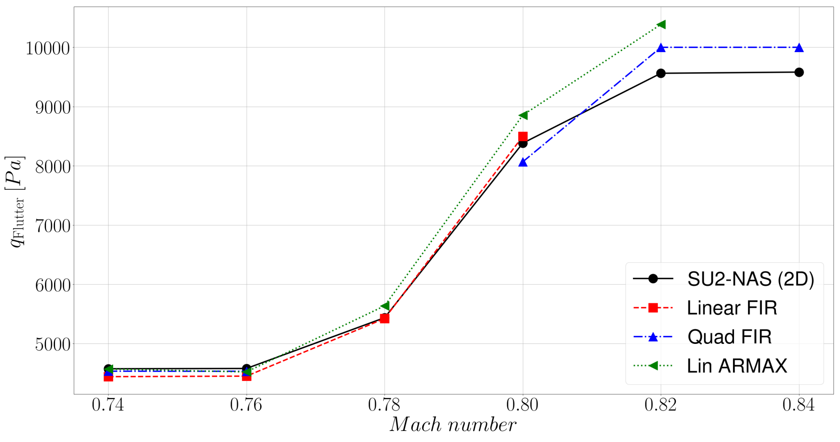


FIGURE 3.7: Flutter dynamic pressure q_{F} with Mach for the different models.

3.4.1 Applicability of ROMs from other Mach numbers

Here, the applicability of using ROMs generated at one Mach number to predict the flutter dynamic pressure at other regimes was evaluated. Table 3.4 shows the error for each model architecture and Mach number against the high-fidelity `SU2-Nastran` results.

Each table represents a matrix of test cases, which is structured in three blocks: the main diagonal, the upper triangular and the lower triangular matrices. An element on the main diagonal conveys the generation of a ROM at a specific Mach number and its application to identify flutter at the same Mach number at which the ROM was generated. An element from either the lower or upper triangular matrix indicates the application of a ROM at a Mach number different from that at which it was generated. In particular, elements belonging to the upper triangular matrix reflect the application of a ROM at higher Mach numbers than the Mach number at which it was generated, and vice versa for the elements on the lower triangular matrix.

It is not unexpected to observe that the smallest errors are found along the main diagonal matrix. However, for $M < 0.80$ it is still possible to obtain reasonable results with ROMs trained at another Mach number, provided the latter is also below that threshold. Particularly for $M < 0.78$, results were in good agreement even with a ROM from another Mach. This is another confirmation of the linear behaviour of the aerodynamics in this region.

By contrast, for Mach at 0.80 and above, it becomes more difficult to find the flutter point with models trained at another Mach condition. It can be concluded that the stronger nonlinear phenomena occuring at higher transonic regimes is practically only possible to capture with ROMs generated from data at similar flow conditions.

Training Mach	Test Case Mach					
	0.74	0.76	0.78	0.80	0.82	0.84
(A) Model #1: Linear FIR						
0.74	2.90	3.28	18.42	45.59	FNF	FNF
0.76	2.32	2.83	18.39	44.84	FNF	FNF
0.78	18.11	17.81	0.25	33.01	40.24	41.52
0.80	64.94	64.46	36.10	1.37	11.75	12.13
0.82	FNF	FNF	FNF	FNF	FNF	FNF
0.84	FNF	FNF	FNF	FNF	FNF	FNF
(B) Model #2: Quadratic FIR						
0.74	0.89	6.14	FNF	FNF	FNF	FNF
0.76	5.44	1.03	FNF	FNF	59.22	15.99
0.78	0.63	4.57	FNF	FNF	FNF	4.36
0.80	10.58	11.39	10.27	3.76	96.86	16.04
0.82	FNF	FNF	86.92	FNF	4.58	FNF
0.84	FNF	FNF	FNF	4.58	6.10	4.37
(C) Model #3: ARMA						
0.74	0.12	1.09	15.72	44.93	FNF	FNF
0.76	0.28	1.27	16.22	46.03	FNF	FNF
0.78	29.63	29.77	3.68	34.06	38.80	FNF
0.80	87.83	91.93	60.61	5.63	1.70	7.81
0.82	FNF	FNF	86.92	1.14	8.62	23.67
0.84	FNF	FNF	FNF	FNF	FNF	FNF

TABLE 3.4: Flutter dynamic–pressure error (%) against benchmark at each Mach number (columns) with a model trained at a different Mach (rows). *FNF* indicates "flutter not found."

3.4.2 Training-signal uncertainty

Generating ROMs can increase the uncertainty in the results because classical neural networks are deterministic and do not handle uncertainty properly. However, sensitivity analysis of key parameters can still be performed.

The signals used to identify the ROM may be contaminated by both aleatoric and epistemic effects. Measurement data from wind-tunnel or flight tests carry facility and instrumentation uncertainties, while CFD time histories can include numerical noise due to discretisation, solver tolerances, turbulence-model closures, mesh and time-step choices. These sources of variability can bias identified dynamics and, if unaccounted for, lead to overconfident predictions. Beyond these intrinsic sources of uncertainty, the high cost of generating training data further constrains the choice of training signals. The results reported so far are based on models trained with dynamic pressure pairs well below the q_∞ suitable for in-flight testing. The ability of the ML model to reproduce the physics is likely to be strongly influenced by such a limitation. Therefore, our goal was to explore the uncertainty of flutter prediction for arbitrary choices of dynamic pressure for training signals. We hope to gain confidence in the robustness of our identification approach to limited, expensive training data.

To do this, at each Mach case, the most robust architecture was selected and a family of ROMs were generated by changing the pair of training datasets (different dynamic-pressure responses). Flutter analyses were then computed for each ROM, obtaining a scatter of results for each Mach case, representing uncertainty to training-signal choice. This procedure is conceptually related to ensemble-based approaches, where multiple models are trained under varying initial conditions or datasets to estimate predictive uncertainty [40, 191], although here the variation arises solely from the limited availability of training signals rather than an explicit ensemble design.

The resulting confidence intervals are plotted in Figure 3.8. The boxplots display the 25%, 50% and 75% percentiles, as well as outlier whiskers of length 1.5 times the box extension, known as inter-quartile range (IQR). The original ROM chosen to report the results in Table 3.3 and the high-order benchmark are also included to assess the systemic error and confidence attainable in each case.

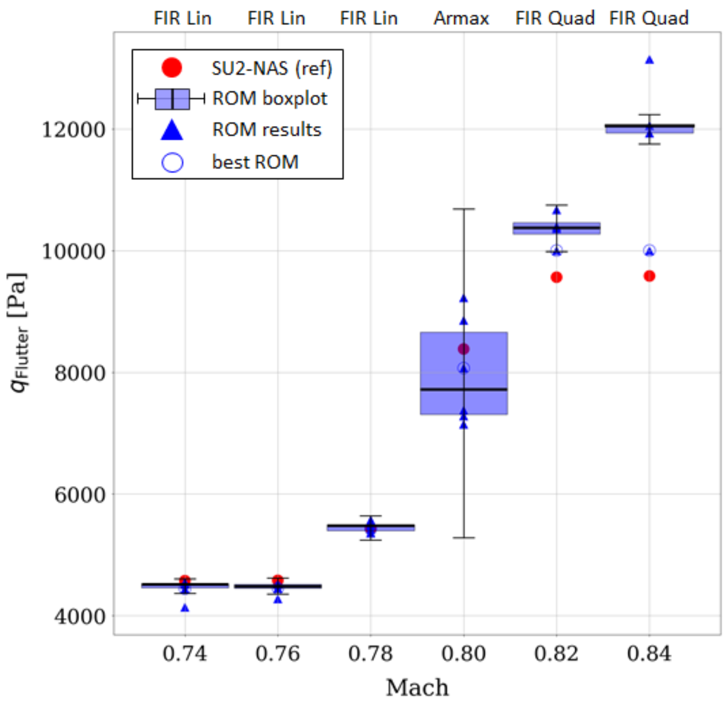


FIGURE 3.8: Uncertainty of the flutter prediction to the dynamic pressures used for the training signals. The boxplots denote confidence on the model performance. The best performing model at that Mach number is reported.

We observe that uncertainty and error from training response choice is small at the lower Mach numbers. The linear behaviour of the fluctuating shock-wave without boundary layer separation observed at $M < 0.80$ regimes likely explains the robustness against the training signals. This would imply that we can confidently build ROMs capable of capturing the physics correctly. In line with the results seen so far, a step change occurs at Mach 0.80: the uncertainty from the training signals becomes significantly large. Yet, the average error came close to the reference value. Therefore, the correct training signal is a difficult choice. As such, an ensembling approach, i.e. a combination of differently trained models, would be convenient to gain confidence on the results. At higher Mach numbers, however, the error is consistently large. The nonlinear phenomena arising from the separated boundary layer are not well predicted by any training signal. In conclusion, the current ROM architecture is inadequate for these higher transonic regimes. It is therefore necessary to investigate an identification system better suited to the more complex flow regimes, like the one proposed in Chapter 4.

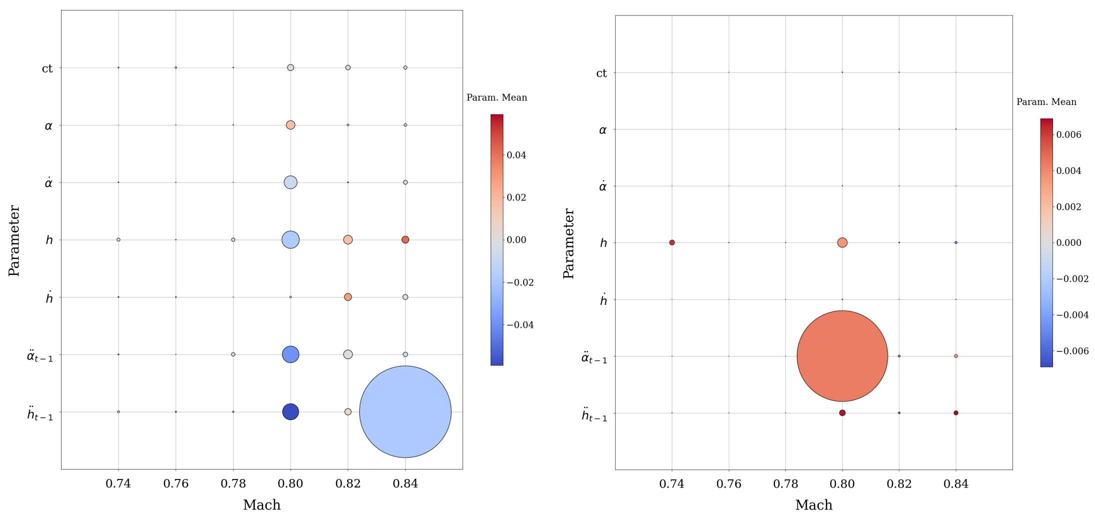
3.4.3 Uncertainty of ROM results to CFD-RANS Turbulence Model choice

This analysis evaluates how the uncertainty introduced by the CFD turbulence closure propagates through the ROM identification process and influences the predicted flutter behaviour. To isolate this effect from pure ROM parametrisation noise, the ROM architecture and training procedure are held fixed while retraining on datasets generated with four different turbulence models: Spalart–Allmaras (SA), Negative Spalart–Allmaras (SA–neg), Spalart–Allmaras with Edwards Modification (SA–E), and Menter Shear Stress Transport (SST).

The largest discrepancies between the turbulence models occur at higher Mach numbers. This is due to the fact that each turbulence model predicts a different shock wave position and thus a different moment coefficient, which leads to a variation in the flutter mechanism. In particular, the SST model predicts flutter to occur at higher dynamic pressures due to the different resolution of the equations at the wall, which leads to a rearward motion of the shock wave.

Figures 3.9–3.10–3.11 show the variability of the ROM parameters to the turbulence model choice for the three model architectures. The weights of the model are normalized to range $\in [-1, 1]$. Circle radii denote the standard deviation scaled by the respective mean. FIR models show larger variability at higher Mach numbers, especially at $M = 0.80$. In contrast, ARMA model experiences this variability at lower Mach numbers, or rather in the linear region. Globally, one can note a strictly relationship between parameters of the model and turbulence model choice, suggesting that the surrogate adapts its identified dynamics according to the closure-dependent wall-shear stress resolution.

Figures 3.12–3.13–3.14 report the uncertainty quantification of the q_{F} at the different Mach cases to the turbulence model used for the CFD-data generation to train the models. Flutter prediction uncertainty increases at higher Mach numbers for FIR models, especially at $M = 0.80$, even though quadratic FIR bandwidth partially includes CFD one. On the other hand, ARMA shows large variability in the extreme of the Mach range analysed, but, surprisingly, it is limited at $M = 0.80$, where the more complex phenomena occur, showing a robust behaviour of this model to changes in the wall shear stresses resolution.



(A) Parameter variability for C_L model.

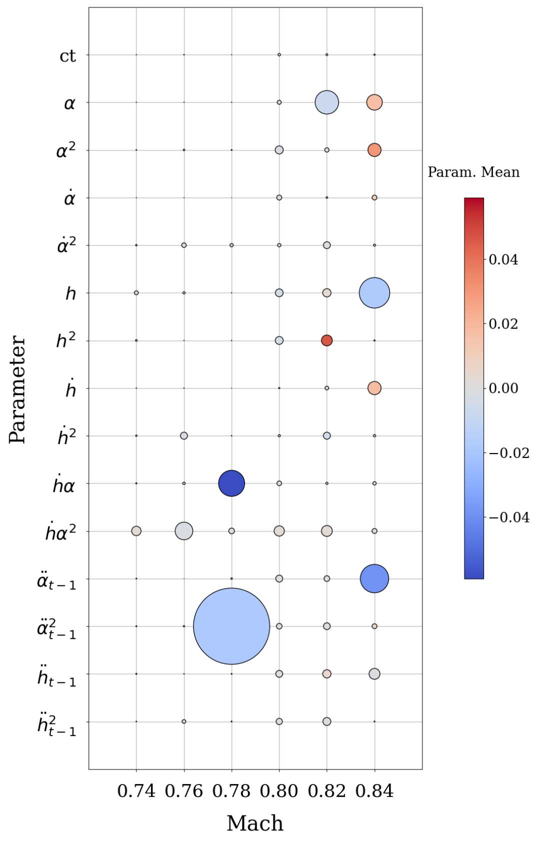
(B) Parameter variability for C_M model.

FIGURE 3.9: Linear FIR (model #1) parameter variability with the turbulence models. Model weights are in normalized form. Circle radii denote the standard deviation scaled by the respective mean.

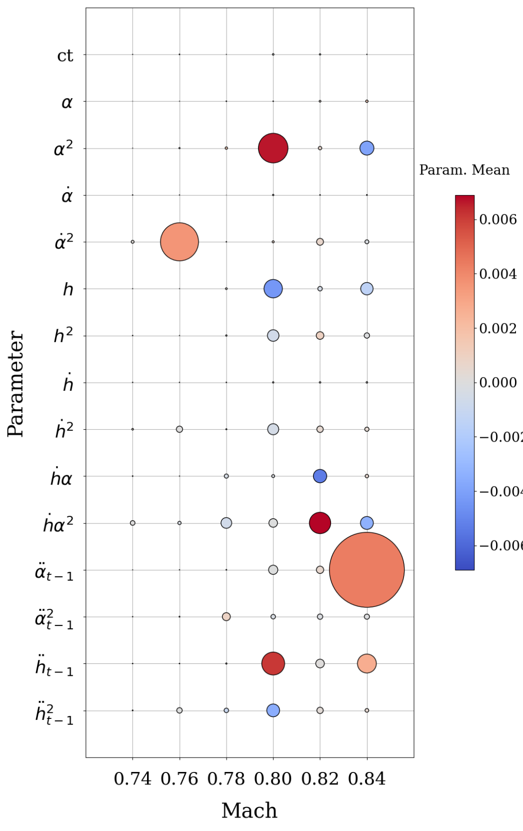
This analysis demonstrates three points. First, turbulence model choice is a leading-order source of uncertainty in the strongly nonlinear regime ($M \geq 0.80$), as closures predict different shock positions and thus different flutter mechanisms. Second, the sensitivity is architecture-dependent: FIR coefficients are most affected at $M = 0.80$, whereas ARMA parameters vary more at lower Mach but lead to stable q_F predictions at $M = 0.80$. Third, below $M = 0.80$ the effect of turbulence model choice remains limited for all ROMs.

3.4.4 Summary of Uncertainties Carried Forward

With the uncertainty quantification results collected in the current work, we can conclude that the confidence in the ROM modelling differs significantly depending on the side of the "switch" point of $M = 0.80$. At lower Mach numbers, the uncertainty in different modelling parameters was found to have a small influence on the flutter prediction results. This is consistent with the well known linear behaviour of aerodynamics. On the other hand, at higher Mach numbers, the uncertainty of the Mach number or the dynamic pressure used for the ROM training had a strong influence on the results. The complex nonlinear effects of large and intermittent boundary layer separation are likely to affect the confidence in the ROM predictability.



(A) Parameter variability for C_L model.



(B) Parameter variability for C_M model.

FIGURE 3.10: Quadratic FIR (model #2) parameter variability with the turbulence models. Model weights are in normalized form. Circle radii denote the standard deviation scaled by the respective mean.

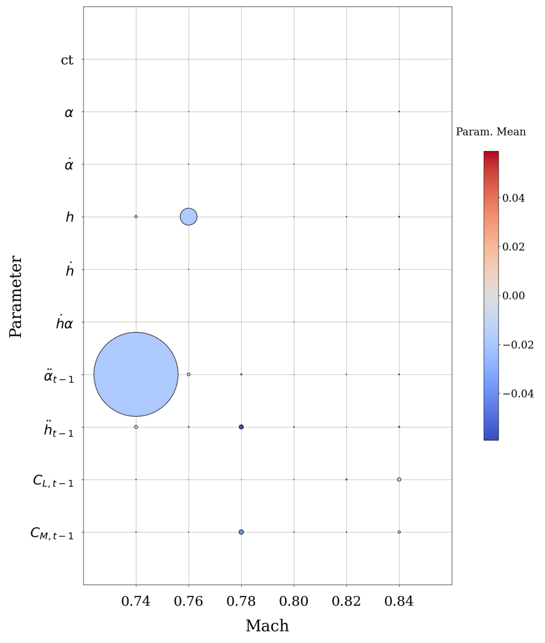
3.5 Computational Cost Analysis

A computational cost analysis has been performed in order to record the efficiency of the ML method against the high–order approach. The total computational cost required to determine the flutter boundary using the ROM-based approach is summarized in Table 3.5.

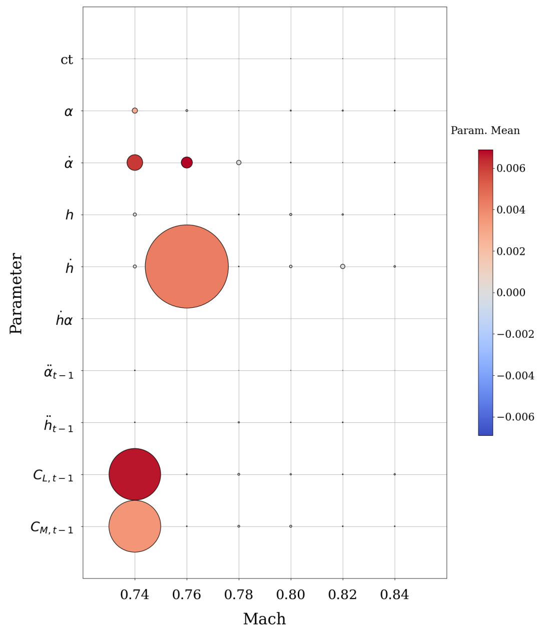
Task	**Computational Cost**
One URANS aeroelastic run	500 CPU hours
Aeroelastic simulations for training data generation	2 runs
URANS run for final verification of the flutter point	1 run
ML model training	0.03 GPU hours
Single ROM-based aeroelastic simulation	0.02 GPU hours
Iterations required per Mach number for identifying the flutter point	5
Number of Mach numbers analysed in the flutter boundary	6

TABLE 3.5: Computational cost breakdown for flutter boundary prediction.

Once trained, the ROM provides computational savings of over 99.9% compared to



(A) Parameter variability for C_L model.



(B) Parameter variability for C_M model.

FIGURE 3.11: ARMA (model #3) parameter variability with the turbulence models. Model weights are in normalized form. Circle radii denote the standard deviation scaled by the respective mean.

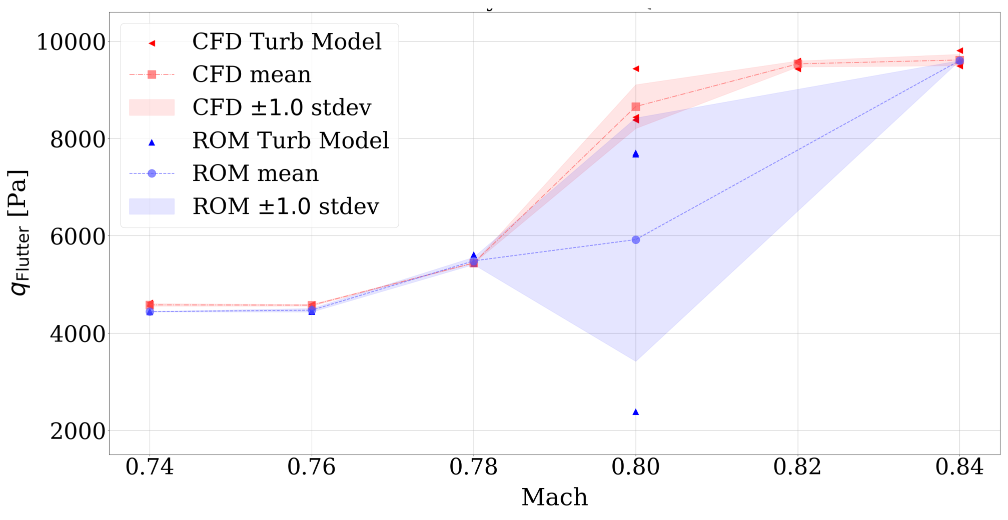


FIGURE 3.12: Uncertainty quantification of the q_F at the different Mach cases to the turbulence model used for the CFD-data generation to train the Linear FIR model.

direct high-fidelity simulations. The total cost of determining the 2D flutter boundary, including training data generation and validation, was approximately 12,600 CPU hours. By using the ROM, the total computational cost was reduced by up to 39.9%. These results highlight the importance of developing a model that effectively captures the key dynamic phenomena with as little training data as possible in order to limit the enormous preliminary cost.

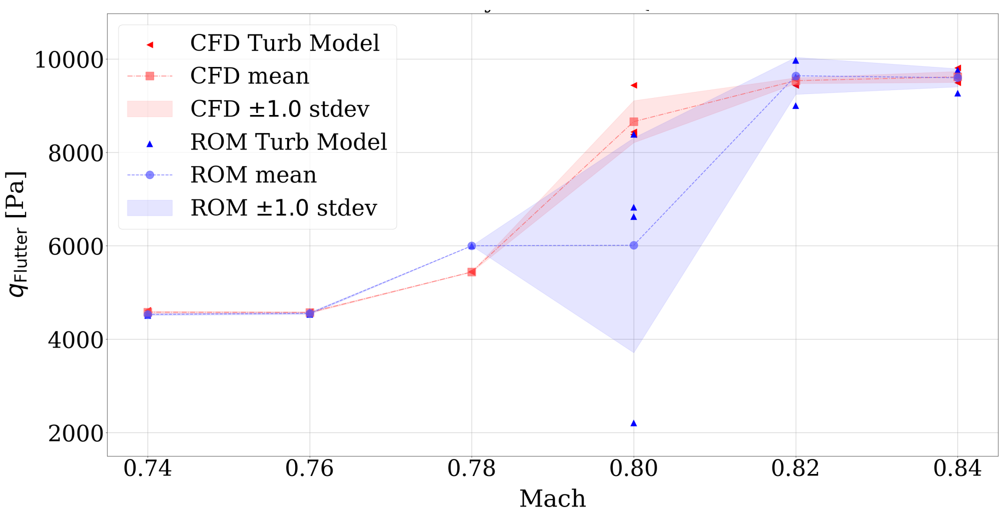


FIGURE 3.13: Uncertainty quantification of the q_F at the different Mach cases to the turbulence model used for the CFD-data generation to train the Quadratic FIR model.

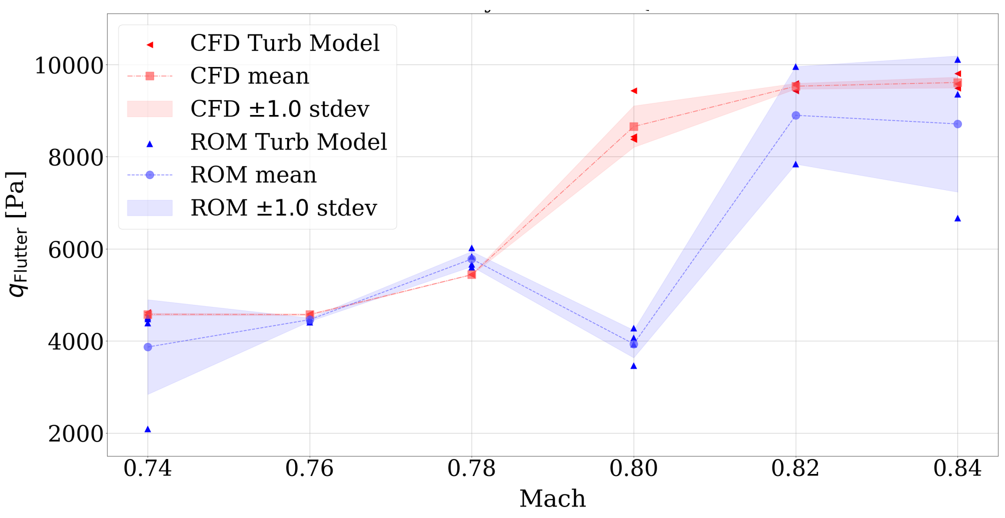


FIGURE 3.14: Uncertainty quantification of the q_F at the different Mach cases to the turbulence model used for the CFD-data generation to train the ARMA model.

3.6 Conclusions

This study explored the systematic development of an aerodynamic ROM leveraging ML for flutter boundary prediction of the NASA-SC(2)0414 supercritical airfoil in the transonic regime. The primary goal was to construct computationally efficient surrogate models while addressing key sources of epistemic uncertainty in transonic aeroelastic analysis. The study was conducted across a Mach number range of 0.74 to 0.84, yielding several key insights.

First, generating an aerodynamic ROM for aeroelastic analysis presents significant challenges. Small inaccuracies in aerodynamic load predictions propagate through the structural response, altering the input conditions for subsequent aerodynamic evaluations. This feedback loop can lead to rapid error accumulation during time integration, underscoring the necessity of developing well-validated models.

Three distinct ROM architectures were considered to account for varying levels of complexity in the flow physics, incorporating progressively higher-order terms in the input functions. These models demonstrated robust flutter boundary predictions with minimal training data, with the exception of higher Mach numbers, where accuracy deteriorated. A critical finding was the identification of a Mach 0.8 "switch" point, where shock-induced boundary layer separation becomes intermittent, driven by airfoil vibrations. This highly nonlinear behaviour was not adequately captured by the basis functions employed in the ROMs, leading to reduced model reliability in this regime.

To assess the robustness of the ROMs, sources of epistemic uncertainty were examined. Key questions addressed included: (i) How do variations in training data impact flutter boundary predictions? and (ii) How reliable are ROMs when applied to Mach numbers beyond their training conditions? The results indicate that uncertainty propagation remains limited across most Mach numbers but increases significantly near the Mach 0.8 switch point, where ROM coefficients exhibit erratic behaviour due to the complex aerodynamic-structural interactions.

The following observations underscore how the findings of this chapter advance the research goals:

1. **Efficient Nonlinear Aerodynamic modelling**
   By building data-driven models that capture transonic flutter boundaries with minimal training data, this chapter aligns with the thesis aim of creating computationally efficient surrogates for aerodynamic problems involving unsteady transonic loads. This directly contributes to Research Objective O1. The study demonstrates that small, well-chosen datasets can lead to effective ROMs, thereby reducing the reliance on full-scale CFD-CSD simulations.

2. **Robustness and Uncertainty Quantification**
   The chapter's systematic uncertainty analysis highlights how nonlinear flow phenomena degrade model predictions. By analysing the sensitivity to training set variations and identifying regions of poor extrapolation performance, this chapter advances Research Objective O2. The identification of a Mach 0.8 "switch" point clarifies the limitations of simple basis-function expansions and underscores the importance of employing multiple, or more advanced, ROM architectures to build confidence in the transonic regime.

3. **Limited but High-Value Training Data**
   Because each unsteady CFD run is expensive, having an approach that uses only a

few training signals is extremely important. The presented ROMs show that carefully chosen input motions can still reproduce flutter dynamics well, provided the underlying flow remains only weakly nonlinear.

4. **Extrapolation and Model Generalisation**
   Although the ROMs demonstrate satisfactory accuracy at lower Mach numbers, they struggle to extrapolate past the high transonic regime where shock-induced separation intensifies. This outcome motivates enhanced frameworks—e.g., higher-order or multi-fidelity training—to capture larger excursions in nonlinear aeroelastic behaviour.

5. **Foundation for 3D Extension**
   The 2D study in this chapter provides a valuable testing ground for how intermittent boundary layer separation couples to structural vibration. Identifying the need for more sophisticated approaches at Mach ≥ 0.80 paves the way for subsequent chapters where the complexity of 3D geometries and shock-boundary layer interactions can be more rigorously addressed.

These findings highlight the importance of utilizing multiple ROM architectures concurrently to improve confidence in predictions. Discrepancies between models can serve as an early indicator of inadequacies in the chosen basis functions, reinforcing the necessity of aligning ROM formulations with the underlying physics of the problem. This insight further leads to the fundamental question: how well does the CFD training dataset capture the correct aerodynamic physics for model generalisation?

Statement of Contributions

David Massegur contributed to the implementation of the Python scripts used for the presented ROMs.

Chapter 4

Parametric Nonlinear Volterra Series via Machine Learning for 3D Unsteady Transonic Loads

This study introduces an approach for modelling unsteady transonic aerodynamics within a parametric space, using Volterra series to capture aerodynamic responses and ML to enable interpolation. Addressing Research Objective O1, the chapter extends the modelling of unsteady aerodynamic loads to 3D configurations by incorporating nonlinear effects through higher-order Volterra kernels. The first- and second-order Volterra kernels are derived from indicial aerodynamic responses obtained through CFD, with the second-order kernel calculated as a correction to the dominant linear response. ML algorithms, specifically NN and GPR, are used to interpolate kernel coefficients within a parameter space defined by Mach number and angle of attack. Results underscore the benefit of including the second-order kernel to address strong nonlinearity and demonstrate the effectiveness of NNs. The approach achieves a level of accuracy that appears sufficient for use in preliminary design.

4.1 Introduction

In aerospace and mechanical engineering, product design often relies on hierarchies of mathematical models containing parameters for diverse operating conditions. Their complexity may be governed by regulations or constrained by computational cost. For example, optimisation tasks may involve hundreds of design variables [405, 186], and uncertainty propagation may require large parameter spaces that account for geometric imperfections, material properties, and flow conditions [403, 408]. Because the design process requires multiple evaluations across these spaces, lower-fidelity models (e.g., simplified physics-based solvers or coarse discretisations) are typically used for intensive analyses [238, 259]. Meanwhile, higher-fidelity models-often formulated as nonlinear partial differential equations on fine grids-are reserved for later stages [332]. In addition, the type of analysis significantly influences the computational requirements; for example, aeroelastic assessments may require either time-accurate simulations or high-resolution frequency-domain discretisations to capture transient interactions or flutter phenomena.

ROMs offer the possibility of retaining, in principle, the same accuracy provided by a full model at much lower computational costs. Their ability to capture nonlinear behaviour is often a key factor when choosing a model for a particular application [96, 320]. ROMs can also be parametric, with their development frequently leveraging coefficient interpolation within certain parameter ranges [6, 8, 10, 398]and where ensuring smoothness in the parameter space is considered an important prerequisite for efficient interpolation of coefficients [9]. Over the past few decades, a substantial body of literature has emerged on this topic [96, 134, 194, 219, 310, 317].

Among the many ROM approaches proposed, Volterra series holds significant prominence in several engineering fields [321]. Volterra series [361] are the multi–dimensional extension of the impulse response concept from linear system theory, allowing the use of Volterra kernels for the characterisation of nonlinear systems. The method consists of a series of kernels that capture the various nonlinear interactions between the system's inputs and outputs. Each kernel in the series corresponds to a specific degree of nonlinearity, progressively representing higher-order effects. One of the main challenges in applying the Volterra series lies in the precise identification of the kernel coefficients in particular for higher order kernels, which can require a large amount of data and significant computational resources [94, 185].

Various strategies have been introduced to address these challenges. Marzocca et al. [227] employed Volterra series from aerodynamic indicial functions to investigate nonlinear aeroelasticity, while Loghmanian et al. [218] used a genetic algorithm to optimize discrete-time Volterra kernels. Da Silva et al. [319] introduced orthonormal Kautz functions for identifying nonlinearities, and Nie and Wu [211] proposed Matched Volterra ROM to improve transonic load prediction, especially for large-amplitude oscillations. Israelsen and Smith [166] further streamlined kernel identification by projecting kernels onto Laguerre polynomials. ML-based approaches offer new ways to increase efficiency [89, 90, 292, 379]. Levin et al. [201] recently proposed a two-step method that identifies linear kernels from small-amplitude inputs and treats higher-order kernels as corrections for larger amplitudes, extending Balajewicz and Dowell's Sparse Volterra Series framework [19], which captures nonlinearities and memory effects at reduced computational cost.

In this study, we focus on the identification of Volterra kernels at selected points within a parameter space and on the subsequent reconstruction or interpolation of the kernel coefficients across a larger set of points. Rather than directly predicting long unsteady responses comprising thousands of timesteps, as done by sequence models, such as classical ML surrogates (RNN/GRU/LSTM), we compress the output space: the unsteady input-output mapping is represented by a small set of linear and nonlinear Volterra kernels with finite memory. ML is then tasked with learning smooth maps from the parameter vector to the kernel coefficients. This factorisation (identification $\rightarrow$ interpolation) reduces the effective dimensionality, mitigates sequential error accumulation, and better exploits data efficiency: once kernels are known, responses to arbitrary inputs are reconstructed by fast convolutions, eliminating the need for autoregressive rollout of full signals. While the parameter space in our examples is defined by M and α_0, the methodology is versatile and can be applied to deliver the unsteady aerodynamic response for any combination of parameters, making it well-suited for optimisation or flutter tracking along specific parameter paths, such as constant M or constant α_0. To determine the Volterra kernel coefficients, we adopt the approach of Levin et al. [201], which separates the identification of linear and nonlinear components of the system's response; this proves pragmatic and robust, as the linear kernels, which typically dominate the system, are uniquely determined. For interpolation, we explore two machine-learning techniques, GPR and FCNN, with the latter demonstrating superior accuracy in nearly all experiments. The effectiveness of the Volterra ROM is demonstrated through three test cases: a synthetic aerodynamic model, a 2D airfoil, and a 3D wing, focusing on transonic conditions.

4.2 Methodology

The methodology is designed around Volterra series for a single-input single-output (SISO) system. First, we outline the approach for identifying the linear kernel independently from the higher-order kernels. Next, we introduce two ML techniques to construct a parametric Volterra model for a SISO system, with dependencies on α_0 and M. Although this study focuses on SISO systems, the proposed approach and methods are extendable to multi-input systems. It is worth noting that kernel identification in such cases is discussed less extensively in the literature, as pointed out by Worden [378].

4.2.1 Volterra series for a SISO system

Given a discrete-time, time-invariant and nonlinear system with single-input u, the output $y(n)$ can be approximated be means of the p-th order Volterra series:

$$y(n) = \sum_{i=1}^{p} H_i[u(n)], \tag{4.1}$$

where the Volterra series is composed by p operators, H_i, each with a memory depth m, i.e. only the past m samples of the input influence $y(n)$. Physically, m sets the time horizon over which the system retains memory of previous excitations; in practice, it is chosen large enough for the aerodynamic transients (e.g. indicial responses) to have effectively decayed.

The first operator is as follows:

$$H_1[u(n)] = \sum_{k=1}^{m} h_1(k)\, u(n-k), \tag{4.2}$$

and corresponds to a linear convolution operator, which can be identified through indicial responses[286]. In this work, we refer to $H()$ as the kernel and $h()$ as the kernel coefficients. This terminology, while consistent throughout this work, may differ from conventions used in other textbooks. The methodology presented here focuses on the reconstruction of these kernel coefficients. The following two operators are of the form:

$$H_2[u(n)] = \sum_{k_1=1}^{m} \sum_{k_2=1}^{m} h_2(k_1,k_2)\, u(n-k_1)\, u(n-k_2), \tag{4.3}$$

$$H_3[u(n)] = \sum_{k_1=1}^{m} \sum_{k_2=1}^{m} \sum_{k_3=1}^{m} h_3(k_1,k_2,k_3)\, u(n-k_1)\, u(n-k_2)\, u(n-k_3), \tag{4.4}$$

In general, the i-th operator, H_i, represents the i-th order discrete-time convolution between the input u and the Volterra kernels h_i:

$$H_i[u(n)] = \sum_{k_1=1}^{m} \cdots \sum_{k_i=1}^{m} h_i(k_1,\ldots,k_i,\ldots) \prod_{z=1}^{i} u(n-k_z), \tag{4.5}$$

where, in most cases, the kernels are symmetrical with respect to any permutation of the indices, i.e. $h(k_1,k_2) = h(k_2,k_1)$. For a system with multiple inputs, u_i, the Volterra operators are extended to take into account all possible combinations of the inputs, see for example [19, 21].

It is well known that increasing the order of the Volterra series introduces a large number of kernels, the identification of which may become demanding or prohibitive if no simplifications are introduced as pointed out by Levin et al. in Ref. [201]. In practice, a well-known assumption consists in removing cross-memory effects, $h(k_i, k_j) = 0$ for $i \neq j$, as presented by Balajewicz and Dowell [19] for applications similar to those presented herein. It is worth noting that data regularisation is also exploited to reduced the number of coefficients as mentioned by Cheng et al. [64]. This means, in practice, that kernels of a given operator are expanded in terms of a set of basis functions, typically Laguerre polynomials (refer to [52] for a thorough presentation). For instance, the first order kernel for a SISO system is represented by a series expansion with the first R Laguerre polynomials:

$$h_1(j) = \sum_{r=1}^{R} \theta_r \, L_r(t_j), \tag{4.6}$$

which means that the j-th kernel is obtained from the combination of R Laguerre polynomials evaluated at the time t_j. The first order output can therefore be expressed as:

$$y(t) = \sum_{k=1}^{m} \sum_{r=1}^{R} \theta_r \, L_r(t_k) \, u(n-k), \tag{4.7}$$

The order of the Laguerre polynomials R is set on a case by case basis. In the present study, a number of kernels up to several hundreds was used, whereas only ten to twenty Laguerre polynomials were sufficient to obtain a level of accuracy deemed acceptable. Moreover, the regularisation provided by the Laguerre polynomials also provides additional robustness in the identification process. In the context of Laguerre polynomials, it is necessary to point out that a series expansion is in principle incompatible with a parametric realisation of the Volterra series, which means that the kernels are reconstructed over a parameter space (the expression "interpolation" may be used to be consistent with the literature, even though different techniques are exploited in this study). As mentioned by Amsallem and Farhat [8], a set of interpolated Laguerre coefficients θ_r, may not provide a consistent expansion of the interpolated kernels. As such, this study exploits Laguerre polynomials primarily for data regularisation but not for reconstructing the kernels over a parameters space.

4.2.1.1 Identification of Linear Volterra kernel for a SISO System

Herein, a step function or a smoothed step function were used for all test cases. When using the linear Volterra kernels only, the approximated response is equivalent to a conventional convolutional integral in discrete time:

$$y(n) = \sum_{k=1}^{m} h_1(k) \, u(n-k), \tag{4.8}$$

Eq. 4.8 can be written in matrix form for n values of the response:

$$\{y\} = [U]\{h\}, \tag{4.9}$$

where U is $n \times m$ matrix (a lower triangular square matrix if $n = m$) containing the input values re-arranged so that the input signal starting from the row corresponding to the number of the column:

$$U(i,j) = u(i-j), \tag{4.10}$$

also expressed in matrix form:

$$[U] = \begin{bmatrix} u(0) & & & & \\ u(1) & u(0) & & & \\ u(2) & u(1) & u(0) & & \\ \vdots & \vdots & \vdots & & \\ u(m) & u(m-1) & u(m-2) & \dots & u(0) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ u(n) & u(n-1) & u(n-2) & \dots & u(n-m) \end{bmatrix}. \tag{4.11}$$

Note that in case a step input signal of magnitude α is used, $u(i \geq 0) = \alpha$ and U has the form:

$$[U] = \alpha \begin{bmatrix} 1 & 0 & 0 & \dots & 0 \\ 1 & 1 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & 1 & 1 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix}. \tag{4.12}$$

The linear kernels are then obtained by inverting Eq. 4.9:

$$\{h\} = [U]^{+}\{y\}. \tag{4.13}$$

where $[U]^{+}$ denotes the pseudo-inverse of $[U]$ if $n > m$ and the inverse of $[U]$ if $n = m$. The pseudo-inverse of a matrix $[A]$ (shape $n \times m$) is obtained from Singular Value Decomposition (SVD):

$$[A] = [U]\,[\Sigma]\,[V]^{T}, \tag{4.14}$$

where U (shape $n \times r$) and V (shape $r \times m$) are orthonormal matrices and Σ a diagonal matrix containing the r singular values of A. Note that generally all references use the symbols U and V to present SVD, which are also used in this work for consistency. The Volterra coefficients are computed from the entire step response, with the memory depth m corresponding to the number of time steps (typically ranging from 100 to 500, depending on the chosen CFD time step size). In this study, no truncation of the Volterra series was performed. This choice may, however, be confusing as the symbol U is also used for the matrices in which the input signals are recast (Equations 4.11, 4.12) but we do so because most of the references on the topic use the same symbol.

In the case Volterra kernels are projected onto the subspace defined by l Laguerre polynomials:

$$\{h\} = [B]\,\{\theta\} \tag{4.15}$$

where $[B]$ is a $m \times l$ matrix. Eq. 4.9 is re-written as:

$$\{y\} = [U]\,[B]\,\{\theta\}, \tag{4.16}$$

and also solved using SVD to obtain the pseudo-inverse $(([U]\,[B])^{+})$ of the $n \times l$ matrix $[U]\,[B]$:

$$\{\theta\} = ([U]\,[B])^{+}\,\{y\}, \tag{4.17}$$

For all test cases, approximately 20 Laguerre polynomials were used. This number was determined through an optimisation process aimed at minimizing the approximation error.

4.2.1.2 Identification of Higher-order Volterra kernels for a SISO System

Following Ref. [201], a two-step approach is exploited. It is not applicable in general and only justified if the linear part of the response dominates over the nonlinear contribution.

The approach exploits a number of input signals with identical shape but increasing magnitude (u_A, u_B, …, u_N) and the corresponding CFD responses, y_A, y_B, …, y_N. Attention must be taken to choose the magnitude of u_A such that the linear response alone provides the necessary accuracy. Linear kernels may then be identified using Eq. 4.13 from the input signal of magnitude α_A:

$$\{h_1\} = [U_A]^{-1}\{y_A\}, \tag{4.18}$$

where the matrix $[U_A]$ corresponds to $[U]$ in Eq. 4.11. To a minimum, one requires at least one response with magnitude $\alpha_B > \alpha_A$. Then, one can assume that y_B can be approximated with a second order Volterra series:

$$\{y_B\} = [U_B]\{h_1\} + [U_{B2}]\{h_2\}, \tag{4.19}$$

where $[U_B]$ is build like the matrix $[U]$ in Eq. 4.11 and $[U_{B2}]$ contains the square of the terms of the input signal, in the same form of $[U_B]$:

$$[U_{B2}] = \begin{bmatrix} u(0)^2 & & & & \\ u(1)^2 & u(0) & & & \\ u(2)^2 & u(1)^2 & u(0)^2, & & \\ \dots & & & & \\ u(m)^2 & u(m-1)^2 & u(m-2)^2 & \dots & u(0)^2 \\ \dots & & & & \\ u(n)^2 & u(n-1)^2 & u(n-2)^2 & \dots & u(n-m)^2 \end{bmatrix}, \tag{4.20}$$

In particular, when the input signal is a step function, $[U_{B2}]$ reads:

$$[U_{B2}] = \alpha_B^2 \begin{bmatrix} 1 & 0 & 0 & \dots & 0 \\ 1 & 1 & 0 & \dots & 0 \\ \dots & & & & \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix}. \tag{4.21}$$

Eq. 4.19 shows that the second order kernels, h_2, can be determined to maximise the accuracy of y_B reconstruction:

$$[U_{B2}]\{h_2\} = \{y_B\} - [U_B]\{h_1\}, \tag{4.22}$$

leading to the solution for h_2:

$$\{h_2\} = [U_{B2}]^+ (\{y_B\} - [U_B]\{h_1\}). \tag{4.23}$$

If more than two signals are available, higher order kernels can be identified. For instance, if two signals U_B and U_C and their responses are known, a third order series:

$$\{y_B\} = [U]\{h_1\} + [U_2]\{h_2\} + [U_3]\{h_3\}, \tag{4.24}$$

where $[U_3]$ is analogous to $[U_2]$ but with the input terms raised to the third power. For the signals B and C:

$$\begin{aligned} \{y_B\} &= [U_B]\{h_1\} + [U_{B2}]\{h_2\} + [U_{B3}]\{h_3\} \\ \{y_C\} &= [U_C]\{h_1\} + [U_{C2}]\{h_2\} + [U_{C3}]\{h_3\}, \end{aligned} \tag{4.25}$$

h_2 and h_3 can be identified from Eq. 4.25:

$$\begin{aligned} [U_{B2}]\{h_2\} + [U_{B3}]\{h_3\} &= \{y_B\} - [U_B]\{h_1\} \\ [U_{C2}]\{h_2\} + [U_{C3}]\{h_3\} &= \{y_C\} - [U_C]\{h_1\}, \end{aligned} \tag{4.26}$$

simultaneously solving for h_2 and h_3:

$$\left\{ \begin{array}{c} \{h_2\} \\ \{h_3\} \end{array} \right\} = \left[\begin{array}{cc} U_{B2} & U_{B3} \\ U_{C2} & U_{C3} \end{array} \right]^+ \left\{ \begin{array}{c} \{y_B\} - [U_B]\{h_1\} \\ \{y_C\} - [U_C]\{h_1\} \end{array} \right\}. \tag{4.27}$$

As pointed out in [201], Eq. 4.27 is not the only possible strategy. Results here were obtained with second order kernels only, in order to keep within limits the computational demand for the generation of responses. However, different tests with higher-order series were carried out for few parameter samples, showing the effectiveness Eq. 4.27 and its superiority with respect to a multi-step approach (h_2 from response y_B, then h_3 from response U_c etc).

4.2.2 Machine Learning Models for Volterra Kernels Interpolation

Two ML models are investigated to interpolate the Volterra kernels defined in Equations 4.2–4.3 over a parameter space of M and α_0. These kernels represent the linear and nonlinear components of the Volterra series (Equation 4.1), respectively, and require interpolation to capture the unsteady aerodynamic behaviour under varying transonic conditions. The ML-predicted kernels are then used in the convolution operation (Equation 4.19) to reconstruct the system response for arbitrary inputs.

To account for the inherently different dynamics of first- and higher-order kernels, two ML models are trained separately: one for the linear kernels and another for the nonlinear kernels. Each model takes as input M, α_0, and the static lift and pitching moment coefficients (C_{L0}, C_{M0}). The first model uses a FCNN architecture. Its inputs pass through multiple hidden layers to output the Volterra kernel coefficients, and training is performed by minimizing the MSE between the FCNN predictions and high-fidelity reference data for the kernels. The Adam optimizer performs backpropagation. Since the performance of neural networks is highly sensitive to the choice of hyperparameters, Bayesian optimisation is employed as an efficient strategy to navigate the large, nonconvex search space and identify well-performing configurations (refer to

Section 2.5.7). This strategy systematically explores the hyperparameter space of learning rate, network depth, number of neurons, and batch size, thus avoiding extensive manual tuning. Table 4.1 lists the hyperparameter design space used in Bayesian optimisation. The final configuration of the FCNN models for each test case is reported in Appendix A.2.

Hyperparameter	**Value**	**Step size**
Learning rate	10^{-4} to 10^{-6}	$5 \cdot 10^{-6}$
Number of Hidden Layers	1 to 8	1
Number of Neurons per Hidden Layer	4 to 204	8
Batch size	1 to 8	1
Activation function	TanH, ReLu, PReLu	/

TABLE 4.1: Hyperparameter design space for the Bayesian optimisation of the FCNN models.

The GPR model is also tested. Unlike the deterministic FCNN, GPR provides a Bayesian framework that models the function space as a Gaussian process, fully defined by a mean function and a covariance function. As in the FCNN approach, two GPR models are built for the linear and nonlinear kernels. By choosing appropriate covariance functions, it is possible to capture the kernel dependence on the design space. In this work, summing the contributions of an RBF and a Mateŕn 5/2 function provided the most accurate results.

4.3 Synthetic Aerodynamics

To demonstrate the validity and feasibility of the concept, we utilize a system of differential equations specifically designed to replicate the indicial responses of real wings and airfoils. This approach allows for the efficient generation of data while providing control over the system's nonlinearity and noise levels. The differential equations mimic the state-space formulation proposed by Leishman [198]:

$$\begin{aligned} \frac{dx_1}{d\tau} &= -\frac{2V}{c} b_1 x_1 \\ \frac{dx_2}{d\tau} &= -\frac{2V}{c} b_2 x_2 \\ \frac{dx_3}{d\tau} &= -\frac{2V}{c} b_3 x_3, \end{aligned} \tag{4.28}$$

where x_1, x_2, x_3 play the role of the aerodynamic lag states and τ is the reduced time. We introduce a first-order, linear, expression of the lift coefficient:

$$C_L^l(\tau) = C_{L/\alpha} \left(A_1 b_1 x_1(\tau) + A_2 b_2 x_2(\tau) + A_3 b_3 x_3(\tau) \right), \tag{4.29}$$

where the coefficients $C_{L/\alpha} = 2\pi$, $A_1 = 0.670$, $A_2 = 0.330$, $b_1 = 0.30$, and $b_2 = 0.0455$ are modified from Jones' approximation of the Wagner function. We introduced a dependence of the system coefficients on two parameters: M and α_0. The lift curve slope is influenced by M through the Prandtl-Glauert factor, while the coefficient A_3 depends nonlinearly on both M and α_0, ranging from 0 to -0.15. The value $b_3 = 0.15$ was selected to approximate a dynamic stall response.

We also introduce a second, nonlinear expression of the lift coefficient, C_L^{nl}:

$$C_L^{nl}(\tau) = C_{l/\alpha}\left(A_1 b_1 x_1(\tau) + A_2 b_2 x_2(\tau) + A_3 b_3 x_3(\tau) - C_{nl} x_2(\tau) x_3(\tau)\right), \tag{4.30}$$

The nonlinearity term $C_{nl} = -0.35$ was selected to mimic the qualitative behaviour observed in transonic airfoil responses obtained from CFD simulations. To account for variability, white noise was introduced to all signals. A summary of our analysis is provided in Algorithm 2.

Algorithm 2 Procedure for reconstructing the system response using Volterra series and ML

```
1: Generate $n$ samples within the parametric space defined by $M$ and $\alpha_0$.
2: Calculate the response for each sample to a step input signal with an amplitude of
   one degree using Equations 4.28 and 4.29.
3: Compute the linear Volterra kernels based on the responses.
4: Calculate additional responses to a step input signal with an amplitude of two de-
   grees using Equations 4.28 and 4.30 to account for nonlinear effects.
5: Compute the nonlinear Volterra kernels based on the additional responses.
6: Split the $n$ samples into three subsets: training ($n_1$), test ($n_2$), and validation ($n_3$).
7: Use the training and test subsets ($n_1$ and $n_2$ samples) to train the GPR and FCNN
   models.
8: Predict the Volterra kernels for the validation subset ($n_3$ samples) using the trained
   GPR and FCNN models.
9: Reconstruct the system response to a sinusoidal input in the validation subset using
   the Volterra kernels predicted by the trained GPR and FCNN models. Compare
   the reconstructed responses with the exact solutions derived from the differential
   equations.
```

We defined the parameter space for M ranging from 0.40 to 0.85 and α_0 between -2 and 8 [deg]. Using Latin hypercube sampling (LHS), we generated 70 data points for both variables. These were divided into $n_1 = 45$ for training, $n_2 = 15$ for test, and $n_3 = 10$ for validation.

4.4 2D Test Case

The second test case is for a 2D airfoil section based on the NACA0012 profile. The choice was motivated by the presence of highly nonlinear transonic phenomena. In this

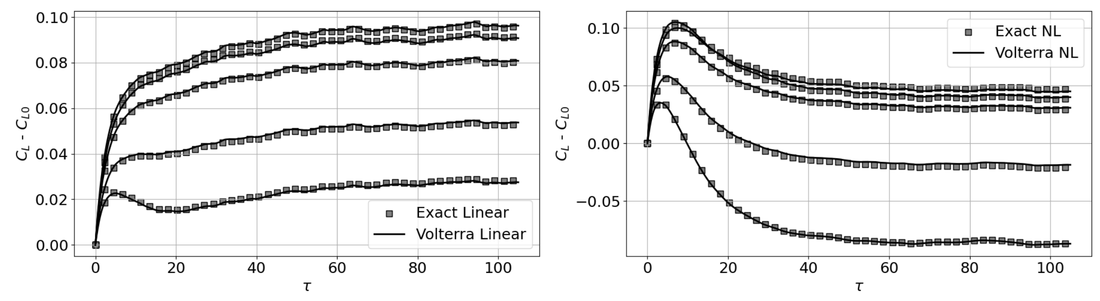


FIGURE 4.1: Linear (left) and nonlinear (right) synthetic responses under a step input. Squares indicate exact responses; continuous lines denote Volterra series reconstructions.

Model Name	ε_{C_L}	
	Linear	Nonlinear
FCNN	0.0032	0.0070
GP	0.0035	0.0078

TABLE 4.2: MAE for the linear and nonlinear sinusoidal responses of the two ML models in the synthetic aerodynamics test case.

regime, a supersonic pocket forms over the suction side and is terminated by a shock, producing an abrupt pressure recovery. For symmetric profiles, such the NACA0012 in our case, a weaker supersonic region and shock may also appear on the pressure side, especially at higher M or small negative incidences, so that two shocks can co-exist on opposite surfaces. The chordwise position and strength of these shocks vary sharply with (M, α_0), leading to large pressure gradients and a pronounced transonic drag rise. The adverse pressure gradient across the shock thickens the boundary layer and can trigger a separation bubble downstream of the shock foot. Depending on boundary layer state (laminar, transitional, or turbulent), the separated region may reattach within a short distance or persist towards the trailing edge. The resulting pattern, shock, separation bubble, and possible reattachment, creates a strongly nonlinear response to small perturbations in incidence, Mach number, or forcing amplitude.

CFD indicial step–responses were generated through the Unsteady–RANS (URANS) formulation, employing SU2 v7.5.0, a cell-centred finite volume software suite [104], with the one–equation Spalart-Allmaras model; the negative turbulence production option was always active. To accelerate the convergence of CFD simulations, we adopted a $1v$ multigrid scheme. For discretizing convective flows, we used the JST central scheme with artificial dissipation, while flow variable gradients were calculated using the Green Gauss method. Linear solving was accomplished with the biconjugate gradient stabilisation method, complemented by ILU preconditioning. URANS simulations were restarted from the steady–state solution. The nondimensional timestep employed was $\Delta\tau = 0.15$, and the total nondimensional time of the simulations was set to $\tau = 113.6$, allowing for the complete development of the flow during the step motion.

A structured O–mesh with 177,112 elements was generated. To maintain a consistent boundary layer resolution and adequately capture shock waves, we adopted a $y^+ = 1$. The computational domain spans 100 chord lengths from the solid wall to the farfield boundary. For a visual representation of the mesh, refer to Figure 4.2.

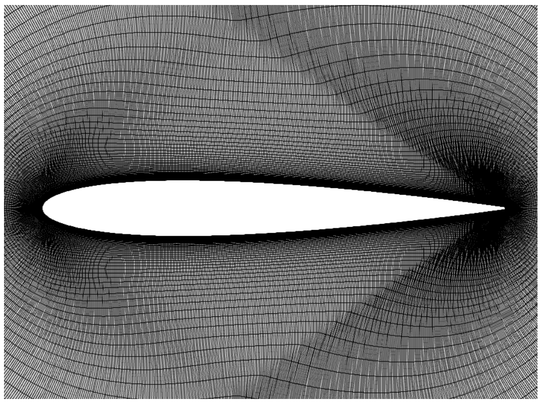


FIGURE 4.2: Impression of the NACA0012 CFD grid.

We defined a parameter space for M (0.58 to 0.76) and α_0 (0 to 7 [deg]) to capture the physics associated with the shock-boundary layer interaction. Using LHS, we generated 70 data points for both variables (see Figure 4.3). The data was split into 80% for training and 20% for test to evaluate the model performance. The excitation signal for each of the 70 step responses consists of an indicial plunge response generated with vertical velocities of $-1\,m/s$ and $-2\,m/s$, enabling the identification of both linear and nonlinear dynamics. The Volterra series was then employed to reconstruct the dynamic response to sinusoidal input signals at specific reduced frequencies. The time step of the CFD simulations and the memory depth m were chosen to ensure a sufficient resolution of the reduced frequency of the input signal and hence guarantee the physical consistency of the reconstruction. Red markers in Figure 4.3 denote flight conditions of CT2 and CT5 experiments [192] with the bands indicating their variations in pitch during the harmonic forced motions. These two signals are used as validation points, where the prediction involves harmonic motion, with both mean and amplitude being critical variables. We may claim that this approach tests the model predictive capability beyond the bounds of the training region.

Similar to the previous test case, the initial step involved identifying both the linear and nonlinear Volterra kernels for the training and test samples within the $M - \alpha_0$ design space. We then employed the GPR and FCNN algorithms to predict the kernel coefficients at the points in the validation set, and to reconstruct the harmonic signal associated with the AGARD CT2 and CT5 cases [192].

For the CT2 case, the parameters are $M = 0.60$, reduced frequency $k = 0.0811$, mean angle of attack $\bar{\alpha} = 3.16$ [deg], and amplitude of pitch oscillation $\alpha_A = 4.59$ [deg]. Figure 4.4a shows the results for this case. The hysteresis loops for lift coefficient C_L and pitching moment coefficient C_M are presented, comparing the experimental data to

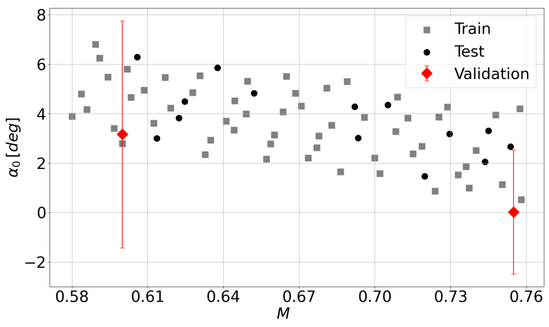


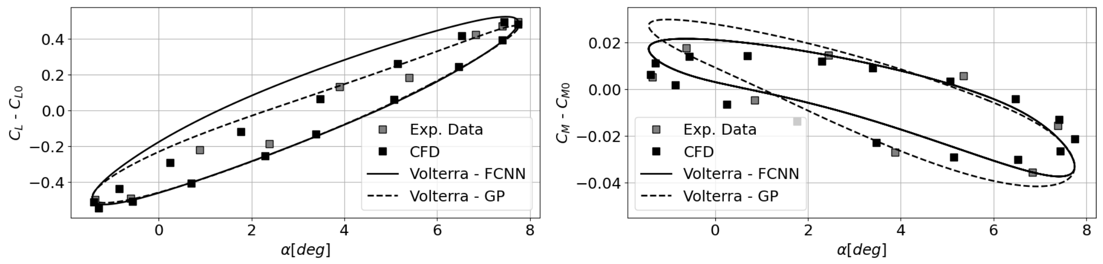
FIGURE 4.3: Samples for M and α_0 for NACA0012 test case. Red markers denote flight conditions of CT2 and CT2 experiments [192].

the nonlinear Volterra reconstructions using FCNN (continuous line) and GPR (dashed line). At this reduced frequency, the loop area primarily reflects aerodynamic damping and the phase lag of the circulatory component relative to the motion. Both ML models provide a good reconstruction of the C_L hysteresis. However, for the C_M hysteresis, a slight discrepancy is observed, likely due to the significant pitch variation, which challenges the accuracy of the Volterra series. Despite this, FCNN captures the dynamic behaviour of C_M more effectively than GPR.

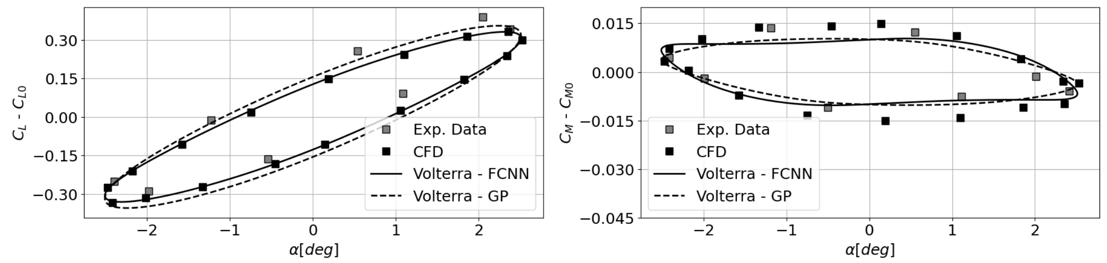
For the CT5 case, the parameters are $M = 0.755$, reduced frequency $k = 0.0814$, mean angle of attack $\bar{\alpha} = 0.016$ [deg], and amplitude of pitch oscillation $\alpha_A = 2.51$ [deg]. Here, compressibility strengthens the shock system even at small angles. Figure 4.4b shows the corresponding hysteresis loops for C_L and C_M. In this case, FCNN outperforms GPR for both aerodynamic coefficients. The C_L hysteresis is well-captured by both models, while C_M hysteresis shows some deviations, with FCNN providing a more accurate fit.

These results are further reinforced by Table 4.3, which provides the `MAE` values for C_L and C_M on the validation set, quantifying the average absolute difference between the reconstructed aerodynamic coefficients with the Volterra ROM and their reference CFD counterparts over the two validation cases. FCNN achieves approximately half the error of GPR for C_L and a significantly lower error for C_M. These highlight the FCNN capability to better handle the nonlinear complexities associated with the aerodynamic responses under study.

Overall, the nonlinear Volterra reconstruction of the aerodynamic coefficients was reasonably good in both cases, particularly for C_L hysteresis, aligning with results from Yao et al. [391]. However, minor discrepancies were found in C_M, especially in the CT2 case, likely due to the wide pitch variation during the oscillation and higher mean incidence. The significant variations in the kernels coefficients between training points make this a highly challenging test case for the proposed methodology. Exploring the use of higher–order kernels could potentially mitigate this issue, although it would dramatically increase the computational demands.



(A) CT2: $M = 0.60 - k = 0.0811 - \bar{\alpha} = 3.16$ [deg] – $\alpha_A = 4.59$ [deg]



(B) CT5: $M = 0.755 - k = 0.0814 - \bar{\alpha} = 0.016$ [deg] – $\alpha_A = 2.51$ [deg]

FIGURE 4.4: Harmonic signal reconstruction with Volterra framework for the AGARD CT2 and CT5 problems [192].

Model Name	ε_{C_L}	ε_{C_M}
FCNN	0.0075	0.0065
GP	0.0168	0.0096

TABLE 4.3: MAE on the validation set for the two ML models in the 2D test case.

4.5 3D Test Case

To further increase the complexity of the model geometry and flow features, the third test case is the Benchmark Super Critical Wing (BSCW). It is a rigid semi–span wing with a rectangular planform and a supercritical airfoil shape, the NASA SC(2)-0414 profile, from the AIAA Aeroelastic Prediction Workshop [1]. The wing is mounted on a flexible system with two degrees of freedom, allowing movement in both pitch and plunge. However, in this study, we focus exclusively on pitch responses. The BSCW has been specifically designed for flutter analysis and exhibits shock wave motion, shock-induced boundary layer separation, and interactions between the shock wave and detached boundary layer. In addition to shock dynamics, the BSCW highlights strong flow-structure coupling. Unsteady aerodynamic loads directly influence the structural pitch motion, while the phase relationship between lift and moment coefficients governs the effective aeroelastic damping. At low incidence, when the flow remains attached or only mildly separated, C_L and C_M tend to remain nearly in phase. By contrast, at higher angles of attack dynamic stall phenomena emerge, giving rise to pronounced phase lags and hysteresis loops, particularly in the moment coefficient.

[1]https://nescacademy.nasa.gov/workshops/AePW3/public/wg/highangle

These effects can alter the onset of flutter. The severity of this behaviour increases with Mach and Reynolds numbers: Mach governs shock strength and position, whereas Reynolds number controls boundary layer state and the extent of shock–induced separation. Their interaction produces amplitude–dependent restoring and damping forces, which are difficult to capture with conventional ROM formulations.

The experimental reference data for this configuration were obtained in the NASA Langley Transonic Dynamics Tunnel (TDT), a variable–pressure facility capable of operating with a heavy gas medium (R-134a). The use of heavy gas allows transonic Mach numbers to be reached at lower tunnel speeds and dynamic pressures by increasing density and reducing the speed of sound. This, however, also implies that standard perfect gas assumptions ($\gamma = 1.4$, constant C_P) are not strictly valid. Accurate reproduction of the experiments therefore requires accounting for real gas properties.

4.5.1 Reference Dataset

Also in this test case, we generated CFD indicial step–responses using URANS formulation with SU2 v7.5.0. The nondimensional time step employed was $\Delta\tau = 0.029$. The total nondimensional time of the simulations was set to $\tau = 27.2$ to ensure full development of the flow. A mixed-type grid with $15.6 \cdot 10^6$ elements and 130,816 surface elements was generated, structured on the wing surface and in the first layers of the boundary layer, voxel in the rest of the computational domain. A $y^+ = 1$ is adopted, after a preliminary mesh convergence study that ensured an adequate resolution of the boundary layer and shock wave. The computational domain extends 100 chords from the solid wall to the farfield. Aerodynamic coefficients were calculated using a reference chord length of 0.4064 [m] and surface of 0.3303 [m^2]. An impression of the grid can be obtained from Figure 4.5.

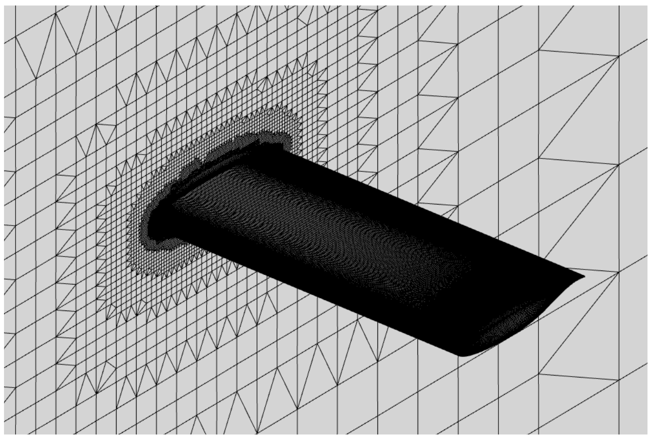


FIGURE 4.5: Impression of the BSCW structured grid.

Following the same approach as in the other cases, the two independent parameters for the models are α_0 and M with ranges of $[0, 5]$ [deg] and $[0.70, 0.84]$, respectively.

These ranges were selected for consistency with the test cases in the Second and Third Aeroelastic Prediction Workshops. The necessary n samples for α_0 and M are defined through LHS with a total of 70 points (Figure 4.6). Of these, 60% are selected for training, 20% for testing, and the remaining 20% for validation. The three regions marked in Figure 4.6 highlight the increasing complexity of the physics phenomena, in terms of boundary layer separation and shock wave formation on both sides of the wing. Region I, at lower M and α_0, represents conditions where the flow remains mostly attached to the surface, and shock waves are either weak or absent, resulting in minimal boundary layer separation. Region II corresponds to moderate M and α_0, where the onset of shock-induced boundary layer separation begins to occur, particularly on the upper surface of the wing, leading to more complex aerodynamic behaviour. Region III, at higher M and α_0, captures the most challenging aerodynamic conditions, with strong shock waves forming on both the upper and lower surfaces of the wing, resulting in extensive boundary layer separation and significantly nonlinear flow characteristics. This progressive increase in aerodynamic complexity across the regions provides a comprehensive dataset to validate the effectiveness of the Volterra ROM in capturing the challenging unsteady aerodynamic responses. Blue and orange circles indicate the conditions plotted in Figure 4.7.

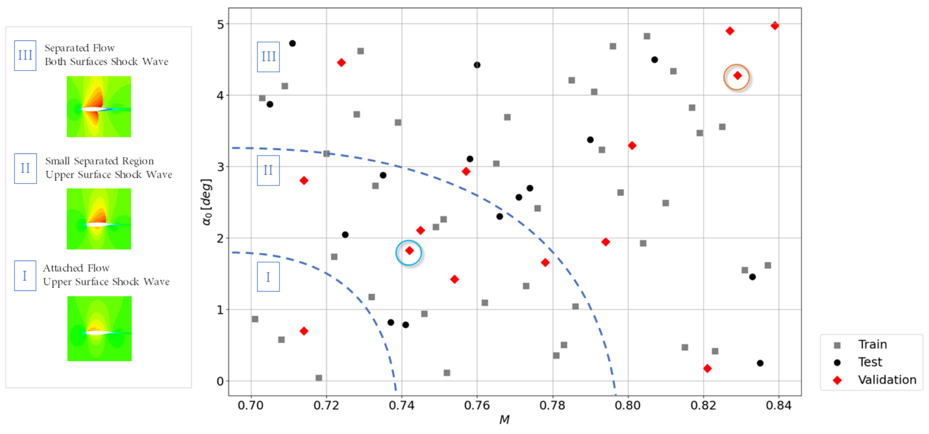


FIGURE 4.6: Training, test and validation samples for M and α_0. Three regions define the increasing complexity of the physics phenomena to be captured. Blue and orange circles indicate the conditions analysed in this section.

The excitation signal for the step responses is a "smoothed step" of the pitch angle around an axis through the wing mid–chord:

$$\alpha(\tau) = \alpha_0 \left(1 - e^{-\tau/\tau_{ref}}\right) \tag{4.31}$$

where τ is the non–dimensional time and τ_{ref} is equal to 0.6. This value is the result of a trial and error process aiming at reducing the impulsive response without affecting the circulatory part excessively. As a matter of fact the smoothing of the step input signal aims at cutting the higher part of the signal spectrum. This is not strictly necessary, but it helps achieve smoother CFD responses without affecting the frequency range used for the investigation. We generated pitch step–responses of 1 [deg] and 2 [deg] in order to cover the linear and nonlinear dynamics.

In the region II of moderate $M - \alpha_0$ (see Figure 4.6) the response of the system tends to converge to the steady–state value without large oscillations (blue line in Figure 4.7). On the other hand, when $M > 0.80$ and $\alpha_0 > 3$ [deg] in the region III, the response is characterized by the appearance of a dynamic stall which leads to an initial increase in C_L followed by a decrease toward the steady–state value (orange line in Figure 4.7).

The conditions $M = 0.829$ and $\alpha_0 = 4.277$ [deg] and $M = 0.745$ and $\alpha_0 = 2.105$ [deg], plotted in Figure 4.7, were selected to demonstrate the ROM ability to handle diverse aerodynamic scenarios. These points represent different regions in the $M - \alpha_0$ space, one near the upper boundary and the other closer to the centre. The condition $M = 0.829$ and $\alpha_0 = 4.277$ [deg] was chosen for its complex flow phenomena, including shock waves and significant boundary layer separations, testing the ROM ability to capture highly nonlinear effects. Conversely, $M = 0.745$ and $\alpha_0 = 2.105$ [deg] is in a more moderate flow regime, where the effects of compressibility and flow separation are less pronounced, serving as a baseline for assessing the ROM performance in mildly unsteady conditions.

4.5.2 Results

To evaluate the performance of the Volterra series, we first reconstructed C_L and C_M responses to a step input at $M = 0.829$ and $\alpha_0 = 4.277$ [deg]. As shown in Figure 4.8, the nonlinear Volterra kernels closely match the CFD data, confirming the Volterra series capability to capture complex aerodynamic behaviour.

Building on the methodology introduced in earlier sections, we employed two ML algorithms (FCNN and GPR) to reconstruct the Volterra kernels across the design space. Table 4.4 reports the `MAE` values for the integral aerodynamic loads on the validation set. The results clearly demonstrate the superior performance of FCNN, particularly for nonlinear reconstructions in both coefficients. This is consistent with the fact that nonlinear kernel components reflect amplitude-dependent physics such as shock displacement and pressure-centre migration, which are more difficult to interpolate smoothly across (M, α_0). To provide further insights into the performance of both ML models, the Volterra kernels reconstruction at $M = 0.745$ and $\alpha_0 = 2.105$ [deg] and at $M = 0.829$ and $\alpha_0 = 4.277$ [deg] are reported in Figure 4.9 and Figure 4.10, respectively. In Panel A, the aerodynamic loads at 1 [deg] are reconstructed using linear Volterra kernels, while Panel B displays the reconstruction of aerodynamic loads at 2 [deg] using nonlinear Volterra kernels. Both ML methods effectively capture the main features of the kernels,

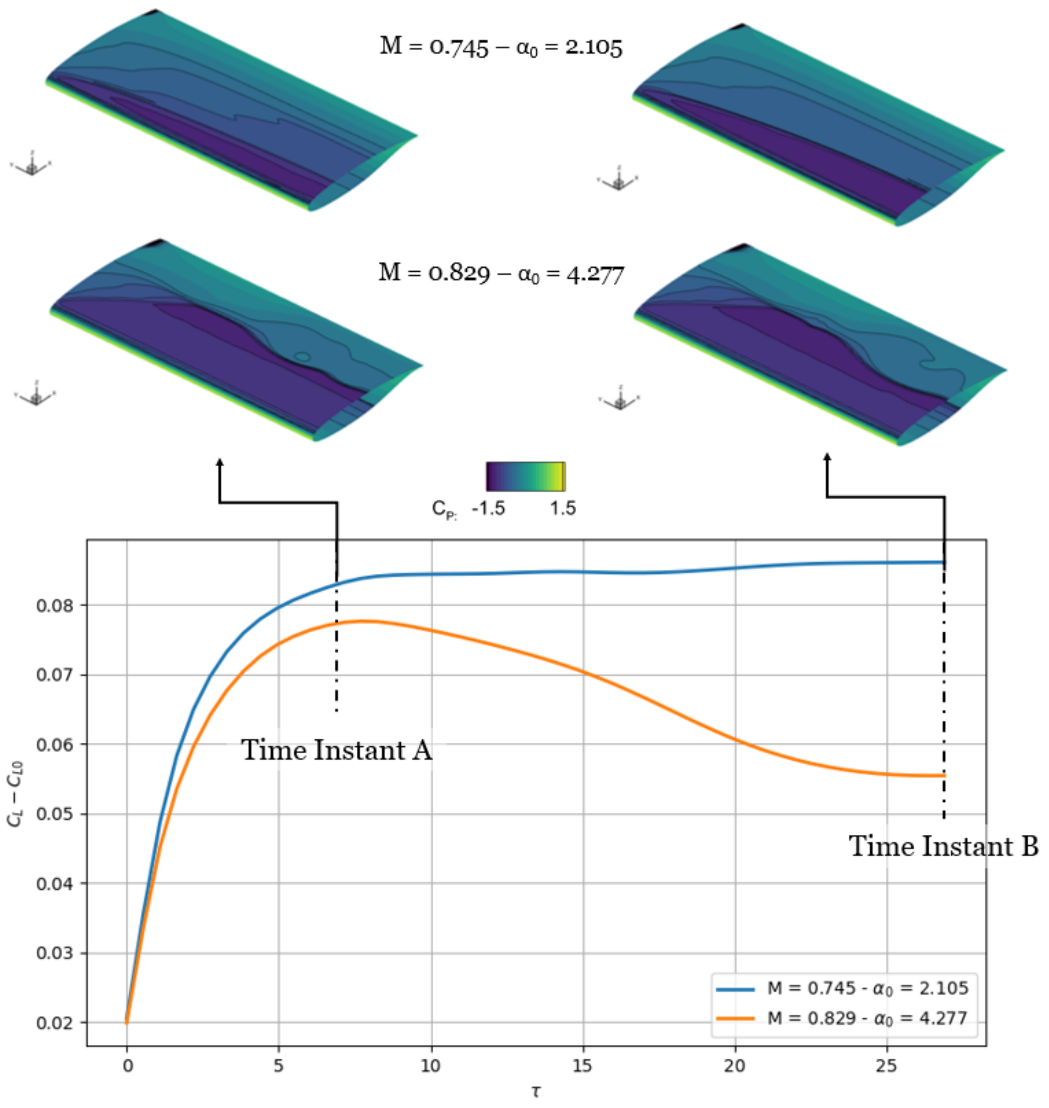


FIGURE 4.7: Lift coefficient responses at two different flight conditions with pitch input angle of 1 [deg]. The steady–state values were subtracted. Pressure coefficient contours are also added at $\tau = 7$ denoted as *Time Instant A*, and $\tau = 30$ denoted as *Time Instant B*.

showing good alignment with the reference. When these kernels are used to reconstruct the responses, they reproduce the CFD behaviour well, especially at moderate M and α_0 (Figure 4.9), where the flow is dominated by circulatory mechanisms and the nonlinear contributions remain relatively weak. It is noteworthy that the linear kernels decrease in magnitude as the step response approaches steady state, consistent with the finite memory of the quasi-circulatory component, whereas the nonlinear kernels retain larger magnitudes at higher indices, indicating low-frequency content associated with shock boundary layer interaction and gradual separation growth. However, the situation changes at high M and α_0 (Figure 4.10), the nonlinear mechanisms intensify,

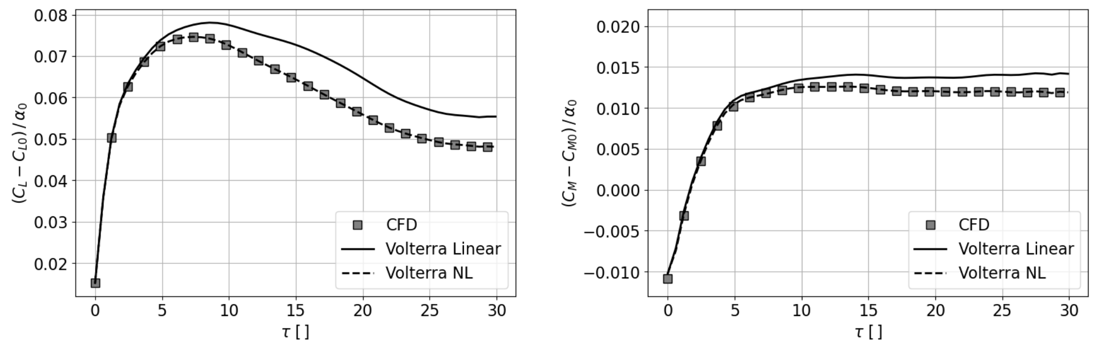


FIGURE 4.8: Aerodynamic loads reconstruction with linear and nonlinear Volterra kernels at $M = 0.829$ and $\alpha_0 = 4.277$ [deg]. The response was normalized by the pitch angle input step, after subtracting the steady–state value.

leading to stronger amplitude dependence and greater sensitivity to kernel interpolation. In this regime FCNN outperforms GPR in both linear and nonlinear cases, its regularisation yielding smoother reconstructions and filtering spurious oscillations while retaining the physically relevant low-frequency content. Based on these advantages, we have chosen to reconstruct Volterra kernels with FCNN for the two proposed applications.

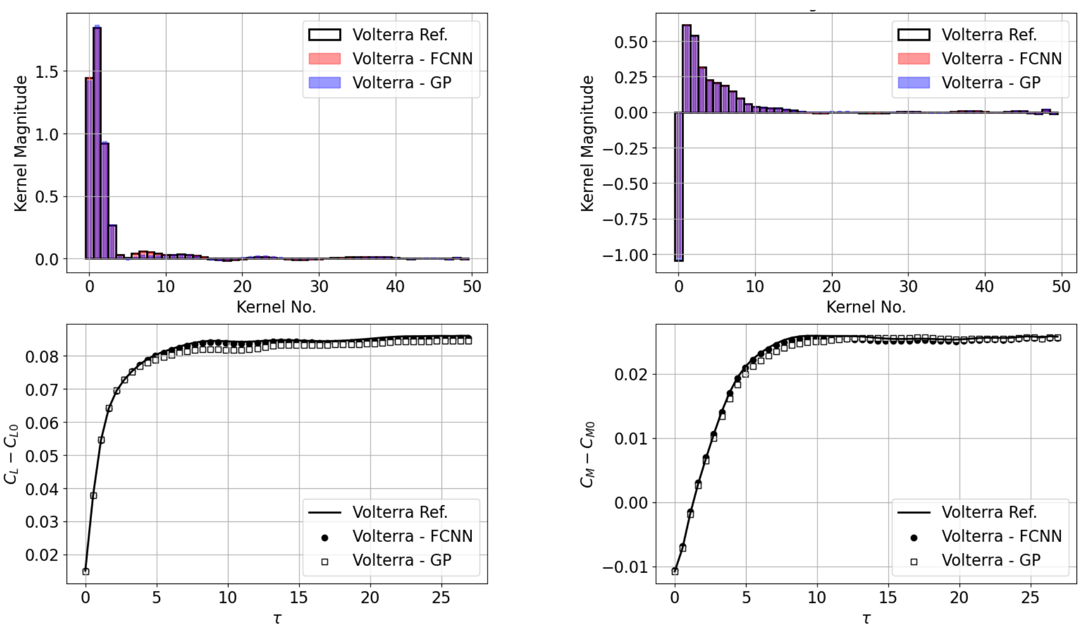
Model Name	ε_{C_L}		ε_{C_M}	
	Linear	Nonlinear	Linear	Nonlinear
FCNN	0.0049	0.0066	0.0010	0.0014
GP	0.0054	0.0081	0.0017	0.0033

TABLE 4.4: `MAE` on the validation set for the two ML models in the 3D test case.

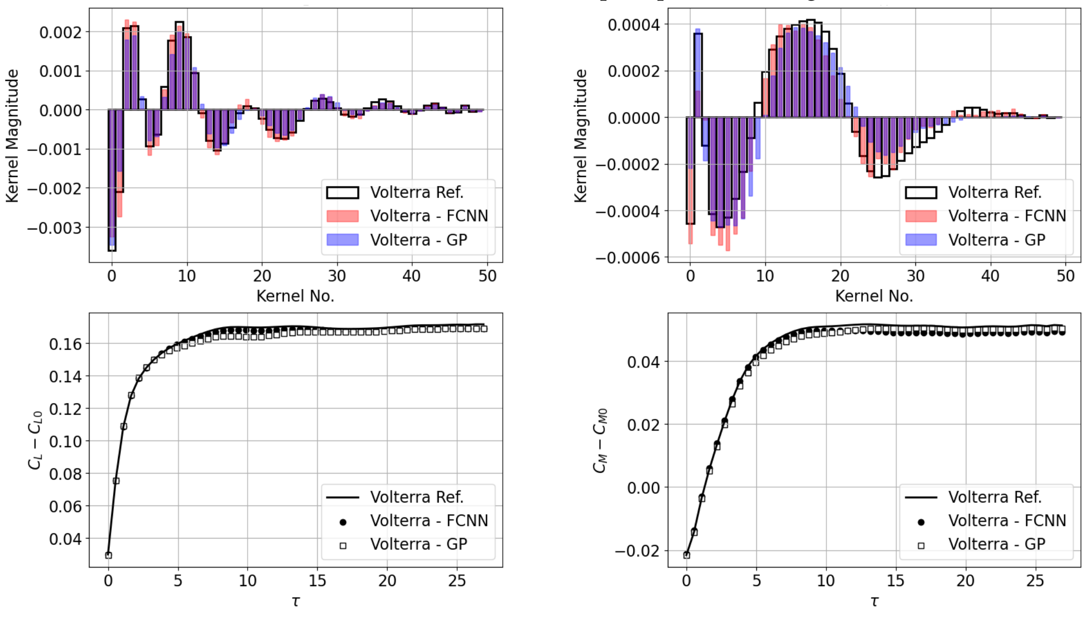
Prediction of Pitch Step–responses

We exploit the Volterra ROM methodology developed and the training data presented in the third test case (section 4.5.2) to reconstruct the pitch step response in a series of α_0 sweeps at different M (Figure 4.11).

The identified step–responses to a pitch input angle of 2 [deg] are reported in Figure 4.12. Each panel consists of six distinct responses constructed maintaining fixed M and varying $\alpha_0 \in [0, 5]$ [deg]. As we gradually increase α_0, the step-responses demonstrate a distinct dynamic stall behaviour. Notably, this behaviour emerges at progressively lower α_0 as M increases, indicating a strong influence of M on the onset of dynamic stall. This is particularly evident at higher M values, where flow separation becomes more extensive, and the interaction between shock waves and the boundary layer becomes more significant. This analysis demonstrates the effectiveness of the methodology in capturing the patterns of the aerodynamic response. The identified step-responses provide valuable insights into how changes in M and α_0 influence the dynamic stall and flow separation. These results showcase the method capability to



(A) Linear Volterra kernels for step–response of 1 [deg]



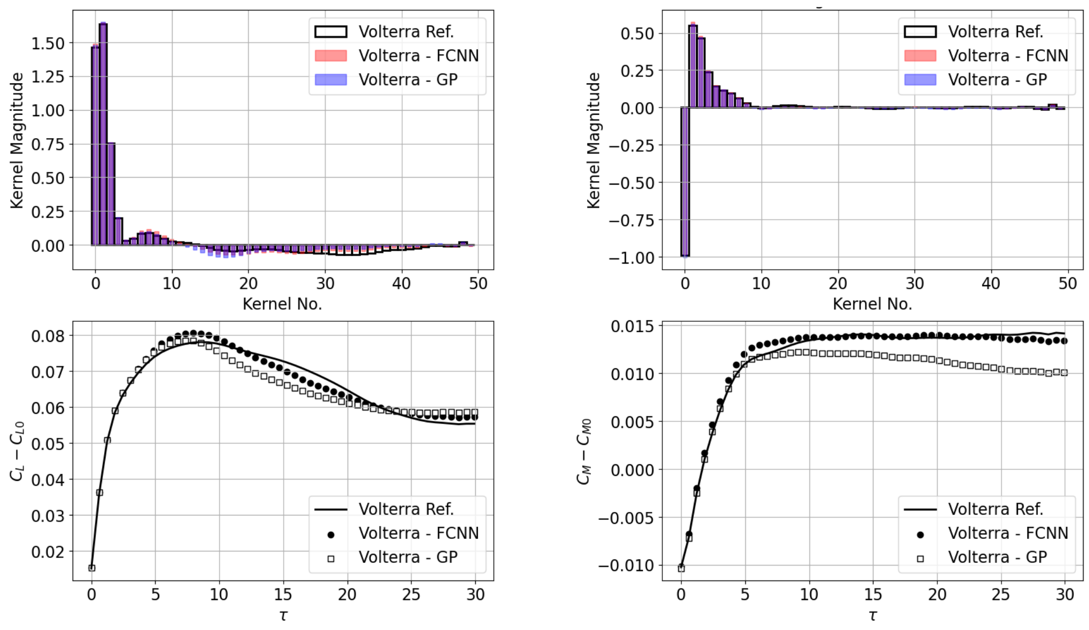
(B) NL Volterra kernels for step–response of 2 [deg]

FIGURE 4.9: Volterra kernels prediction and aerodynamic loads reconstruction at $M = 0.745$ and $\alpha_0 = 2.105$ [deg]. "Volterra Ref." denotes the Volterra nonlinear model derived from CFD data.

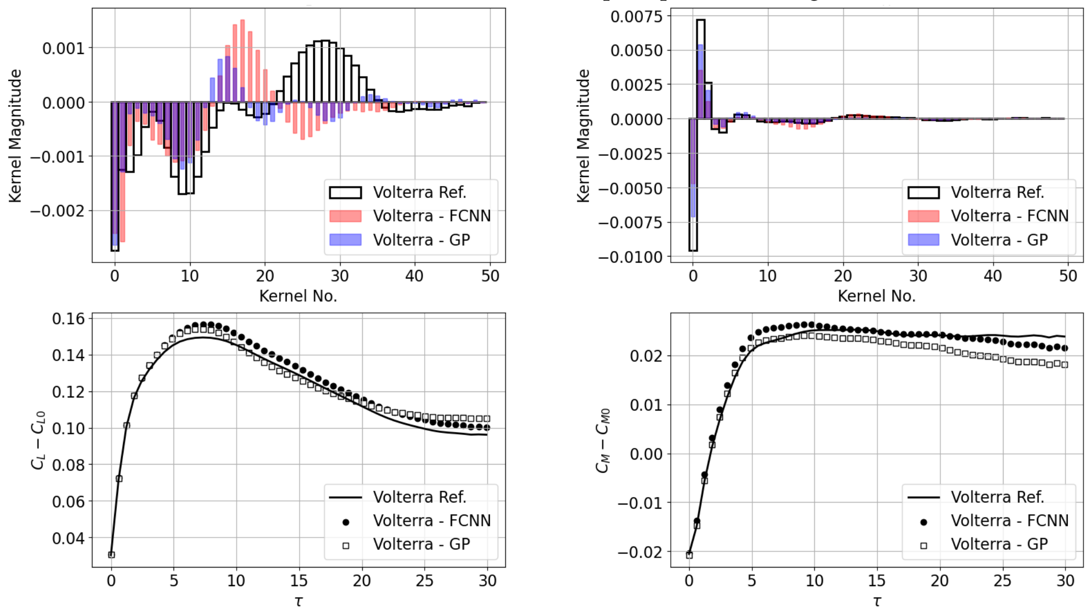
explore aerodynamic responses under a range of operating conditions, reinforcing its potential for practical applications in aerodynamic analysis.

Reconstruction of Harmonic Signals

As a second application of the Volterra ROM methodology and the training data presented in section 4.5.2, we predict the response to a harmonic signal at $M = 0.70$, $\bar{\alpha} = 5$



(A) Linear Volterra kernels for step–response of 1 [deg]



(B) NL Volterra kernels for step–response of 2 [deg]

FIGURE 4.10: Volterra kernels prediction and aerodynamic loads reconstruction at $M = 0.829$ and $\alpha_0 = 4.277$ [deg]. ”Volterra Ref.” denotes the Volterra nonlinear model derived from CFD data.

[deg] and $q = 170$ [psf]. We generated two distinct responses, one with a reduced frequency k of 0.2215, and an amplitude of $\alpha_A = 1.03$ [deg], and another with $k = 0.4386$ and $\alpha_A = 0.50$ [deg]. The validation conditions were based on Piatak et al. experimental data [269]. We initially validated our CFD data by computing the real and imaginary components of C_P/deg by performing the Fast Fourier Transform (FFT) and taking the peak frequency, and comparing them to the findings of Piatak et al. [269]. As shown in Figure 4.13, we observed a good agreement between our CFD and experimental data.

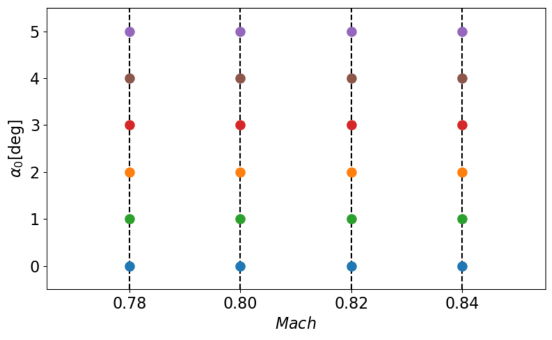
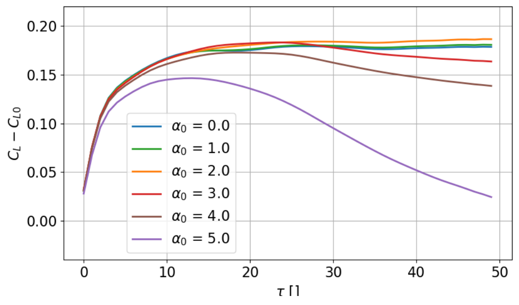
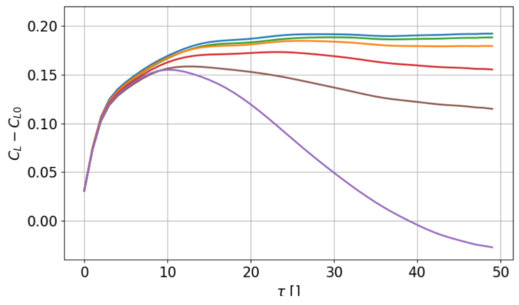


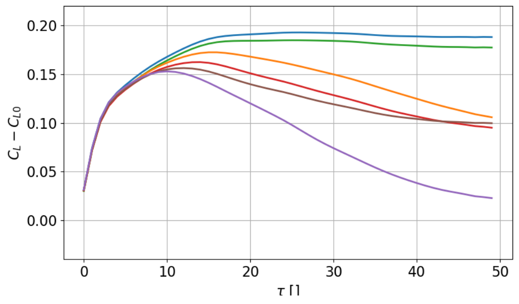
FIGURE 4.11: Parametric space of M and α_0 used to identify pitch step-responses with the Volterra ROM.



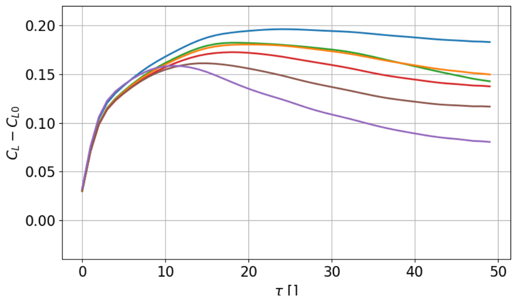
(A) $M = 0.78$



(B) $M = 0.80$



(C) $M = 0.82$



(D) $M = 0.84$

FIGURE 4.12: Pitch step–responses of 2 [deg] for several α_0 at four distinct fixed M reconstructed using nonlinear Volterra kernels predicted by FCNN model.

It is interesting to note that the pitch oscillation induces a forward and backward motion of the shock wave, resulting in significant variations in both the real and imaginary components within that specific region. These observations are in line with the findings reported by Heeg et al. [144].

After validating the CFD data, the reconstruction of C_L and C_M can be performed using the Volterra ROM as illustrated in Figure 4.14. The Volterra kernels are identified using FCNN and then convolved with the input signal to obtain the predicted results. The reconstruction of the harmonic signal is demonstrated at two reduced frequencies, 0.2215 and 0.4386. However, when predicting at high α_0, the accuracy of the results can

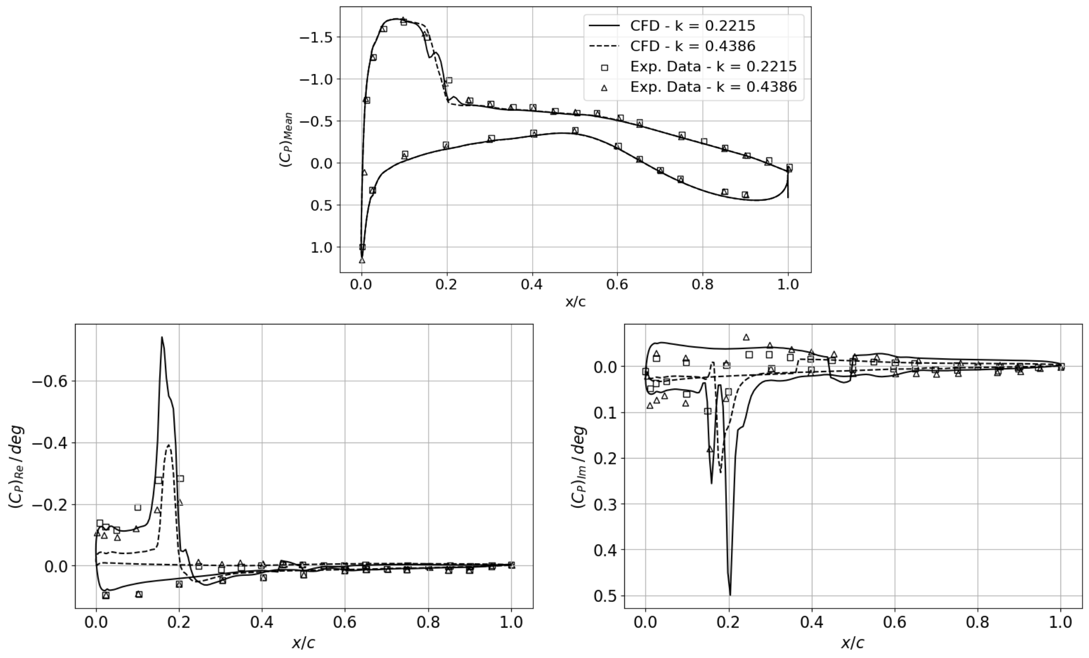
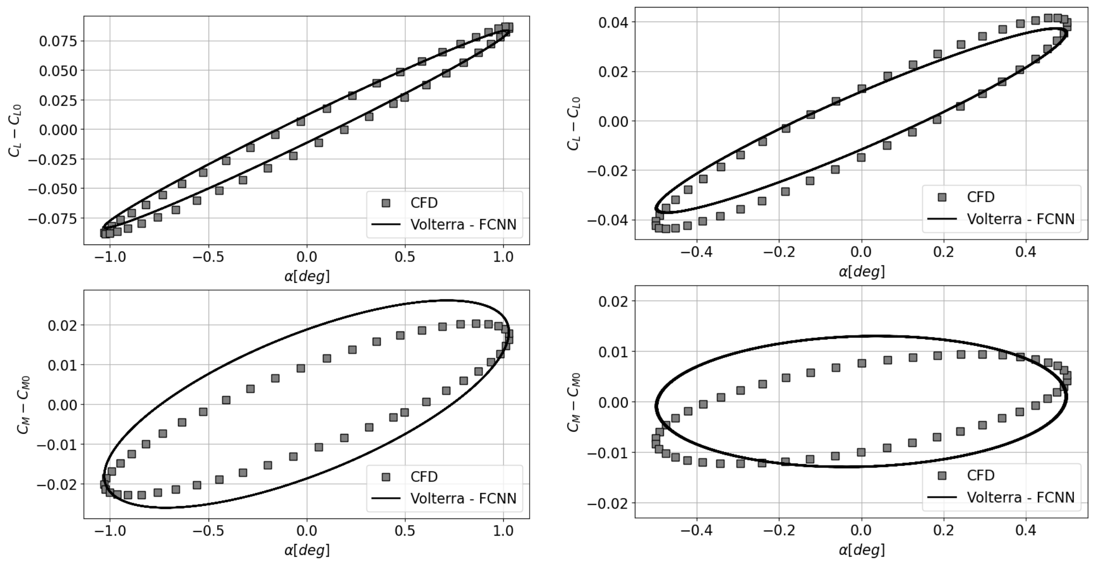


FIGURE 4.13: Mean, real and imaginary parts of the pressure coefficient at peak frequency at 60% of the span ($\bar{\alpha} = 5$ [deg], $M = 0.70$ and $q = 170$ [psf]).

be challenging due to the global instability of the flowfield, which is associated with boundary layer separation. Nevertheless, the Volterra ROM closely follows the curves.



(A) $k = 0.2215 - \alpha_A = 1.03$ [deg] (B) $k = 0.4386 - \alpha_A = 0.50$ [deg]

FIGURE 4.14: Harmonic signal reconstruction with Volterra ROM at $\bar{\alpha} = 5$ [deg], $M = 0.70$ and $q = 170$ [psf].

The method presented involves an "offline" identification of Volterra kernels using step–responses. These kernels are then applied to different frequency values of harmonic inputs to achieve dependable estimates of nonlinear loads. The reliability of these estimations can be evaluated by comparing the identified model to the step–response data. The robustness of this approach ensures accurate nonlinear estimation, as long as the step–responses match the desired levels closely (refer to Figure 4.15).

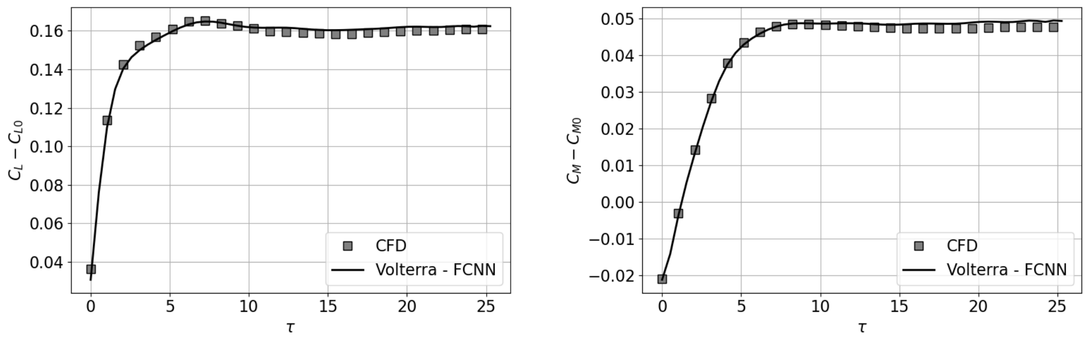


FIGURE 4.15: Step–response prediction at $\alpha_0 = 5$ [deg], $M = 0.70$ and $q = 170$ [psf].

Finally, we present four M sweeps in Figure 4.11. As α_0 increases, we observe a corresponding increase in the magnitude of hysteresis, as depicted in Figure 4.16. These findings are consistent with the experimental results reported by Heeg et al. [146], further validating our observations.

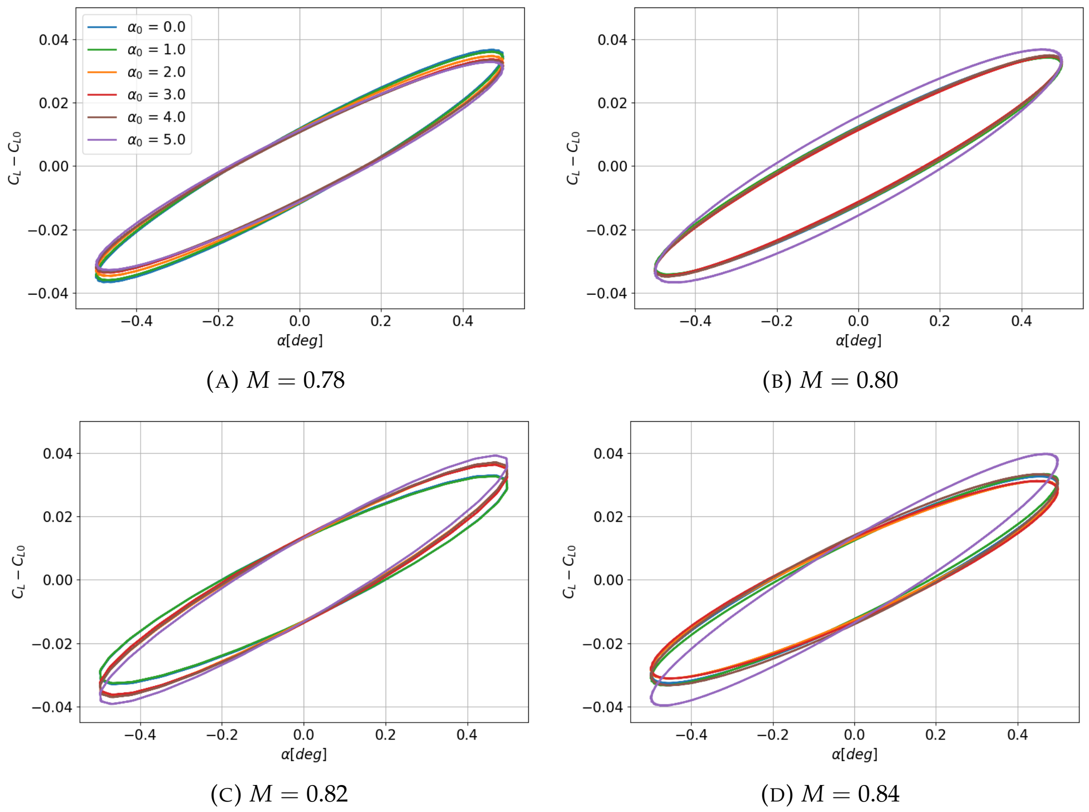


FIGURE 4.16: Harmonic signal reconstruction for different α_0 at fixed M ($k = 0.4386$ - $\alpha_A = 0.50$ [deg]) using nonlinear Volterra kernels predicted by FCNN.

4.5.3 Volterra framework for flutter analysis

Building on the Volterra series identification outlined earlier in this chapter, we now embed the resulting aerodynamic ROM of the BSCW within the structural model to efficiently simulate aeroelastic responses and predict flutter boundaries. This integration leverages the time-domain nature of the Volterra-based approach to account for both transient and sustained oscillations in a coupled aero-structural system.

Before presenting the governing equations of the structural motion of the wing, it is useful to distinguish between Limit Cycle Oscillation (LCO) and flutter. LCO occurs when nonlinearities and feedbacks result in a stable, self-sustaining oscillation of limited amplitude, often leading to fatigue and reduced performance. In our framework, we detect LCO by integrating the coupled equations of motion at different dynamic pressures and monitoring whether the oscillation amplitude reaches a stable, non-zero limit. In contrast, flutter involves unbounded oscillation growth that ultimately leads to failure. In traditional modal analysis, flutter is revealed when one or more eigenvalues of the coupled system have positive real parts, indicating exponential growth. However, linear methods may be inadequate in transonic or other highly nonlinear regimes. Here, the Volterra-based aerodynamic model efficiently captures nonlinear unsteady flow over a wide range of conditions, enabling both early flutter detection and LCO prediction at significantly lower computational cost than full CFD-based approaches.

Compared with Chapter 3, where the focus was on a 2D airfoil and the aerodynamic ROM was obtained through end-to-end ML (directly mapping motion states to aerodynamic loads), the present chapter shifts to a 3D wing and adopts a two-stage approach. First, the nonlinear system dynamics are identified using Volterra series operators at selected flight conditions, which explicitly encode memory and nonlinearities through kernel convolutions. ML is then employed not to predict loads directly, but to interpolate these identified kernels across the $M - \alpha_0$ space, yielding a parametric ROM that can be coupled with the structural model. While the end-to-end 2D approach proved effective in quasi-linear conditions, it struggled as nonlinearities intensified: autoregressive roll-outs accumulated error outside the training domain, polynomial augmentations captured only weak hysteresis, and the lack of explicit memory limited the ability to represent shock-boundary layer interactions. By contrast, the nonlinear Volterra kernels used here incorporate memory and amplitude-dependent cross-effects, allowing the framework to capture stronger transonic nonlinearities and maintain stability over long horizons. This distinction highlights the methodological evolution: from direct load prediction in 2D to operator-based system identification and interpolation in 3D, enabling more reliable nonlinear response modelling, stable flutter tracking, and efficient mapping of flutter boundaries and LCO behaviour.

Structural System

Having established the Volterra-based aerodynamic framework, we now turn our attention to the structural equations governing the BSCW wing. As depicted in Figure 4.17, the pitch angle is denoted by θ, and the plunge displacement by h. The system is described by the following second-order ordinary differential equations:

$$\begin{cases} m\ddot{h} + S_\theta\ddot{\theta} + c_h\dot{h} + K_h h = -L \\ S_\theta\ddot{\theta} + I_\theta\ddot{\theta} + c_h\dot{\theta} + K_\theta\theta = M \end{cases} \tag{4.32}$$

In these expressions, m, S_θ, and I_θ are, respectively, the wing's mass, first moment of inertia, and second moment of inertia. The quantities K_h and K_θ denote linear spring constants, while c_h and c_θ indicate structural damping coefficients (commonly set to zero if viscous damping is negligible). Aerodynamic loads, represented by the lift L and pitching moment M, are referenced to a hinge axis at the 50% chord. For the present problem, both the elastic axis and the centre of gravity coincide at the mid-chord.

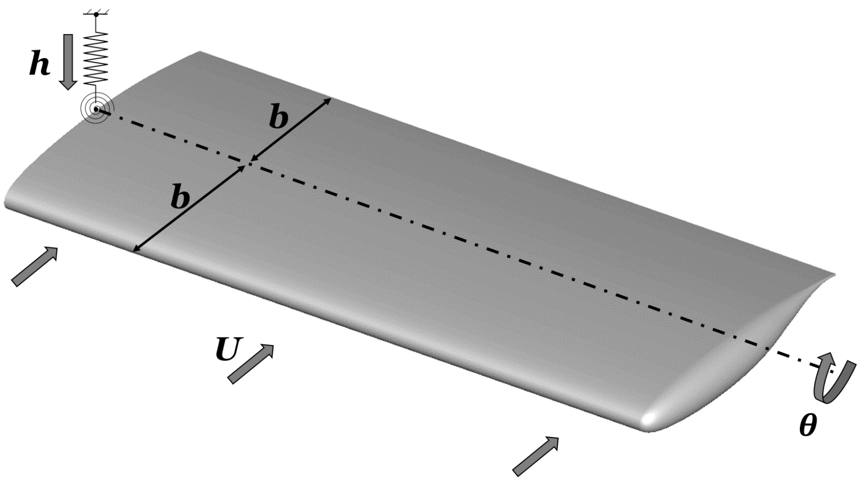


FIGURE 4.17: Schematic of the half-span BSCW subject to pitch and plunge motion.

It is often more convenient to express the two-degree-of-freedom system in matrix form:

$$M_s\ddot{x} + C_s\dot{x} + K_s q = f \tag{4.33}$$

where the generalised coordinate vector q, the mass matrix M_s, the damping matrix C_s, the stiffness matrix K_s, and the force vector f are defined as:

$$q = \begin{pmatrix} h \\ \theta \end{pmatrix}, \quad M_s = \begin{bmatrix} m & S_\theta \\ S_\theta & I_\theta \end{bmatrix}, \quad C_s = \begin{bmatrix} c_h & 0 \\ 0 & c_\theta \end{bmatrix}, \quad K_s = \begin{bmatrix} k_h & 0 \\ 0 & k_\theta \end{bmatrix}, \quad f = \begin{pmatrix} -L \\ M \end{pmatrix}$$

To couple these structural dynamics with the Volterra-based aerodynamics in a time-marching framework, we recast Eq. (4.33) into a system of first-order ordinary differential equations:

$$\begin{cases} \dot{q} = Aq + Bu \\ y = Cq + Du \end{cases} \tag{4.34}$$

where q collects the state variables $(h,,\theta,,\dot{h},,\dot{\theta})$, and u represents external inputs (here, the aerodynamic loads). In the standard aeroelastic problem where the output y coincides with the state vector, the matrix C becomes the identity and D vanishes. We then write:

$$q = \begin{pmatrix} h \\ \theta \\ \dot{h} \\ \dot{\theta} \end{pmatrix}, \qquad A = \begin{bmatrix} 0 & I \\ -M_s^{-1}K_s & -M_s^{-1}C_s \end{bmatrix}, \qquad B = \begin{pmatrix} 0 \\ M_s^{-1}f \end{pmatrix}$$

For time-domain simulations, the continuous equations are discretized to enable efficient numerical integration. The resulting discrete-time system is:

$$q^{n+1} = A_D\, q^n + B_D\, f^n, \tag{4.35}$$

with:

$$A_D = (I - \Delta t \beta A)^{-1} \, (I + \Delta t(1-\beta)A), \qquad B_D = (I - \Delta t \beta A)^{-1} \, B$$

where Δt is the time step and β is a parameter in the chosen integration scheme.

In the present work, the pitch and plunge DOFs are combined into a single effective angle of attack α_{eff}, allowing us to embed the Volterra kernels directly into the state space formulation. Specifically, the required lagged values of the α_{eff} are appended to the state vector q, forming an augmented state vector x whose dimension includes both the structural DOFs and the memory terms from the Volterra kernel. Although treating pitching and plunging separately would be more general, the pragmatic choice of unifying them into a single effective angle helps reduce complexity while still capturing the fundamental unsteady effects.

At each discrete time step n, the aerodynamic loads—lift L and pitching moment M—are obtained from the Volterra kernels using α_{eff}^n. Rather than treating θ and h separately, α_{eff}^n is defined as a function of the current structural state $q^n = \left(h^n, : \theta^n, : \dot{h}^n, : \dot{\theta}^n\right)$. Consequently, the aerodynamic lift and moment are obtained at each time step via:

$$L^n = H_L \, \alpha_{\text{eff}}^n, \qquad\qquad M^n = H_M \, \alpha_{\text{eff}}^n$$

where H_L and H_M are the matrices of linear Volterra kernels specific to lift and moment, respectively, and α_{eff}^n is the vector of the m effective angle of attack values used in the model for the n^{th} time instance which depends on each DOF:

$$\alpha_{\text{eff}}^n = \begin{bmatrix} 0 & 1 & 1/V & 0 \end{bmatrix} q^n,$$

with V denoting the freestream velocity.

Introducing α_{eff}^n and its lagged values into the system entails extending the original state vector q^n to account for the memory terms in the Volterra kernels. Specifically, if one requires m steps of the effective angle-of-attack history, only $m-1$ additional entries are required, because α_{eff}^n itself is computed directly from q^n. The augmented state x^n thus becomes:

$$x^n = \begin{pmatrix} h^n \\ \theta^n \\ \dot{h}^n \\ \dot{\theta}^n \\ \alpha_{\text{eff}}^n \\ \alpha_{\text{eff}}^{n-1} \\ \dots \\ \alpha_{\text{eff}}^{n-m+1} \end{pmatrix}$$

Once the augmented state x^n is defined, the system response at $n+1$ can be estimated with:

$$x^{n+1} = \mathbf{A}\, x^n + \{f\}^n \tag{4.36}$$

where f^n includes both external forces and nonlinear correction to the aerodynamic loads for higher–order Volterra kernels. The block structure of the state matrix $\mathbf{A}$ incorporates the discrete–time structural operator A_D, its input mapping B_D, and the appropriate shifts for the lagged angle-of-attack entries:

$$
\mathbf{A} = \begin{array}{cc}
 & \begin{array}{cc} 2\,DOF & m \end{array} \\
\begin{array}{c} 2\,DOF \left\{ \vphantom{\begin{array}{c}a\\a\\a\end{array}} \right. \\ m \left\{ \vphantom{\begin{array}{c}a\\a\\a\\a\end{array}} \right. \end{array} &
\left[\begin{array}{c|cccc}
A_D & & \mathbf{0} & & \\
 & & B_D\,H_L & & \\
 & & B_D\,H_M & & \\
\hline
 & 0 & & & \mathbf{0} \\
 & 1 & \ddots & & \\
\mathbf{0} & & \ddots & & 0 \\
 & \mathbf{0} & & & 1
\end{array}\right]
\end{array}
$$

This augmented formulation maintains a clear separation between the structural solver (hidden in A_D and B_D) and the Volterra-based aerodynamic model (encoded in H_L and H_M).

The structural proprieties of the BSCW wing test case [143, 145, 282] are reported in Table 4.5. Here, the structural damping is neglected and the positions of the elastic axis and the centre of gravity are coincident at the mid–chord.

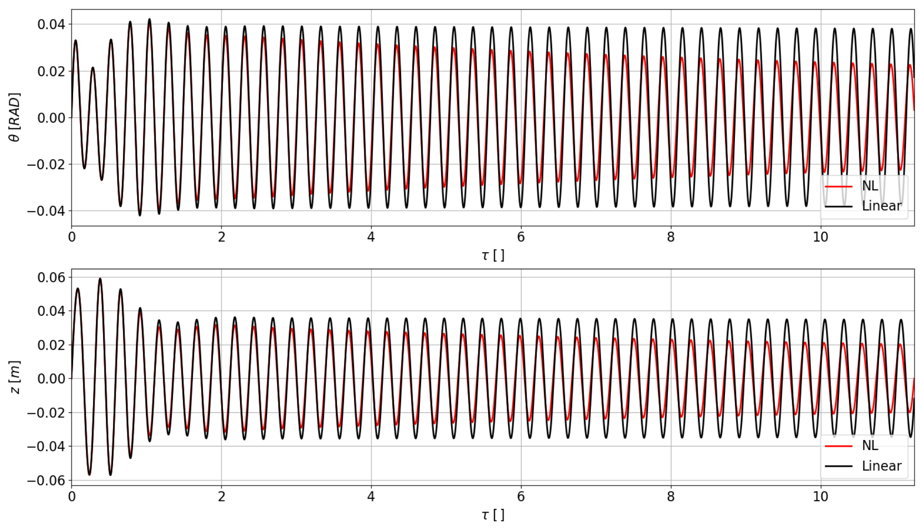
BSCW properties	
K_θ	$4018.64\,\mathrm{Nm\,rad^{-1}}$
K_h	$38484.12\,\mathrm{N\,m^{-1}}$
m	$87.91\,\mathrm{kg}$
I	$3.765\,\mathrm{kg\,m^2}$
c	$0.4064\,\mathrm{m}$

TABLE 4.5: Structural properties of the BSCW model.

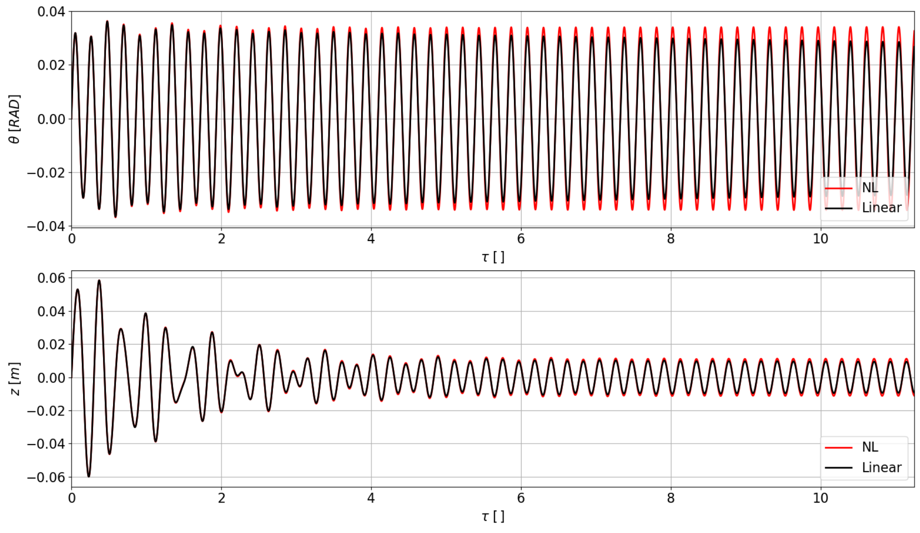
Incorporating Volterra kernels into this structural solver yields a robust, time-accurate means of capturing flutter instabilities and LCO. As shown later in the results, the aerodynamic kernel provides both the linear and nonlinear unsteady loads, enabling accurate representation of shock-wave oscillations and shock-induced boundary layer separation at transonic conditions. Consequently, this reduced-order, discrete-time coupling of the structural system and Volterra-based aerodynamics provides a computationally efficient way to obtain accurate aeroelastic predictions over a range of flight conditions.

Flutter prediction

Figure 4.18 illustrates the BSCW wing flutter response predicted via the Volterra-based aerodynamic model, under both linear and nonlinear settings, for two distinct flight conditions. At lower Mach numbers and moderate angles of attack, as in Figure 4.18a, the flutter arises earlier for the linear case—consistent with classical flutter theory. By contrast, at higher Mach numbers and larger angles of attack (Figure 4.18b), the nonlinearity in the aerodynamic loads induces flutter before the linear model would predict. We suspect that this phenomenon corresponds to a subcritical LCO.



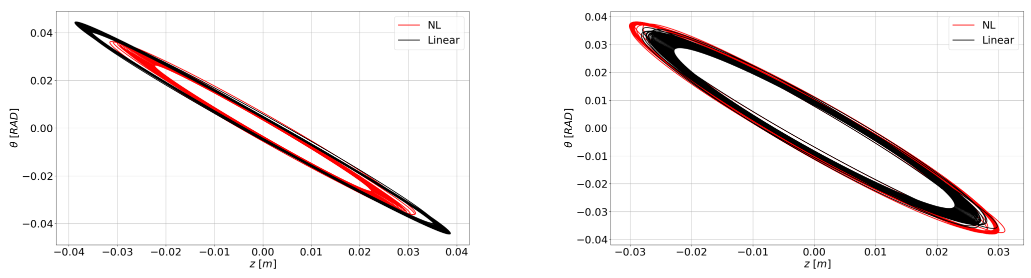
(A) $M = 0.745 - \alpha_0 = 2.105\,[deg] - q = 180\,[psf]$



(B) $M = 0.829 - \alpha_0 = 4.277\,[deg] - q = 303\,[psf]$

FIGURE 4.18: Pitch and plunge responses in time during flutter at two different flight conditions for BSCW wing.

By examining the stability behaviour over a range of Mach numbers and angles of attack, one can map out the flutter dynamic pressure surface shown in Figure 4.20. As Mach and angle of attack increase, the required dynamic pressure for flutter generally rises. However, above $M = 0.80$ and approximately $\alpha_0 = 3$ [deg], predicting the onset of flutter becomes especially sensitive to the flow conditions. When extensive boundary layer separation occurs, aerodynamic damping is drastically reduced, thus amplifying the system susceptibility to flutter.



(A) $M = 0.745 - \alpha_0 = 2.105\,[deg] - q = 180\,[psf]$

(B) $M = 0.829 - \alpha_0 = 4.277\,[deg] - q = 303\,[psf]$

FIGURE 4.19: Pitch–plunge hysteresis during flutter at two different flight conditions for BSCW wing.

In attached or mildly separated flows, C_L and C_M are effectively in phase, facilitating relatively straightforward flutter predictions. However, in strongly separated regimes (often referred to as "dynamic stall" conditions), the phase shifts between C_L and C_M become more pronounced, and nonlinear hysteresis effects emerge. Consequently, low reduced-frequency oscillations (around 0.05 in the test conditions) can trigger large-amplitude motions unless the unsteady aerodynamic damping is sufficiently high.

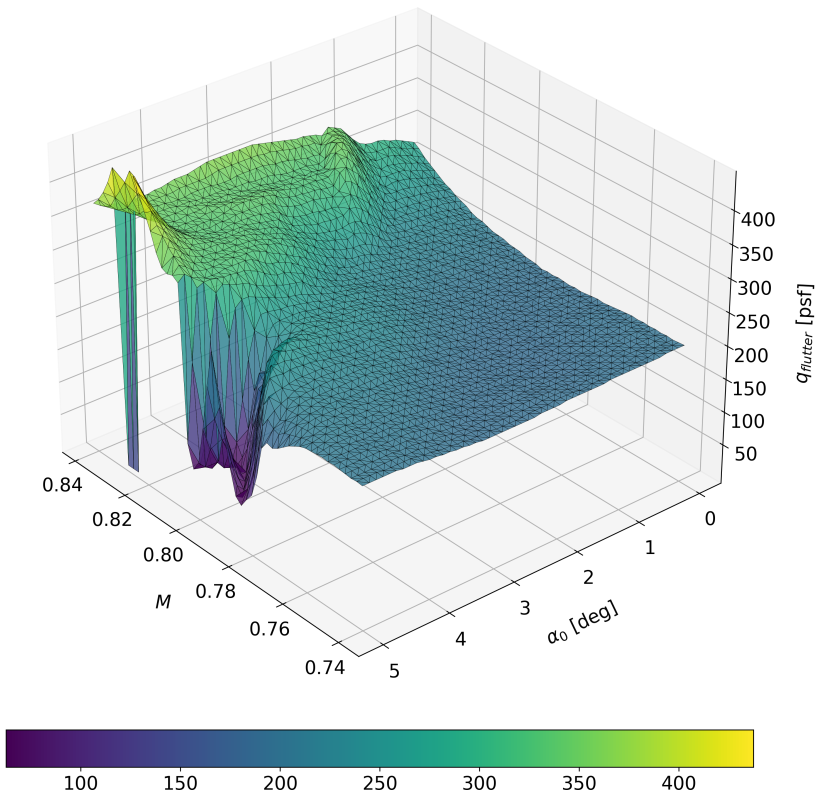


FIGURE 4.20: Flutter boundary for BSCW wing.

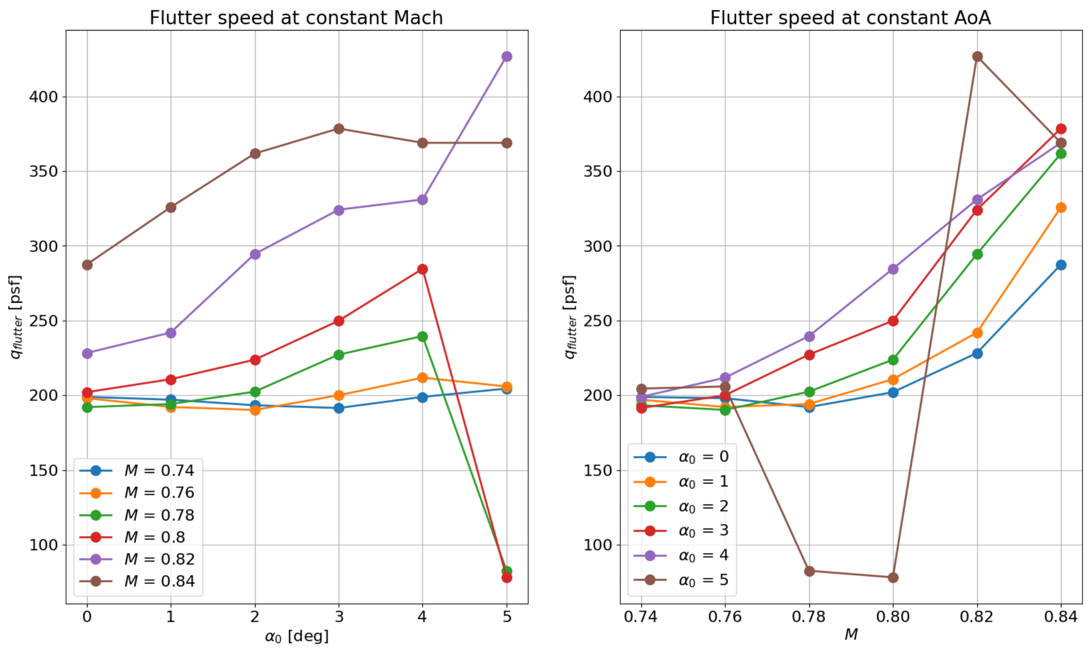


FIGURE 4.21: Flutter speed at constant Mach and constant α_0 for BSCW wing.

Data Fusion

While the Volterra ROM captures the dominant nonlinear transonic mechanisms (shock–boundary layer interaction, dynamic stall, hysteresis), residual bias can remain near the most nonlinear regimes and at the edges of the training set. Conversely, high-fidelity data (e.g. wind-tunnel flutter points) are sparse but quantitatively reliable. Multi-fidelity data fusion combines the broad coverage of the low-fidelity model (Volterra ROM) with the accuracy of high-fidelity measurements to refine q_{flutter}. In this framework, dense low-fidelity evaluations from the Volterra ROM are first used to construct a smooth predictor of flutter speed across the domain. The predictions of this low-fidelity model, x_i, are then evaluated at the locations of the available high-fidelity samples and used to augment the input space, $x_i^{aug} = [\alpha_i, M_i, \hat{\eta}(x_i)]$. A Co–Kriging (CK) model, $\hat{\eta}(x_i^{aug})$, is subsequently trained on this augmented set, as described by Da Ronch et al. [77], allowing the low-fidelity trend to act as an informative feature that improves correlation among the high-fidelity data. This procedure yields a bias-corrected estimate of flutter speed, effectively combining the strengths of both levels of fidelity. The resulting fused model shifts portions of the predicted flutter boundary towards the experimental trend with only a limited number of high-fidelity points. In regimes where nonlinear separation reduces aerodynamic damping, the CK approach corrects systematic bias from the low-fidelity surrogate while retaining its dense coverage. Figure 4.22 shows the updated flutter boundary obtained with the data fusion technique proposed. Red circles denote experimental data from Dansberry et al. [83].

Figure 4.23 further breaks down the new results along cuts of constant Mach or constant angle of attack. In several regions, the boundary moves to lower dynamic pressures,

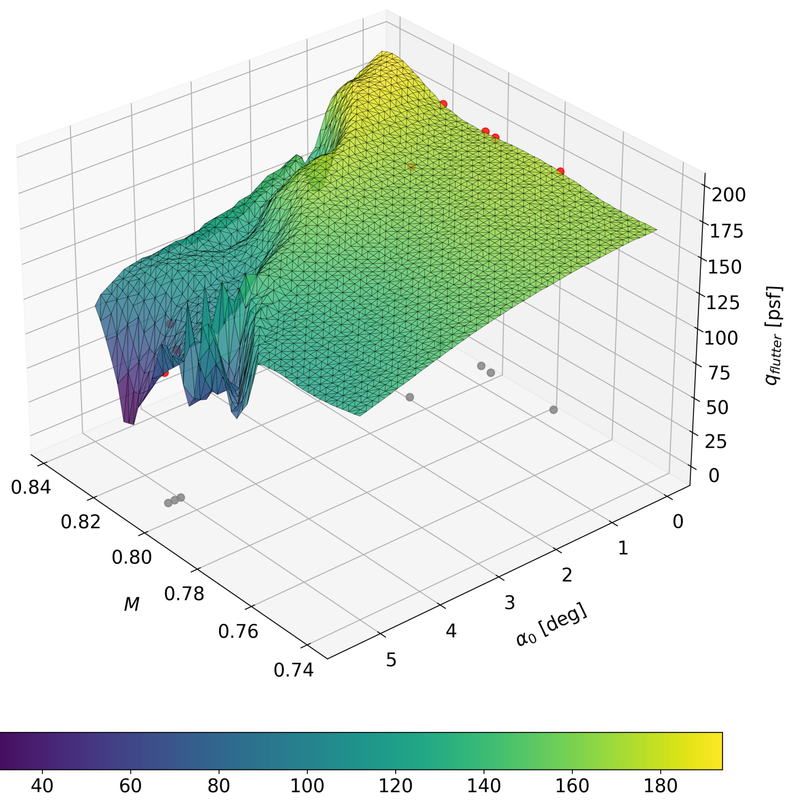


FIGURE 4.22: Flutter boundary for BSCW wing with data fusion. Red circles denote experimental data from Dansberry et al. [83]. Grey circles indicate their projection on the $M - \alpha_0$ plane.

highlighting how even sparse high-fidelity data can realign the surrogate predictions with experimental trends and reduce the residual bias of the ROM.

4.6 Computational Cost Analysis

A detailed computational cost analysis was performed to evaluate the efficiency of the Volterra ROM in comparison to the high-fidelity CFD simulations. Table 4.6 summarizes the costs for training and prediction. In the 2D case, CFD requires 126,000 CPU hours for 140 runs, with a single run taking 900 hours. In contrast, the Volterra ROM with the GPR model required 0.01 GPU hours for training and only 0.0003 GPU hours (approximately 1 second) for a single prediction. The FCNN model in the same 2D case involved 1.6 GPU hours for optimisation, 0.03 hours for training, and similarly 0.0003 GPU hours for a single prediction. In a practical application, the virtually instantaneous predictions could provide significant computational savings. The 3D case demonstrates an even larger discrepancy, as CFD requires 560,000 CPU hours for 140 runs for training (4,000 hours for a single run). The Volterra ROM with the GPR model

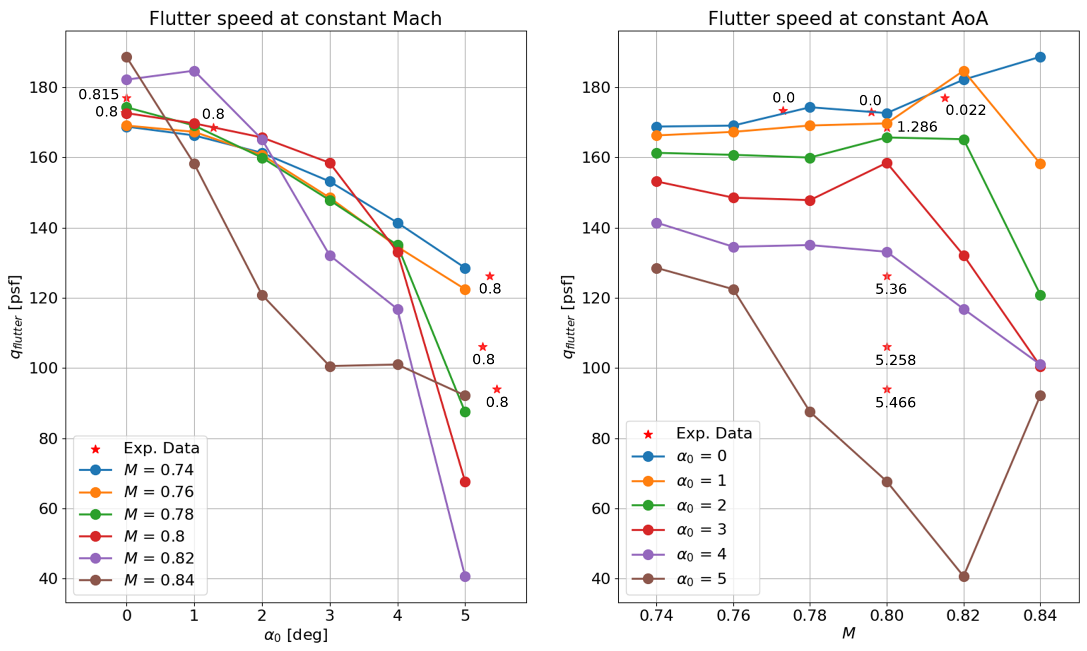


FIGURE 4.23: Flutter speed at constant Mach and constant α_0 for BSCW wing with data fusion. Red stars denote experimental data from Dansberry et al. [83], with the adjacent numbers indicating the corresponding Mach or α_0 condition.

needed just 0.01 GPU hours for training and 0.0003 GPU hours for prediction, while the FCNN model required 1.6 GPU hours for optimisation, 0.03 GPU hours for training, and the same 0.0003 GPU hours for prediction.

This comparison highlights the efficiency and scalability of the Volterra ROM, providing accurate predictions with drastically lower computational demands compared to traditional CFD simulations.

Each CFD simulation was performed on a high-performance computing system with an Intel Skylake-based architecture, utilizing 3 nodes with 40 CPU cores each. In contrast, the training of the Volterra model was executed on an Intel XEON W-2255 CPU paired with a NVIDIA RTX A4000 GPU.

Test case	**CFD (CPU hours)**		**Volterra ROM (GPU hours)**			
	Simulation (140 runs)	(1 run)	Model Type	optimisation (50 trials)	Training (1 model)	Prediction (1 sample)
2D	126,000	900	GPR	–	0.01	0.0003 (~ 1s)
			FCNN	1.6	0.03	0.0003 (~ 1s)
3D	560,000	4,000	GPR	–	0.01	0.0003 (~ 1s)
			FCNN	1.6	0.03	0.0003 (~ 1s)

TABLE 4.6: Computing cost comparison between Volterra ROM and CFD for the two test cases. 140 runs include 70 for small-amplitude input and 70 for large-amplitude input.

4.7 Conclusions

Predicting steady and unsteady aerodynamic characteristics in the transonic regime is needed for the development of high-speed aircraft. A compromise between lowering the computing costs and increasing the prediction accuracy is generally found in a ROM representation of the aerodynamics. Many are the possible representations, but none has become the standard. We considered a ROM based on Volterra series for a single-input, single-output system, where separate Volterra models were constructed for the two aerodynamic outputs of interest (C_L and C_M).

A key advantage of this framework is that it inverts the usual mapping of few inputs to many outputs. By parametrising the Volterra series in the 2D design space, the high-dimensional family of unsteady responses is compressed into a compact set of kernel coefficients. This efficient reduction of the output space alleviates the computational burden of kernel identification and simplifies subsequent predictions. Any new flow condition can be recovered by evaluating the same low-order kernels, rather than generating a whole new, full-order solution.

The series contains first-order (linear) and second-order (nonlinear) Volterra kernels to adequately capture the physical complexity resulting from unsteady, transonic flows around 2D and 3D configurations. A two-step procedure is used to identify the Volterra kernels, justified by the dominant nature of the linear component in any nonlinear response. First, the linear kernels are identified from a response obtained with a small-amplitude input. Then, the nonlinear kernels are identified from the difference between a large-amplitude input response and the small-amplitude input response. To increase the readiness of the approach for future applications, such as aeroelasticity and optimisation, we developed a parametric Volterra series by applying two ML algorithms over the $M - \alpha_0$ design space.

Through these steps, we have demonstrated that the Volterra series can accurately capture both linear and nonlinear aerodynamic responses in the transonic regime. By reconstructing the kernels with GPR and (more effectively) FCNN, the methodology successfully reproduced complex phenomena such as dynamic stall over a wide range of M and α_0. The ML-based interpolation allowed the model to effectively generalize within the parametric space and handle different aerodynamic conditions with high accuracy. The analysis of the dynamic response via Volterra ROM may allow substantial savings in computational cost compared to CFD, as the training only exploited two responses per flow condition and the predictions can be produced in the entire parameters space at virtually no computational costs. The approach proved scalable for both 2D and 3D configurations, confirming its potential for more complex aerodynamic systems. However, in the present formulation, pitch and plunge motions were combined into a single effective angle of attack used to excite the system. This modelling choice, while simplifying the identification process and reducing computational effort, introduces a significant limitation. By compressing two DOFs into a single input, the framework cannot distinguish the independent contribution of pitch and plunge to each aerodynamic output.

The results also demonstrate that flutter in the BSCW wing can arise from a combination of classical linear mechanisms and nonlinear shock/boundary layer interactions, especially beyond $M = 0.80$. These interactions introduce subcritical LCO and significant hysteresis, complicating stability predictions. By enriching the Volterra ROM with high-fidelity data through CK, the method matches experimental flutter trends more closely than the low-fidelity method alone, thereby improving the reliability of flutter estimates in strongly nonlinear regimes.

From a broader thesis perspective, these findings reinforce key research objectives:

1. **Rapid, Non-Intrusive Nonlinear Aerodynamic Modelling**
   The Volterra-ROM framework reproduces key transonic nonlinearities, such as dynamic stall, shock-boundary layer coupling, and limit-cycle oscillations, while reducing the inference computational burden by two orders of magnitude compared to time-accurate CFD. This chapter addresses Research Objective O1 by developing a ROM for modelling the temporal evolution of global aerodynamic loads. It provides better generalisation capabilities than the quasi-linear surrogate model presented in Chapter 3, providing higher fidelity without compromising turnaround time for preliminary design or flight-load certification.

2. **Output-Space Compression and Prediction Efficiency**
   By expressing the many high-dimensional flow responses through a compact set of first- and second-order kernels, the method flips the usual "few-inputs$\rightarrow$many-outputs" mapping into a ""few-inputs$\rightarrow$few-kernels" paradigm. This compression facilitates the identification of kernels and enables the fast and efficient evaluation of new flow conditions across the design space.

3. **Robust Parametric and Multi-Fidelity Extensions**
   The Volterra ROM is first identified at selected conditions, and ML models (GPR or FCNN) interpolate smoothly the kernels across large portions of the design space. The resulting parametric ROM enables rapid flutter tracking, which is then refined by Co-Kriging with sparse high-fidelity data (wind-tunnel) to correct systematic bias. This hierarchy preserves the efficiency of the Volterra-ML surrogate while aligning the predictions with experimental trends.

4. **Scalability to Realistic 3D Configurations**
   Successful application to the BSCW wing confirms that the approach scales with state dimension, retaining accuracy despite complex 3D shock dynamics and subcritical LCO.

To maintain kernel identification tractable and limit model complexity, the Volterra expansion was truncated at second order. This level of approximation is sufficient for small-amplitude manoeuvres, but higher-order memory terms may become significant in large-amplitude phenomena such as transonic buffet or deep stall. Extending the framework to third- or fourth-order kernels would in principle address these effects, but the associated growth in coefficients scales poorly with both state dimension and parameter space, making such models computationally prohibitive.

Future work shall be directed towards extending the methodology to multi-input systems to better capture coupled aeroelastic effects. Although straightforward in principle, identification of the nonlinear kernels requires system responses in which the inputs are applied simultaneously to capture any interaction between them. Such a development will further extend the capabilities of the ROM and provide broader utility in aeroelastic stability analysis, flight control, and design optimisation. Also, while the present study relies on a classical data fusion strategy (Co-Kriging), Chapter 5 demonstrates that more advanced techniques, such as multi-fidelity Bayesian neural network, can further enhance predictive accuracy and uncertainty quantification across fidelity levels.

Chapter 5

Multi–Fidelity Bayesian Network for Uncertainty Quantification in Aerodynamic Loads

Multi-fidelity surrogate modelling has gained significant momentum in the aerospace industry due to its potential to combine the computational efficiency of lower-fidelity simulations with the accuracy of high-fidelity data. Typically achieved through data fusion, this strategy reduces the overall cost of data generation while maintaining predictive accuracy. However, although traditional ML methods have been widely used to improve surrogate models and fuse data from multiple sources, there is a continuing need for novel approaches that further improve predictive reliability, particularly with respect to uncertainty quantification, without incurring excessive costs for generating high-fidelity samples. In direct response to Research Objective O2 of this thesis, this chapter presents a Bayesian neural network framework tailored for multi-fidelity aerodynamic prediction. Using transfer learning, we systematically combine CFD data of different fidelities. The probabilistic formulation of this model allows for explicit uncertainty quantification over the input space, making it particularly well-suited for the inherently nonlinear and complex nature of transonic flows and coupled wing-propeller interactions. Our results show that the proposed multi-fidelity Bayesian model outperforms classical data fusion approaches such as Co-Kriging, both in terms of prediction accuracy and generalisation to unseen flow conditions.

5.1 Introduction

Accurate prediction of aerodynamic loads is central to the design and optimisation of modern aircraft. In transonic flows, where shock waves and boundary layer separation can significantly affect aerodynamic performance, conventional modelling approaches often struggle to balance fidelity and computational cost. This has led to a growing interest in strategies that integrate data from multiple sources of varying fidelity, exploiting the efficiency of low-fidelity models alongside the accuracy of high-fidelity simulations. Such strategies are particularly relevant for complex flight regimes and broader applications, including aeroelasticity and flight dynamics [42, 110, 373, 389].

Low-fidelity modelling methods focus on capturing the dominant flow physics at a reduced computational cost [54, 74, 342]. Their efficiency is derived from simplifying assumptions and empirical corrections that approximate system behaviour without fully resolving complicated phenomena [180, 219, 224, 245, 368]. Examples include vortex-based approximations [313, 387] and various reduced-order formulations [96, 194, 201]. Although these approaches can quickly provide important aerodynamic indicators, they often exhibit limited reliability in regions of the flight envelope that exhibit a highly nonlinear behaviour [105, 406].

Data fusion (DF) methods attempt to address this limitation by merging multiple levels of fidelity into a more comprehensive surrogate model [140, 291]. A canonical example is combining a low-fidelity, data-driven predictor with sparse, high-fidelity observations to refine estimates in highly nonlinear regions. Co-Kriging (CK) remains one of the most established DF methods, using Gaussian process formulations to blend low- and high-fidelity data, improve prediction accuracy, and quantify model uncertainty [114, 216, 268]. Its robust performance has kept it relevant for newer aerodynamic optimisation tasks [77, 139].

ML, particularly in the form of deep learning (DL), has opened new possibilities for DF by providing flexible architectures that effectively capture nonlinear relationships [316, 326]. Although large training datasets are required, often expensive to generate using high-fidelity CFD, DF approaches can mitigate this burden by leveraging the combination of low- and high-fidelity data. Transfer learning (TL) is a widely used technique in this context, where models are first trained on rich low-fidelity data before being fine-tuned with limited high-fidelity samples [210, 414], improving predictive capabilities in resource-constrained or computationally expensive applications [58, 212].

Although these ML-driven DF surrogates generally outperform traditional data-driven methods in describing highly nonlinear behaviour [105, 131, 281], many still lack integrated uncertainty quantification mechanisms. Bayesian Neural Networks (BNNs) address this gap by learning probability distributions over weights, thus quantifying both model and data uncertainties [157, 348]. Studies have investigated BNNs in multi-fidelity scenarios [239, 315], discussing the computational cost and challenges of combining data from heterogeneous sources. One promising solution involves hybrid approaches, such as combining GPR and BNN layers, to investigate the different strengths of different techniques [179].

These approaches complement other research directions that aim to unify DF techniques with uncertainty quantification. An example is the Multifidelity Monte Carlo method [262], which leverages correlated low- and high-fidelity models to accelerate statistical estimation while retaining accuracy guarantees. Similarly, Deep Gaussian Processes [82] extend classical GPR by introducing hierarchical latent functions, providing a way to propagate uncertainty across multiple levels of fidelity and capture complex correlations. More recently, Multifidelity Kolmogorov-Arnold Networks [156] have been proposed as a neural architecture capable of decomposing nonlinear mappings into a hierarchy of additive and compositional components, offering a flexible framework for integrating information across fidelities while preserving theoretical approximation guarantees. While these studies are attracting increasing attention, our work focuses on a more straightforward framework, built on the idea that combining two relatively simple concepts such as BNN and TL can already yield satisfactory results, without the need for more exotic or overly complex approaches.

For this reason, the aim of our study is to develop a ML-based multi-fidelity surrogate model that predicts integrated aerodynamic loads, using exclusively BNN layers, and integrating TL for the data fusion process. This approach has led to the design of a more straightforward architecture, solely based on the DL paradigm, capable of capturing the complex nonlinearities of transonic flows and coupled wing-propeller interactions. This architecture will be directly compared with CK since both allow for multi-fidelity prediction while quantifying epistemic and aleatoric uncertainties. CK was chosen as the baseline not only because of its simple formulation, but also because it is a widely used standard technique applied to aerospace problems [77, 110, 139].

5.2 Methodology

This section details the steps involved in creating the surrogate model. We implemented a multi-fidelity framework integrating BNNs with TL technique to harness diverse data sources and enhance model generalisation. The approach includes quantification of uncertainty to ensure reliable predictions, along with systematic optimisation of model hyperparameters for optimal performance. CK model is introduced in order to perform a comparative analysis, benchmarking the efficacy of our proposed method.

5.2.1 MF-BayNet

The core of our methodology revolves around a multi-fidelity surrogate model that combines BNNs with TL, herein referred to as MF-BayNet, which is designed to predict integrated transonic aerodynamic loads, specifically lift and pitching moment coefficients, across different flight conditions. First, the process begins by training a BNN on a large dataset of LF data to capture general trends of aerodynamic behaviour over the full design space. Then, the model undergoes TL on mid-fidelity (MF) data where certain layers of the model are frozen, so that low-level features remain fixed; then a subset of layers is retrained using the MF dataset. This partially corrects LF inaccuracies—especially in regions with transonic shocks or mild boundary-layer interactions. Finally, a limited number of the remaining trainable layers are fine-tuned using a few expensive HF samples. This final step adds important fine-scale corrections to the network, enabling it to capture strong shocks, boundary-layer separations, and other highly nonlinear phenomena.

Transfer Learning Process

In the context of DF, once the model has captured the basic understanding of the general behaviours during initial training, the first layers of the model are frozen by increasing the fidelity of the training set. This means that all frozen neurons have fixed values of weights during training, which in the case of BNN are represented by the mean and standard deviation of each probability distribution. After this phase, we get a pre-trained network that has gained a basic knowledge of the problem. Then, the subset of trainable parameters of the model is retrained on one or more small sets of increasingly higher fidelity data. A schematic of the architecture is illustrated in Figure 5.1.

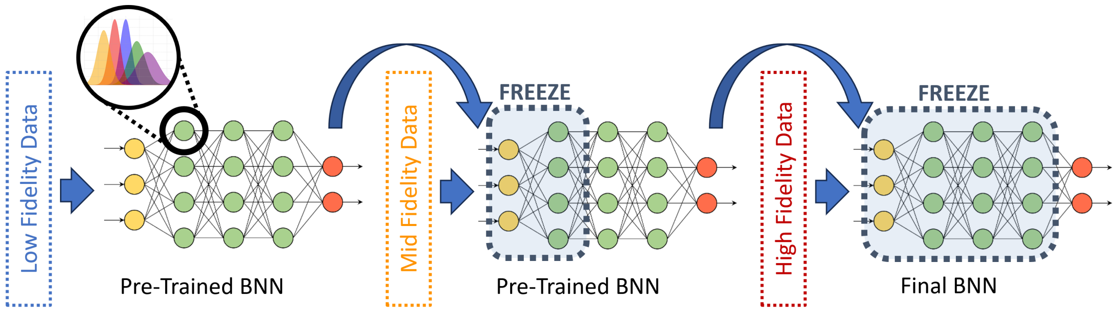


FIGURE 5.1: Schematic of the MF-BayNet architecture

The TL approach significantly reduces the computational cost and time required for training due to the reduced subset of trainable neurons, while improving the model ability to make accurate predictions using different fidelities and emphasizing the importance of higher fidelity data. In addition, this method relies only on DL principles, which makes it more straightforward and robust compared to other methods. Two TL processes were executed: the first on MF data and the second on HF data points to fine-tune the model. In the second process, fewer layers were frozen compared to the first, allowing for more refined adjustments. By progressively transferring learned representations from one fidelity level to the next, the MF-BayNet framework fully exploits large amounts of approximate LF data while carefully incorporating MF and HF points to correct errors in the more challenging regions. This approach drastically reduces the need for large HF datasets while achieving high accuracy and built-in uncertainty quantification. By restricting this last stage of training to a minimal number of layers, or even just the output layer, the network retains most of the generalised aerodynamic knowledge acquired from the LF and MF datasets. Consequently, the model maintains predictive capabilities even in areas lacking HF data, potentially outperforming models trained exclusively on limited HF samples. The TL process is explained in Algorithm 3.

Algorithm 3 MF-BayNet Training with Low-, Mid-, and High-Fidelity Datasets

```
1: Input: Low-Fidelity Dataset $D_{LF}$, Mid-Fidelity Dataset $D_{MF}$, High-Fidelity Dataset $D_{HF}$, Initial Model $M$
2: Output: Trained Multi-Fidelity Model $M_{MF-BayNet}$
3: Step 1: Train on Low-Fidelity Data
4: $M \leftarrow$ Initialize Bayesian Neural Network (BNN)
5: $M_{LF} \leftarrow$ train($M$, $D_{LF}$)
6: Step 2: Transfer Learning to Mid-Fidelity Data
7: freeze($M_{LF}$, $N_{frz}^{LF \rightarrow MF}$)
8: $M_{MF} \leftarrow$ train($M_{LF}$, $D_{MF}$)
9: Step 3: Transfer Learning to High-Fidelity Data
10: freeze($M_{MF}$, $N_{frz}^{MF \rightarrow HF}$)
11: return $M_{MF-BayNet} \leftarrow$ train($M_{MF}$, $D_{HF}$)
```

Prediction Means and Confidence Interval

Once the TL process is completed, we use Monte Carlo Sampling [51] (MCS) to obtain the prediction means and standard deviations from the surrogate model, which are used for estimating the uncertainty and reliability in predictions. First, multiple forward passes are performed through the BNN. Each pass generates a different set of weights due to the nondeterministic behaviour of the model, effectively creating an ensemble of models. Next, the outputs of these forward passes are averaged to obtain the mean prediction. This gives us an estimate of the expected value of the prediction. Finally, the standard deviation of the outputs from multiple forward passes is calculated to assess the uncertainty in the predictions, incorporating both epistemic and aleatoric components. The choice of a good number of passes is therefore a trade-off between computational efficiency and the accuracy of the uncertainty estimation. A higher number of passes leads to a more precise evaluation of both the mean prediction and its associated uncertainty, but at the cost of increased inference time. In practical applications, an optimal balance must be found based on the specific requirements of the task. For example, in scenarios where fast predictions are needed, a lower number of passes may be preferable despite a slight reduction in accuracy, whereas in offline analyses, a larger number of passes can be used to maximize reliability. To determine the optimal setting, conduct sensitivity analyses or compare the uncertainty estimates against reference solutions is suggested.

Uncertainty Quantification

Uncertainty quantification (UQ) is essential for understanding the reliability of the model predictions. The MF-BayNet surrogate model provides a probabilistic interpretation of predictions, offering insights into both model and data uncertainty. The total uncertainty in the prediction, the predictive uncertainty (P_u), is defined as the sum of epistemic (E_u) and aleatoric uncertainty (A_u) [57, 92, 93].

$$P_u = E_u + A_u \tag{5.1}$$

Epistemic uncertainty refers to the uncertainty in the model parameters. This can be visualized as the spread of the posterior weight distribution $p(w|D)$. In ML, this type of uncertainty emerges when the model has not encountered data that adequately represents the entire design domain, or when the domain itself needs further refinement or completion. This type of uncertainty arises due to deficiencies from a lack of knowledge or information [76, 290]. In contrast, aleatoric uncertainty arises from the inherent variability in the input data. Given a specific input and fixed weight parameters, high aleatoric uncertainty indicates that the output estimate is subject to noise. This kind of uncertainty refers to the intrinsic randomness in the data, which can derive from factors such as data collection errors, sensor noise, or noisy labels [290].

This distinction is particularly important in multi-fidelity modelling with TL, where the TL process impacts the aleatoric uncertainty associated with different fidelity inputs. Only the subset of neurons with trainable parameters captures the probabilistic information from these inputs, making discrepancies between fidelity levels a significant source of aleatoric uncertainty. Although higher-fidelity datasets are generally more reliable, this is not always clear beforehand, complicating the assessment of overall uncertainty during TL. These complexities make it challenging to fully isolate aleatoric uncertainty, as some degree of epistemic uncertainty remains intertwined with it. In the first part of the results presented in this study, the uncertainty primarily reflects epistemic contributions from the model itself. In contrast, the second part specifically examines aleatoric uncertainty, though a complete separation between the two remains difficult due to their inherent interactions in the multi-fidelity TL process.

5.2.2 Model Optimisation

Choosing the right hyperparameters of the network is a complex task. This requires a deep understanding of the model architecture and the specific characteristics of the data at different fidelity levels in order to create a good hyperparameter optimisation process. We implemented a Bayesian optimisation [323] strategy to obtain the best set of hyperparameters for each model. Bayesian optimisation strength lies in its iterative approach to fine-tuning hyperparameters using Bayesian probability distributions, rather than exhaustively testing every possible combination. Each iteration involves training the network with a defined set of hyperparameters and optimizing them based on past trials performance with respect to the validation set metric. This cycle repeats until the optimal outcome is attained.

The design parameters targeted for the optimisation process include the number of units per layer (N_{units}), the total number of layers in the model (N_{layers}), activation functions, optimisation function, batch size, number of epochs, and the learning rate for each training phase. After the $i-th$ TL phase, the model needs to be retrained with a different learning rate value (lr_i). Other parameters include the prior distribution (initial probability distribution for each weight), determined by the mean (μ_{prior}) and variance (σ_{prior}) values of a Gaussian function, and the number of layers to freeze during TL (N_{frz}). The last step is a critical task, as freezing too many layers might prevent the model from adapting to the new, higher-fidelity data, while freezing too few layers can lead to excessive retraining and potentially overfitting. The design space for all hyperparameters, including the possible values and step size for each variable, is presented in Table 5.1. A total number of 300 trials per model tested has been executed, with an average time per trial of 3 minutes. The dataset was divided into 70% for training and 30% for validation.

Hyperparameter	**Value**	**Step size**
N_{layers}	3 to 6	1
N_{units}	16 to 176	16
lr_i	$1 \cdot 10^{-4}$ to $1 \cdot 10^{-1}$	$5 \cdot 10^{-3}$
μ_{prior}	-1.5 to 1.5	$5 \cdot 10^{-2}$
σ_{prior}	$1 \cdot 10^{-4}$ to $1 \cdot 10^{-2}$	$5 \cdot 10^{-4}$
N_{frz}	1 to $(N_{layers} - 1)$	1
Activation f	ReLU, PReLU, LeakyReLU	–

TABLE 5.1: Hyperparameters design space in Bayesian optimisation.

During each trial of the optimisation process, the model parameters, which are represented by the mean and variance values of the probability distributions of each neuron in the BNN model, along with bias values of the activation functions, are optimised based on the `MAE` on the validation set.

5.2.3 Co-Kriging

CK is a traditional approach for integrating low- and high-fidelity simulation data and serves as a benchmark in this study. Following the methodology outlined in Chapter 4, the CK function, denoted as $\hat{\eta}$, is first computed from low-fidelity evaluations and applied at high-fidelity sample points. The input parameters, such as Mach and AoA, at these high-fidelity locations, x_i, are then expanded to include the CK-estimated low-fidelity values, forming the augmented vector $x_i^{aug} = [x_i, \hat{\eta}(x_i)]$. This expanded dataset enables a refined CK function, $\hat{\eta}(x_i^{aug})$, to enhance correlation modelling between fidelity levels.

To implement this framework, CK model was first trained on the MF dataset. In this stage, the augmented kernel vector was defined as $k_i^{aug} = k_{\text{MF}}(x_i, X_{\text{MF}}) + \hat{f}_{\text{LF}}(x_i)$, where $\hat{f}_{\text{LF}}(x)$ denotes the prediction of a single-fidelity BNN trained exclusively on LF samples, as described in Section 5.3. Once the intermediate CK model converged, it was employed as a surrogate interpolator to construct a new CK formulation trained on HF data. In this second stage, the augmented kernel vector was expressed as $w_i^{aug} = k_{\text{HF}}(x_i, X_{\text{HF}}) + \hat{f}_{\text{CK-MF}}(x_i)$, where $\hat{f}_{\text{CK-MF}}(x)$ represents the prediction from the previously trained CK surrogate that incorporates both LF and MF information. Through this sequential application of CK principles, the final GPR-based model integrated all three fidelity levels, yielding a multi-fidelity surrogate with enhanced predictive accuracy and reduced uncertainty, formally expressed as $\hat{f}_{\text{CK}}(x) = \mathcal{GPR}(\mu(x), k(x, x'))$, where $\mu(x)$ and $k(x, x')$ are conditioned on the joint LF, MF, and HF datasets.

5.2.4 Performance metrics

The error metrics ε_μ [%], ε_σ [%], and ε_{tot} [%] are computed to evaluate the performance of the models predictions on test cases for each output variable. The Percentage Error on Mean Prediction (ε_{μ_i} [%]) is derived by calculating the MAE between prediction means $\tilde{\mu}_y$ calculated on MCS, and ground truth values y, normalized by the output label range, thus $\text{Range}_i = |\max(\mathcal{D}_i) - \min(\mathcal{D}_i)|$ where $\mathcal{D}_i$ represents the vector of values on dataset column i:

$$\varepsilon_{\mu_i}[\%] = \left(\frac{\text{MAE}(y, \tilde{\mu}_y)}{\text{Range}_i}\right) \times 100. \tag{5.2}$$

The Percentage Error on Standard Deviation (ε_{σ_i} [%]) measures the model uncertainty in its predictions (lower is better). The predicted standard deviations $\tilde{\sigma}_y$ calculated on MCS are averaged and then normalized by the range of the corresponding output label:

$$\varepsilon_{\sigma_i}[\%] = \left(\frac{\text{avg}(\tilde{\sigma}_y)}{\text{Range}_i}\right) \times 100. \tag{5.3}$$

The Total Percentage Error (ε_{tot} [%]) provides an aggregated measure of the overall prediction error across all output labels. This metric is computed as the root mean square of the individual ε_{μ} [%] values, given by:

$$\varepsilon_{tot}[\%] = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(\varepsilon_{\mu_i}[\%]\right)^2}, \tag{5.4}$$

where n is the number of output labels.

5.3 Finite Wing Test Case

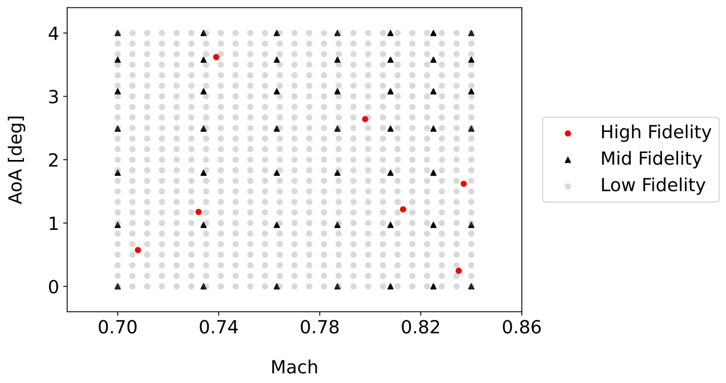
This section outlines the application of the MF-BayNet surrogate model for predicting aerodynamic loads on the BSCW by leveraging a combination of low-, mid-, and high-fidelity datasets. The identified input parameters include AoA and Mach number, which characterize the influence of the transonic flow regime on the aerodynamic performance. C_L and C_M represent the two outputs to predict. To evaluate the effectiveness of the MF-BayNet approach, multiple models were developed, optimised, and compared. Three single-fidelity BNN models were trained to evaluate the impact of TL technique. Each model was individually optimised to minimize the `MAE` on its respective validation dataset, simulating the development of models using one fidelity source at a time. Additionally, the same optimised LF-BNN model was used to provide LF interpolated samples to expand MF dataset for the CK. Finally, the MF-BayNet surrogate model was trained using the TL framework that sequentially incorporated LF, MF, and HF datasets on subset of layers. The number of layers frozen from the left for each TL phase is also defined as a hyperparameter and optimised. The optimised hyperparameters for all models used in this test case are reported in A.3.1.

5.3.1 Multi-Fidelity Datasets

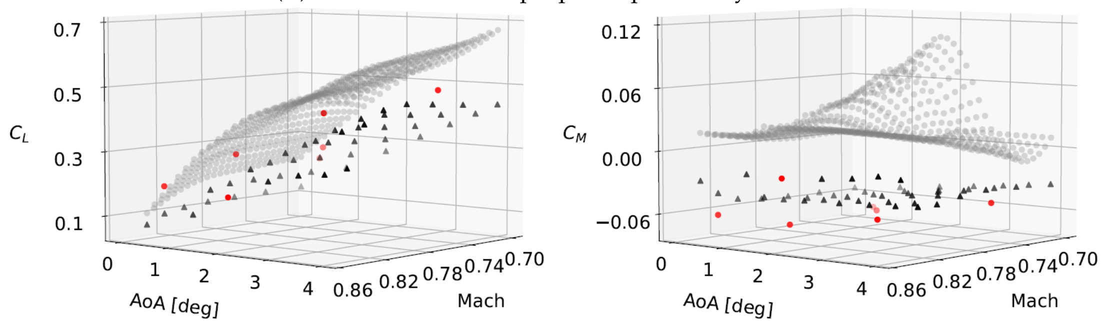
The BSCW configuration and geometry (refer to Section 4.5) make it an ideal test case for generating low-, mid-, and high-fidelity aerodynamic data using various techniques. The wing is elastically suspended on a flexible mount system with two degrees of freedom, pitch and plunge. However, in this case study, we focus solely on the wing itself, excluding the pitch-plunge system.

The datasets include a large number of LF samples and a limited number of mid- and high-fidelity samples. LF data come from panel method, providing quick but approximate solutions. Mid- and high-fidelity data, derived from CFD simulations with different number of grid points, offer detailed and accurate results but are computationally expensive. This hierarchical arrangement of fidelity levels is a key to our multi-fidelity approach, providing both broad aerodynamic coverage and targeted refinement where nonlinear effects become significant. In particular, at relatively low AoA and Mach numbers, the aerodynamic response remains weakly nonlinear, with Mach exhibiting a stronger influence on the loads than AoA. As both parameters increase, nonlinear phenomena become more significant - shock waves form on the wing and boundary-layer separation occur, leading to pronounced changes in lift and pitching moment. In particular, stall onset is observed at relatively low AoA under transonic conditions, highlighting the strong coupling between Mach and AoA in this regime. For this study, AoA ranges from 0 to 4 [deg] and Mach from 0.70 to 0.84 (see Figure 5.2a). By combining these multi-fidelity datasets, the MF-BayNet surrogate model is trained and fine-tuned to capture the full spectrum of aerodynamic behaviours relevant to the BSCW.

Figure 5.2b also highlights differences in aerodynamic coefficient predictions from each fidelity level, emphasizing the necessity for a model capable of effectively distinguishing and emphasizing the key features of each fidelity. The aerodynamic coefficients were calculated using a reference chord length of 0.4064 m, with C_M determined relative to 30% of the chord. Table 5.2 provides a summary of the datasets used for training the MF-BayNet surrogate model.



(A) Distribution of sample points per fidelity level.



(B) Lift and pitch moment coefficients across the design space.

FIGURE 5.2: Design variable distribution (AoA-Mach) and aerodynamic coefficient predictions for each fidelity level.

Fidelity Level	Number of Samples	Simulation Approach
Low	625	Panel Method
Mid	49	RANS - Coarse Grid
High	7	RANS - Fine Grid

TABLE 5.2: Summary of datasets used for training the MF-BayNet model.

Low-Fidelity

Low-fidelity (LF) data were generated using XFoil, a popular tool for the design and analysis of subsonic airfoils which employs a combination of inviscid panel methods with a boundary layer analysis, allowing it to rapidly generate aerodynamic data. For this study, the BSCW wing profile was used. To create a comprehensive dataset, 25 points equally distributed along each parameter of the design space were used, resulting in a total of 625 samples. This extensive dataset offers a broad base of quick, approximate aerodynamic solutions. The XFoil-generated data were corrected applying the equations from Helmbold [147] to account for three-dimensional effects, thereby improving the accuracy for the low-aspect-ratio straight BSCW configuration. These corrections ensure that the LF data better represent the actual aerodynamic behaviour of the wing in three-dimensional flow conditions, making the dataset more valuable for the multi-fidelity model training.

Mid-Fidelity

Mid-fidelity (MF) data were obtained from RANS simulations using SU2 v7.5.1 software [104] with a relatively coarse grid of $2.5 \cdot 10^6$ elements. The grid is a hybrid type, with structured elements on the wing surface and in the first layers of the boundary layer, while voxel elements were used for the rest of the computational domain. The domain itself extends 100 chord lengths from the solid wall to the farfield. The Grid Convergence Index (GCI) for this coarse grid was calculated to be approximately 5.4%, indicating a moderate level of discretisation error. The RANS equations were closed using the one-equation Spalart-Allmaras turbulence model. Convergence was monitored using the Cauchy method applied to the lift coefficient, with a variation threshold of 10^{-7} across the last 100 iterations. A $1v$ multigrid scheme was adopted to accelerate convergence. Convective flow discretisation utilized the Jameson-Schmidt-Turkel central scheme with artificial dissipation, and flow variable gradients were computed using the Green Gauss method. The biconjugate gradient stabilisation linear solver with an ILU preconditioner was selected. The samples were distributed to refine the highly nonlinear regions, particularly at high combinations of AoA and Mach number. This strategic sampling ensures that the mid-fidelity data provide enhanced resolution in critical areas, capturing the complex aerodynamic interactions more effectively.A total of 49 samples were generated, capturing viscosity and transonic flow effects better than

LF data and at a lower computational cost than HF simulations, but still with moderate discretisation errors.

High-Fidelity

HF dataset was previously generated [161], incurring no further computational cost. It consists of 58 RANS simulations with a fine grid comprising $15.6 \cdot 10^6$ elements. The GCI for this fine grid is around 0.6%, indicating a very low discretisation error and that the solution is nearly grid-independent.These HF simulations provide the most detailed resolution of transonic nonlinear features and the most accurate aerodynamic load predictions, serving as a benchmark for validating the MF-BayNet surrogate model. However, they are an order of magnitude more expensive than MF and millions of times more than LF. For fine-tuning, 7 simulations were identified as the minimum number of samples necessary to characterize the discrepancies between mid- and high-fidelity predictions, represented as red dots in Figure 5.2. These samples were not selected arbitrarily; instead, they were deliberately chosen to ensure coverage of the most critical flow conditions, emphasizing regions where transonic shock formation, boundary-layer separation, and other nonlinear effects become more pronounced. This targeted selection captures the most challenging flow regimes with the fewest simulations, thereby minimizing the cost of HF runs while preserving accuracy. This selection was informed by the authors' prior knowledge of the relevant aerodynamic phenomena [161]. The remaining samples were used as a test set to validate the model, demonstrating that with a minimum number of simulations for fine-tuning, the model can achieve a relatively low error on the entire HF dataset.

Discussion

The inherent nonlinearity of aerodynamic behaviour, particularly in the transonic regime, is a significant challenge when developing surrogate models. Figure 5.2 provides a clear illustration of how the aerodynamic coefficients, such as lift coefficient C_L and pitching moment coefficient C_M, vary nonlinearly across the M-AoA design space. The LF data, generated through XFoil, shows a more linear response, failing to capture the abrupt changes and complex phenomena like shock-wave formation and boundary-layer separation that occur at higher Mach numbers and AoA. This limitation necessitates the incorporation of higher-fidelity data to correct for these deficiencies. The mid-fidelity RANS simulations, with their ability to better resolve the flow nonlinear characteristics, particularly near the critical Mach number, offer a significant improvement. However, it is the HF data that most accurately captures the sharp gradients and nonlinearities essential for predicting aerodynamic loads in the transonic regime.

These differences across fidelity levels underscore the importance of a multi-fidelity approach. By integrating data from different fidelities, the MF-BayNet surrogate model more effectively captures and predicts the complex nonlinear dynamics of transonic aerodynamics.

5.3.2 Results

Here, the performance of the MF-BayNet surrogate model, three single-fidelity dataset and the CK method are evaluated by comparing their predictions of aerodynamic coefficients against an independent HF test dataset made of 51 CFD samples, which was excluded from both the HF model training and multi-fidelity training. Specifically, each single-fidelity dataset was partitioned into 70% for training and 30% for validation. The same partitioning was used to train MF-BayNet during TL process. Dataset sizes are shown in Table 5.3. Further, the MF-BayNet surrogate model uncertainty estimates are compared to CK method, highlighting its superior ability to capture and quantify prediction uncertainty. Results also include comparisons with BNNs trained on each dataset individually without TL, with model architectures optimised.

Fidelity Level	Train Samples	Validation Samples	Test Samples
Low	437	188	-
Mid	33	15	-
High	5	2	51

TABLE 5.3: Number of samples used for training, validation, and testing across the three fidelity levels for the finite wing test case. HF test samples (51 in total) are used exclusively for final evaluation and are not included in training or validation.

Comparison of Models Performance

Table 5.4 compares the performance of MF-BayNet with CK in predicting C_L and C_M. It also includes the results of the network trained individually on each dataset without utilizing TL, as well as results from the MF-BayNet framework where LF samples were excluded from training. For the latter, the model was first trained on MF samples and then TL was applied to fine-tune the model on HF samples. All results are averaged over a k-fold validation with a k value of 5.

The results demonstrate that MF-BayNet significantly outperforms all other models for both C_L and C_M. Specifically, MF-BayNet achieves a $\varepsilon_{\mu_{CL}}$ of 4.2% and a $\varepsilon_{\mu_{CM}}$ of 5.5%, which are approximately half of those obtained with CK, which has errors of 7.5% and 7.7% for C_L and C_M respectively. Moreover, the standard deviations errors ($\varepsilon_{\sigma_{CL}}$ and $\varepsilon_{\sigma_{CM}}$) for MF-BayNet are also lower, indicating more consistent and reliable predictions. The BNN models, whether trained on LF, MF or HF datasets, show higher prediction errors and standard deviations, reinforcing the efficacy of the TL and fine-tuning processes utilized by MF-BayNet. Finally, these results confirm the added value of combining datasets across all fidelity levels; removing LF samples from the MF-BayNet training, as well as removing HF samples, slightly increased prediction errors, indicating that lower fidelity data still contribute information that enhances generalisation capability and HF further enhances preliminary training through the fine-tuning process.

Model Name	$\varepsilon_{\mu_{CL}}$	$\varepsilon_{\sigma_{CL}}$	$\varepsilon_{\mu_{CM}}$	$\varepsilon_{\sigma_{CM}}$	ε_{tot}
BNN LF	14.43	1.59	42.02	1.41	31.42
BNN MF	12.63	6.88	7.73	4.92	10.47
BNN HF	15.29	9.00	5.62	5.96	11.52
CK	7.53	2.98	7.70	9.46	7.61
MF-BayNet (LF+MF)	7.55	9.04	8.42	5.77	8.00
MF-BayNet (MF+HF)	5.81	6.95	6.48	4.44	6.15
MF-BayNet (LF+MF+HF)	**4.24**	1.37	**5.50**	0.71	**4.91**

TABLE 5.4: Performance metrics in percentage [%] for different models for the finite wing test case. Errors are calculated on HF test set.

To better illustrate model performance, Figures 5.3 show predictions of C_L and C_M at different Mach numbers. The left panels present predictions at $M = 0.74$, while the right panels show predictions at $M = 0.82$. For C_L predictions, the MF-BayNet surrogate model (orange line) closely follows the HF dataset (red dots), effectively leveraging knowledge acquired through transfer learning and fine-tuning. The MF-BayNet predictions exhibit narrow confidence intervals (brown shaded regions), indicating high predictive certainty. In contrast, the CK model (dark green line) struggles to capture the correct trend, especially at higher AoA and Mach numbers, and shows significantly larger confidence intervals (green shaded regions, particularly noticeable in the right panels). The nonlinear nature of the aerodynamic behaviour is particularly evident at $M = 0.82$, where the C_L curve shows significant nonlinearity, especially at higher angles of attack. This nonlinearity arises from the formation of shock waves on the wing surface, which induce abrupt changes in the pressure distribution, leading to sharp variations in C_L. The MF-BayNet surrogate model successfully captures these complex behaviours, unlike the CK model, which struggles more in representing these nonlinear effects. Similarly, the C_M predictions reveal pronounced nonlinear trends as the aerodynamic centre shifts due to interactions between shock waves and the boundary layer. These interactions complicate the aerodynamic response, causing rapid changes in C_M. The MF-BayNet surrogate model ability to closely match the HF data, along with its narrower confidence intervals, highlights its strength in modelling these challenging nonlinear phenomena. Similar considerations apply to Figure 5.4, where the predictions of C_L and C_M are shown with fixed AoA values and varying Mach. It can be seen that the CK model occasionally produces predictions closer to the HF data points, but still fails to capture the more complex trends indicated by the MF and HF datasets. Instead, it continues to align more closely with the lower fidelity dataset trend. Examination of both figures reveals that the observed nonlinearities arise from the combined influence of Mach number and AoA variations, with the MF-Baynet providing good accuracy over the entire parameter space, under both small and large nonlinearity ranges.

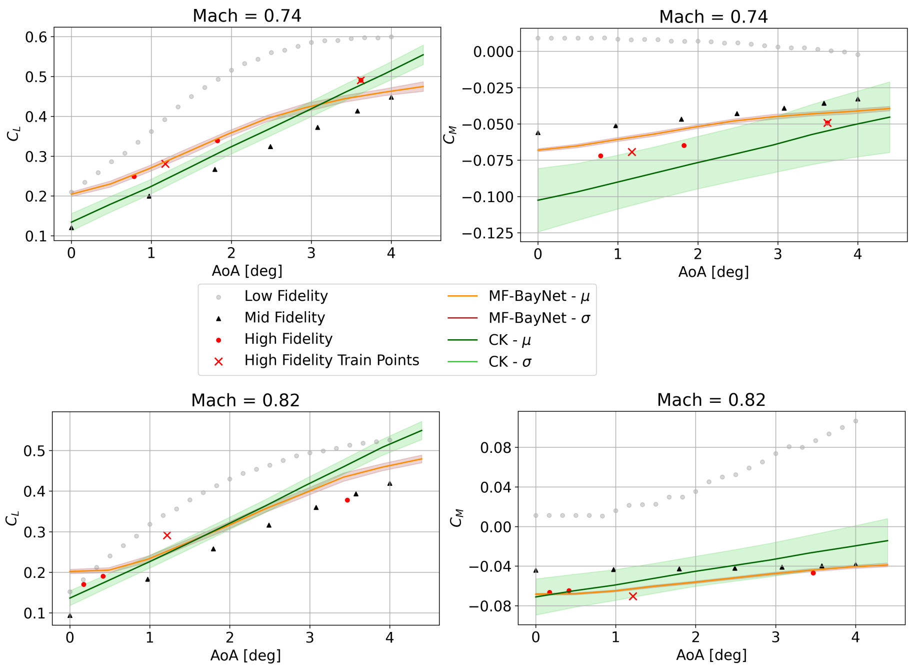


FIGURE 5.3: Comparison of MF-BayNet and CK predictions for C_L and C_M at two Mach numbers ($M = 0.74$, left panel; $M = 0.82$, right panel) across varying AoA for the finite wing test case. Points from different fidelity datasets have been interpolated for 2D plot visualization.

In summary, the single-fidelity model trained on LF data failed to capture the nonlinear transonic aerodynamics due to the oversimplified nature of the underlying data. Including MF data improved predictions by resolving more complex flow features, though discretisation errors still constrained generalisation. The HF-only model achieved low accuracy and poor extrapolation due to the limited number of samples. In contrast, the proposed multi-fidelity framework, which integrates LF, MF, and HF data through TL, achieved superior accuracy and generalisation, outperforming both single-fidelity models and CK. While CK benefited from combining data sources, it underperformed in highly nonlinear regions and provided less reliable uncertainty estimates.

Mid-Fidelity Dataset Size Impact

Since MF data strikes a good balance - being significantly less computationally expensive than HF data, yet providing much richer information compared to LF data - we

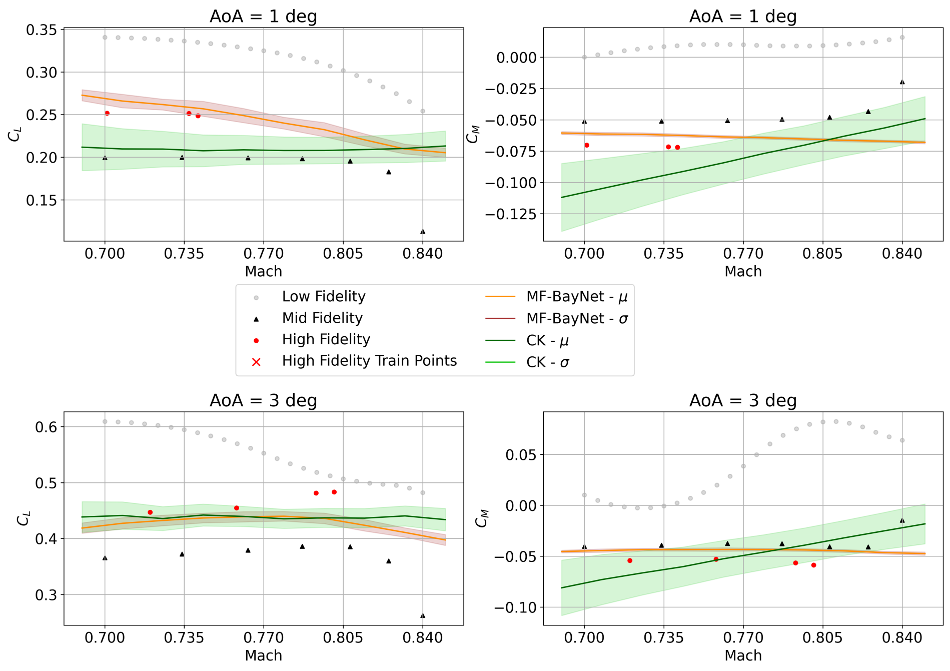


FIGURE 5.4: Comparison of MF-BayNet and CK predictions for C_L and C_M at two AoA values ($AoA = 1.0$ [deg], left panel; $AoA = 3.0$ [deg], right panel) across varying Mach numbers for the finite wing test case. Points from different fidelity datasets have been interpolated for 2D plot visualization.

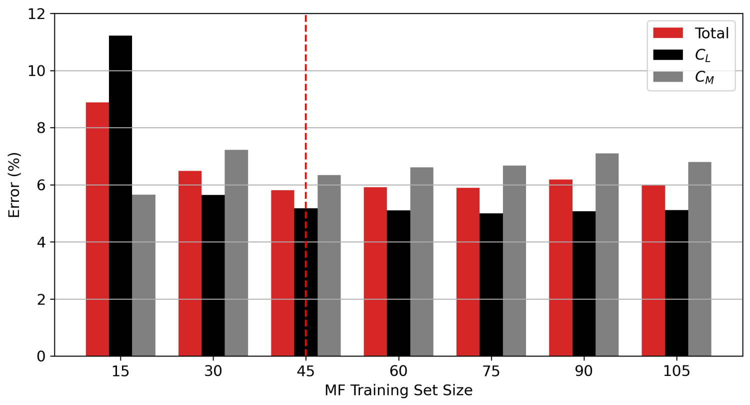


FIGURE 5.5: Influence of MF training set size on model performance for the finite wing test case.

investigated the impact of MF sampling on model performance. To perform this sensitivity analysis, we first generated a denser set of MF points. The original sample consisted of 49 points arranged in a 7×7 grid, while the newly generated points formed a denser 13×13 grid of 169 points. Model performance was evaluated incrementally as the dataset size increased, using 5-fold cross-validation, with each fold divided into 70% training and 30% testing subsets.

In general, increasing the number of MF samples leads to a reduction in prediction error. This behaviour is expected, as more MF points allow the model to better capture the underlying physics. However, as shown in Figure 5.5, the improvement quickly reaches a saturation point where further increases yield minimal or no additional benefit, reflecting the inherent limitations of the simplified representation provided by MF data compared to HF data. In our analysis, performance gains become negligible after approximately 50 MF samples.

Aleatoric Uncertainty Quantification

To assess the impact of aleatoric uncertainty on the predictions, we re-trained the baseline version of our MF-BayNet model—termed the ”Vanilla MF-BayNet”—by systematically adding noise to each fidelity dataset separately. The training dataset was augmented following an augmentation factor value (AF) from 20% to 80%, by adding Gaussian noise with a standard deviation from 2% to 10% of the mean output label value, applied to either C_L (Figure 5.6) or C_M (Figure 5.7).

Results show that ε_μ generally decreases with increasing noise when added to LF and HF datasets for C_L, whereas the MF datasets exhibit an increasing trend in mean error. In contrast, for C_M, the mean error increases with noise in both LF and MF datasets, but adding noise to HF datasets tends to reduce the mean error, especially as AF increases. Across both C_L and C_M, ε_σ remains low and stable, indicating that noise primarily impacts the mean error rather than the variability of model predictions.

As the AF increases in MF datasets, the impact of noise on the mean error becomes more pronounced, while higher AF generally show decreased mean errors in HF datasets. This behaviour can be explained by considering the robustness of HF data to noise and the sensitivity of MF data. HF datasets contain more accurate and detailed information, making the model less sensitive to noise and, in some cases, benefiting from noise through regularisation, which improves the model generalisation and reduces the mean error [49]. Adding noise to MF datasets results in a significant increase in mean error. This higher sensitivity to noise is due to the moderate levels of accuracy and detail in MF datasets, which are significantly degraded by the introduction of noise. The balance between the inherent detail and the added noise disrupts the model learning process, leading to increased errors. LF datasets, already less accurate and detailed, exhibit a mixed response to noise. In some cases, the noise further degrades data quality, increasing the mean error, while in others, it has minimal impact or even aids the model by introducing beneficial variability.

The relatively minor changes in the standard deviation of errors, despite varying noise levels, suggest that the model overall prediction uncertainty remains stable. This indicates that while noise impacts the accuracy (mean error) of predictions, it does not significantly affect the model confidence in its predictions. These observations underscore the importance of understanding and mitigating aleatoric uncertainty to improve predictive modelling performance, particularly by leveraging the robustness of HF data and addressing the sensitivity of MF data.

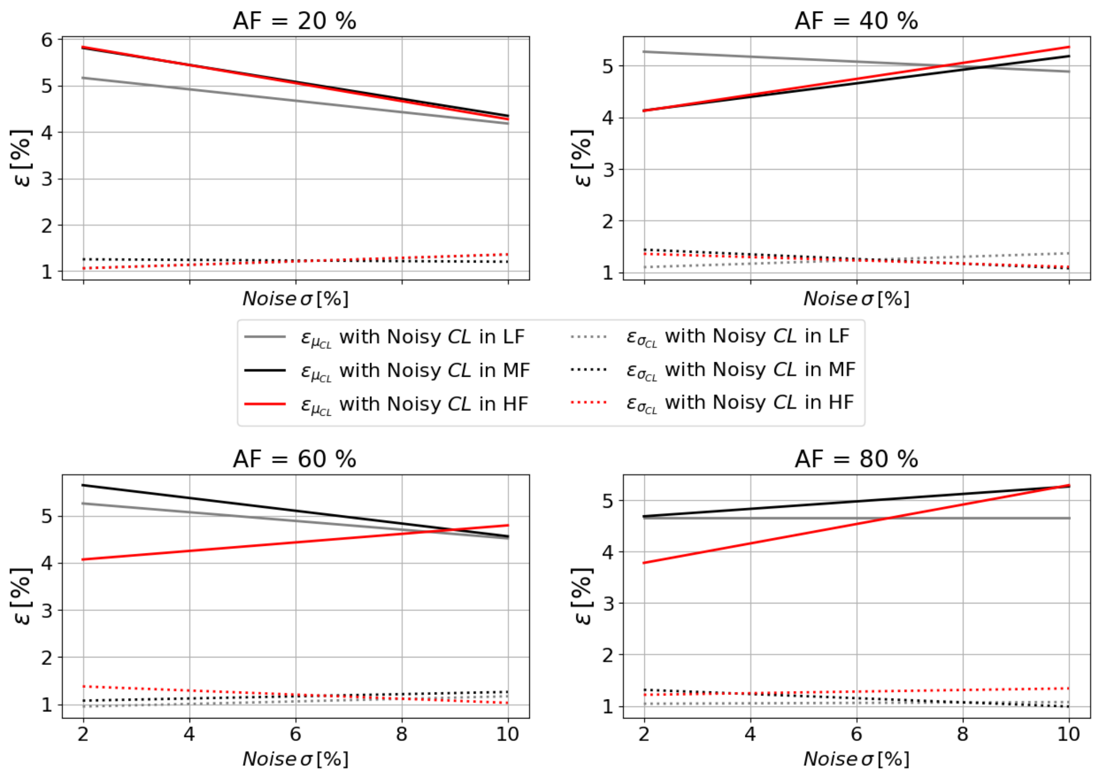


FIGURE 5.6: Impact of aleatoric uncertainty on the model predictions, in terms of error on the standard deviation and mean value of the C_L. Training dataset augmented from 20% to 80% by adding Gaussian noise to the C_L of each dataset separately with a standard deviation ranging from 2% to 10%.

We also studied the effect of aleatoric uncertainty on model predictions when adding Gaussian noise to the entire LF dataset ($AF = 100\%$) (refer to Figure 5.8). The noise standard deviation ranged from 2% to 10%, applied independently to the coefficients C_L and C_M. The graph depicts how the introduction of this noise influences the error associated with these coefficients separately. It is evident that as the noise standard deviation increases, the error in the predictions also increases. This trend is consistent for both C_L and C_M. This indicates that the predictive capability of the model is increasingly compromised as more noise is introduced. However, the standard deviation remains relatively stable for both coefficients, suggesting that while accuracy decreases, the consistency of the model predictions does not fluctuate significantly.

This behaviour can be explained by the structure of our MF-BayNet surrogate model. Adding noise to the entire LF dataset increases the initial errors, which propagate through the subsequent training stages. The model learns from this noisier data initially, which degrades its baseline understanding and leads to higher errors that carry

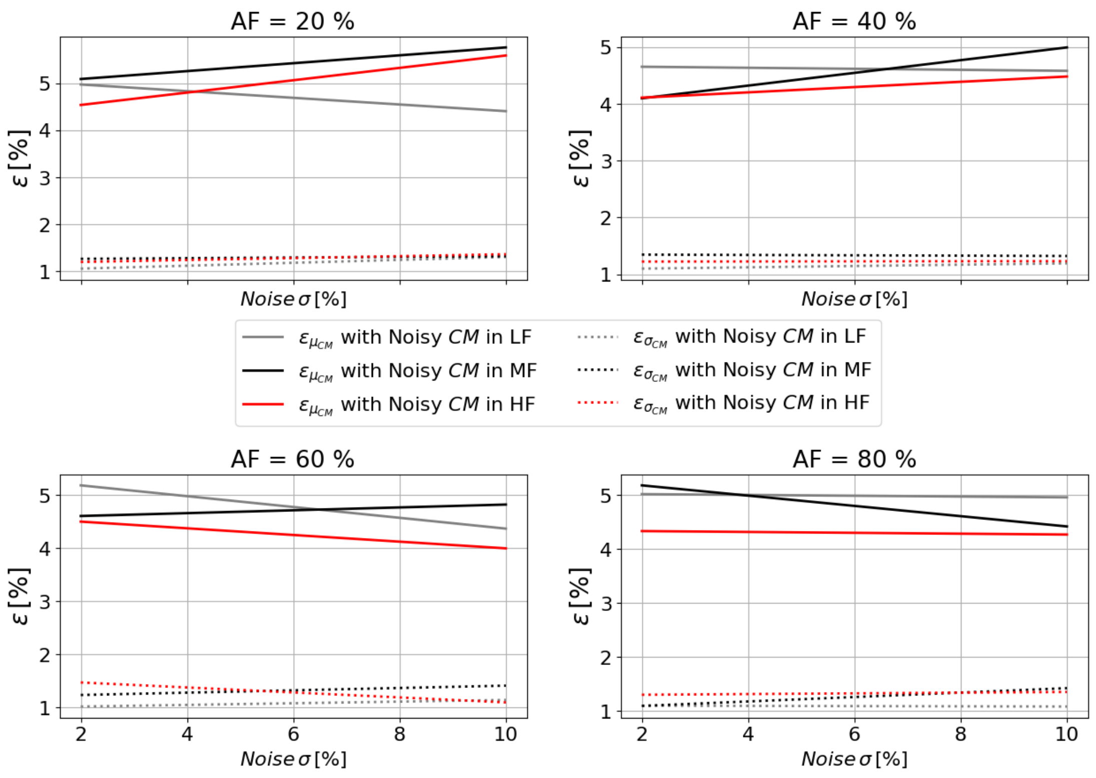


FIGURE 5.7: Impact of aleatoric uncertainty on the model predictions, in terms of error on the standard deviation and mean value of the C_M. Training dataset augmented from 20% to 80% by adding Gaussian noise to the C_M of each dataset separately with a standard deviation ranging from 2% to 10%.

over even after TL processes. Moreover, the stable standard deviation suggests that the noise primarily affects the bias (mean error) rather than the variance of the predictions. This indicates that while the model accuracy decreases due to systematic errors introduced by the noise, its confidence in the consistency of predictions remains unaffected. Therefore, enhancing data quality or employing robust modelling techniques to mitigate the impact of noise can lead to better accuracy without compromising prediction consistency.

A natural extension of the present analysis would be an explicit investigation of epistemic noise, which arises from model uncertainty rather than measurement perturbations. While this work has focused on aleatoric effects introduced through Gaussian perturbations of the aerodynamic responses, future work could explore epistemic contributions by varying the coverage and density of the training data across the parameter space or adjusting the variance of the Gaussian priors placed on the network weights.

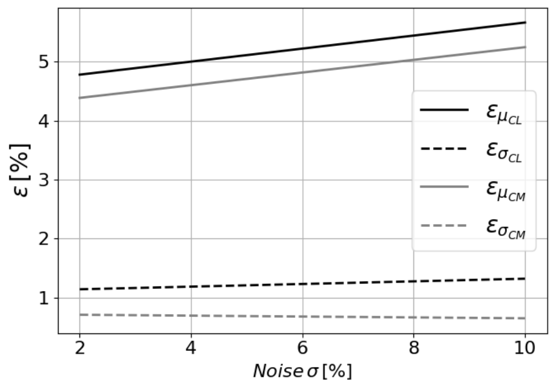


FIGURE 5.8: Effect of aleatoric uncertainty on model predictions: Gaussian noise, with a standard deviation ranging from 2% to 10%, was independently added to the C_L and C_M of the entire LF dataset.

5.4 Full-configuration Test Case

This section presents a five-propeller electric vertical takeoff and landing (eVTOL) test case that illustrates the critical aerodynamic challenges associated with rotor-wing interactions in a real scenario. The test case combines two different fidelity datasets and it is inspired by the modelling exercise carried out by the authors in the EASA funded project MODEL-SI [261]. The vehicle under consideration has four longitudinally symmetric lift propellers mounted forward and aft of the wing, and one pusher propeller mounted at the tail for forward flight. Thanks to this configuration, the drone can switch between a so-called "helicopter mode" and an "airplane mode" depending on the use of the propellers. The fuselage is 4 m long, while the wingspan is 6 m with a mean chord of 1 m of the NACA0012 profile. An impression of the configuration can be obtained from Figure 5.9.

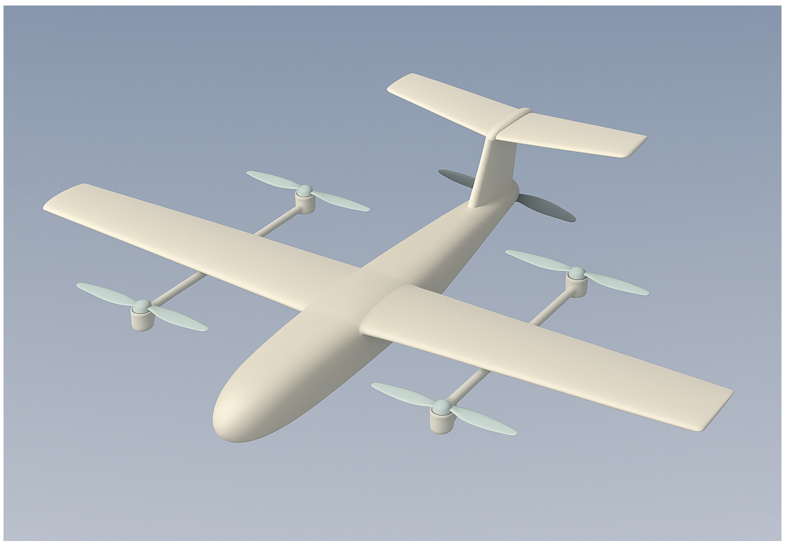


FIGURE 5.9: Impression of eVTOL configuration.

The primary aerodynamic challenge is the complex interaction between the propellers, wing, and fuselage. Forward propellers generate rotor wakes that flow over the wing and impinge directly on the aft propellers, significantly altering the local flowfield. This perturbation changes the lift distribution, adds extra wing loading, and increases drag. Also, the rear propellers suffer a decrease in thrust when operating in the wake of the front rotors for the same power input, resulting in spatially varying performance that is difficult to predict with lower fidelity methods alone. Once the wing itself generates lift, its near-field flow curvature affects the inflow conditions of the forward rotors. Depending on how far forward of the wing the propellers are located, the swirling flow can reduce rotor efficiency if the incoming flow is distorted or partially blocked. These phenomena occur over a wide flight envelope, from vertical takeoff and forward flight to the transition between helicopter and airplane mode, making the problem highly nonlinear.

To capture the most important physical parameters that affect the rotor-wing aerodynamics of the eVTOL vehicle, the inputs include AoA, freestream velocity U_∞, and rotational speeds of five propellers: pusher (`PP`), front right (`FR`), front left (`FL`), rear right (`RR`), and rear left (`RL`). The multi-fidelity framework predicts two outputs for each of the five propellers: thrust coefficient (K_T) and torque coefficient (K_Q). Specifically, the model produces $K_{T_\text{PP}}, K_{Q_\text{PP}}$ for the pusher propeller and $K_{T_\text{FR}}, K_{Q_\text{FR}}, K_{T_\text{FL}}, K_{Q_\text{FL}}, K_{T_\text{RR}}, K_{Q_\text{RR}}, K_{T_\text{RL}}, K_{Q_\text{RL}}$ for the four lift propellers.

The MF-BayNet surrogate model was trained using the TL framework, sequentially incorporating LF and MF datasets. The optimized hyperparameters used in this test case are reported in A.3.2.

5.4.1 Multi-Fidelity Datasets

The strongly coupled and nonlinear flow physics in this eVTOL configuration provide another representative test case for the surrogate aerodynamic model presented. Although HF simulations (e.g., RANS-based CFD) can offer greater accuracy, they become prohibitively expensive—and often difficult to converge—when exploring wide flight envelopes. Indeed, in this study, fully converged CFD data could not be generated due to excessive computational costs. By contrast, lower-fidelity simulations are more affordable yet struggle to capture the pronounced rotor-wing interactions and wake impingement effects. Consequently, a multi-fidelity approach provides a more balanced solution: it integrates fast, lower-fidelity analyses with higher-order corrections from MF data, ensuring that complex flow interactions are accurately captured without incurring prohibitive computing time.

To generate samples, we covered a wide range of operating conditions: AoA varies from -180 to 180 [deg], U_∞ from 0 to 40 [m/s], and each propeller rotational speed up

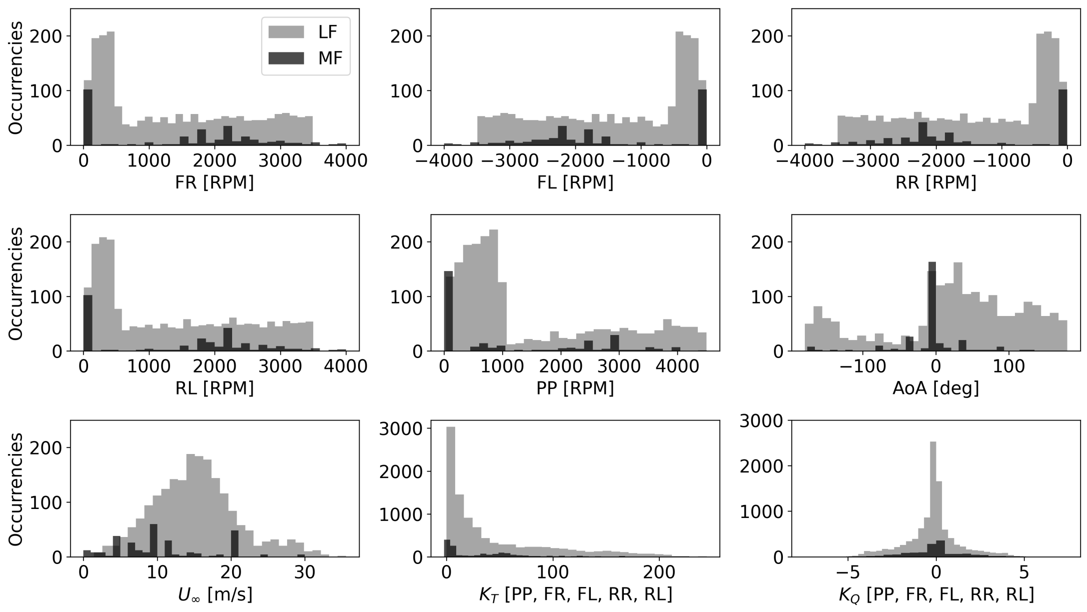


FIGURE 5.10: Histograms of LF and MF datasets distribution across the design space. K_T and K_Q values are flatten for all propellers PP, FR, FL, RR, RL.

to ±4000 [RPM], depending on its position and direction of rotation. These ranges were chosen based on aerodynamic and flight mechanics considerations to ensure that the dataset covers realistic manoeuvrers and bounds of the flight envelope.

LHS strategy was used to generate the LF dataset, resulting in approximately 2000 data points covering the broad ranges described above. These LF points were derived from an analytical aerodynamic function, allowing efficient data generation over a wide parameter space. In contrast, a limited set of MF samples were obtained from more accurate but computationally expensive simulations, resulting in 250 points distributed like in Figure 5.10. This smaller MF set refines and corrects the trends observed in the LF data. Table 5.5 schematically illustrates how both LF and MF data form the training dataset.

Fidelity Level	Number of Samples	Simulation Approach
Low	2,000	BEM
Mid	250	VPM

TABLE 5.5: Summary of datasets used for training the MF-BayNet surrogate model.

Low-Fidelity

The LF aerodynamic model is inspired to the work by Davoudi [86] in which Momentum Theory and Blade Element Method (BEM) are exploited to assess induced velocity and thrust generated by a moving rotor. Momentum theory provides a global analysis of flow through an actuator disk, deriving thrust and power relationships from force and flow balances. BEM refines this approach by discretizing the rotor blade into spanwise segments, each analyzed independently using 2D sectional aerodynamics. These models are most commonly used for aeroacoustic studies [257] to develop flight mechanics models of drones, capturing variations in AoA, chord, and twist along the blade span at low computational cost. In essence, the induced velocity model is inspired by Peter's model [265] and represents a simple yet sufficiently accurate approach for most engineering applications, providing a valuable starting point for rapid iterative analysis in preliminary design.

Mid-Fidelity

For the MF aerodynamic simulations, we employed the DUST framework [352]. It is designed to analyze the aerodynamics of non-conventional aircraft configurations at a level of detail that bridges low- and high-fidelity approaches by implementing a Vortex Particle Method (VPM). Compared to purely potential-based tools or blade-element methods, DUST offers superior accuracy in capturing three-dimensional effects while remaining considerably less expensive than full RANS-based simulations. In the model setup, care was taken to model rotors and propeller blades with the accurate chord and twist distributions. The results shown in this paper were obtained with the option called "nonlinear lifting line" in the user manual [305]. We selected between 30 and 40 lifting lines per blade, the aerodynamic coefficients were given by look-up tables obtained from CFD for a number of blade sections. The maximum number of vortex particles was set to 100 million. The wing surface is discretized using approximately 3,000 panels for each side.

5.4.2 Results

The eVTOL test case presents a more challenging aerodynamic scenario than the finite wing test case, due to strong rotor-rotor and rotor-wing interactions. Such interactions introduce significant nonlinearities, wake impingement effects, and asymmetric flowfields, thereby increasing the difficulty of accurately capturing the physics across the design space. Consequently, this test case provides a high-dimensional benchmark for evaluating the generalizability of DF models.

Following the previous test case approach, the dataset is partitioned into training, validation, and test sets as detailed in Table 5.6. The MF test set, which is completely excluded from the training process, is used to compare the prediction accuracy of both MF-BayNet and CK. The comparison focuses on both mean prediction error and uncertainty quantification performance, with particular attention to the ability of each model to capture the variability introduced by complex aerodynamic couplings.

Fidelity Level	**Train Samples**	**Validation Samples**	**Test Samples**
Low	1,400	600	-
Mid	605	130	130

TABLE 5.6: Number of samples used for training, validation, and testing across the two fidelity levels for the full-configuration test case.

Table 5.7 shows the results for K_T and K_Q for the two different models. For K_T, the two models have comparable performance (3.9% vs. 2.9%), indicating that both methods adequately capture the basic rotor thrust variations under different flight conditions. However, for K_Q, MF-BayNet achieves a lower average error (3.1%) compared to CK ($\sim$5%). The torque coefficient is more sensitive to local flow features (e.g., rotor wake distortion or wing-propeller flow interference), and MF-BayNet's TL structure appears to be better able to leverage the MF data to capture these interaction effects.

Model Name	ε_{K_T}	ε_{K_Q}	ε_{tot}	σ_{tot}
CK	3.93	4.98	4.50	0.26
MF-BayNet	2.99	3.12	3.06	0.87

TABLE 5.7: Averaged thrust and torque coefficients percentage errors [%] for different models for the full-configuration test case. Errors are calculated respect to MF test set.

Figure 5.11 illustrates how the two DF approaches predict the K_T and K_Q values of the pusher propeller (PP) under "airplane mode" conditions, namely at $AoA = -5.0$ [deg], $U_\infty = 10$ [m/s], with the lift propellers off (0 [RPM]). Compared to CK, MF-BayNet follows the reference data more accurately, especially at medium to high RPM values, and captures the shift in aerodynamic behaviour. In contrast, the CK prediction deviates significantly from the MF samples at higher speeds, reflecting a less effective DF for strong nonlinear rotor-propeller interactions. This highlights the benefit of the MF-BayNet framework, which better exploits the MF data to refine and correct the predictions in challenging flow regimes.

5.5 Computational Cost Analysis

5.5.1 Finite Wing Test Case

Tables 5.8 and 5.9 provide a comparative analysis of the computational costs associated with dataset creation and model performance.

Table 5.8 highlights the substantial disparity in computational effort required across different fidelity levels. The HF dataset requires approximately 500 CPU hours per sample, making it over 10 times more expensive than the MF dataset and nearly a million times more computationally intensive than the LF dataset. This underscores the

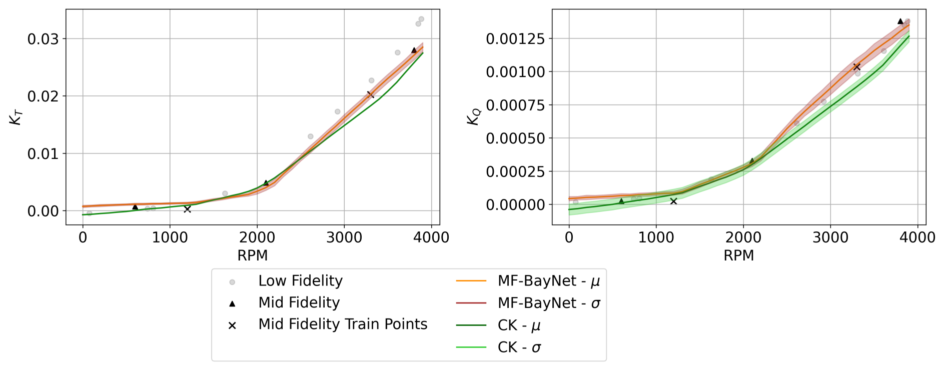


FIGURE 5.11: Comparison of MF-BayNet and CK predictions in Airplane Mode for K_T and K_Q using ($AoA = -5.0$ [deg], $U_\infty = 10$ [m/s], and all the lift propellers set to 0 [RPM]) across varying RPM values for the pusher propeller (PP) in the full-configuration test case. Points from different fidelity datasets have been interpolated for 2D plot visualization.

impracticality of relying exclusively on HF simulations for aerodynamic load predictions, reinforcing the necessity of multi-fidelity modelling. By integrating lower-fidelity data, the MF-BayNet framework effectively mitigates the reliance on computationally expensive HF simulations while maintaining a high level of predictive accuracy.

Dataset	**Run time**	
	(Full dataset)	(1 run)
Panel Method (LF)	0.3475 (625 runs)	0.00056
RANS - Coarse Grid (MF)	2,352 (49 runs)	48
RANS - Fine Grid (HF)	29,000 (58 runs)	500

TABLE 5.8: CPU hours for different simulation runs for the dataset creation process of the finite wing test case.

Table 5.9 presents the GPU hours required for training and prediction across different models using an NVIDIA Quadro P2000 GPU, with execution times averaged over one dataset fold and results calculated using 5-fold cross-validation (k=5). For single-fidelity datasets, the average training times were approximately 41 minutes for the LF model, 35 minutes for the MF model, and less than 5 minutes for the HF model. The MF-BayNet required approximately 58 minutes for training. In comparison, CK took about 5 minutes to fit from LF data using HF points as an additional feature to perform CK. Prediction times across all BNN models were consistent, averaging around 0.025 seconds to compute both mean and standard deviation values through MCS with 100 predictions. Conversely, the CK model demonstrated faster prediction times, requiring only about 0.005 seconds to generate mean and standard deviation predictions due to the simpler architecture and structure.

Model	Training	Prediction
BNN LF	0.68	7.0e-6
BNN MF	0.58	7.0e-6
BNN HF	0.08	7.0e-6
MF-BayNet	0.97	7.0e-6
CK	0.08	1.4e-6

TABLE 5.9: GPU hours for training and prediction costs for the proposed models for the finite wing test case.

5.5.2 Full-configuration Test Case

In Table 5.10 we report the total CPU hours required to generate the LF and MF datasets for the eVTOL test case. As shown, the analytic function (LF) approach is extremely efficient, requiring only 0.2 CPU hours for the entire dataset of 2000 runs (i.e., 0.0001 CPU hours per run). In contrast, the DUST (MF) simulations, while providing higher accuracy due to a more sophisticated modelling of the rotor-wing interaction, are significantly more expensive, amounting to 130 CPU hours per run or a total of 7,800 CPU hours for 60 runs.

Dataset	Run time	
	(Full dataset)	(1 run)
BEM (LF)	0.2 (2000 runs)	0.0001
DUST (MF)	7,800 (60 runs)	130

TABLE 5.10: CPU hours for different simulation runs for the dataset creation process of the full-configuration test case.

Table 5.11 shows the GPU hours required for training and prediction for different models using the same procedure as for the previous test case. MF-BayNet required approximately 60 minutes for training, while CK completed training in about 5 minutes. For prediction, MF-BayNet consistently took around 0.025 seconds per sample using MCS with 100 forward passes, whereas CK required only about 0.005 seconds.

Model	Training	Prediction
MF-BayNet	1	7.7e-6
CK	0.08	1.4e-6

TABLE 5.11: GPU hours for training and prediction costs for the proposed models for the full-configuration test case.

5.6 Conclusions

We developed a multi-fidelity framework using Bayesian neural networks and transfer learning (MF-Baynet) to enhance the accuracy of aerodynamic load prediction in the transonic regime over a three-dimensional wing and to reduce the required HF sample for training the surrogate model. Our primary objective was to leverage the strengths of BNNs, notably their ability to quantify uncertainty, alongside the efficiency and robustness of TL to improve predictive performance across datasets of different accuracy. The resulting surrogate model effectively integrates data of varying fidelities, harnessing the strengths of each fidelity level to produce superior predictive performance and robust uncertainty quantification.

The results obtained on two different test cases–a transonic wing and a full-configuration eVTOL vehicle–demonstrate that the MF-BayNet surrogate model not only surpasses traditional models trained on single-fidelity datasets but also outperforms CK method in predicting integrated aerodynamic loads. These results demonstrate the effectiveness of our hybrid approach in achieving superior accuracy and reliability, especially in complex aerodynamic scenarios where nonlinearity plays a critical role.

Confidence intervals calculated by MF-BayNet model provide a quantitative measure of epistemic and aleatoric uncertainty, offering valuable insights for safety-critical applications. In transonic regimes, where flow behaviour is highly nonlinear and sensitive to small perturbations, such predictive uncertainty becomes important for identifying high-risk regions in the design space. For instance, a wing geometry associated with high uncertainty under specific conditions may require modification or local reinforcement, even in the absence of mean load exceedances. Furthermore, the ability to quantify and localise predictive confidence is particularly relevant for certification processes, where regulators increasingly demand rigorous uncertainty quantification to ensure robustness and compliance across a broad range of operating conditions. Thus, confidence intervals are not merely statistical diagnostics; they are actionable indicators that inform decision-making in both early-stage design and formal validation.

The study also revealed that the impact of aleatoric uncertainty on model predictions varies significantly with the fidelity level of the datasets to which noise is added. Adding noise generally increases the mean error with noisy MF datasets while often decreasing the mean error with noisy HF datasets, particularly as the augmentation factor increases. Noisy LF datasets show a mixed response, with slight increases or decreases in mean error. The standard deviation of the error remains relatively stable across different noise levels, indicating that noise primarily affects the accuracy of predictions rather than the variability. These findings highlight the robustness of HF data to noise and the sensitivity of MF data, suggesting that while noise can regularise and improve model performance with HF data, it can degrade performance with MF data.

Additionally, we observed that adding noise to the entire LF dataset increases the initial errors, which propagates through the subsequent training stages, degrading overall accuracy. This is because the model, which first trains on the LF data, carries forward these systematic errors introduced by the noise even after TL processes. Consequently, while the mean error rises due to these systematic errors, the prediction consistency, as indicated by the stable standard deviation, remains unaffected. Therefore, understanding and addressing these subtleties is extremely important for enhancing predictive performance and mitigating the effects of aleatoric uncertainty.

By integrating these insights into the broader thesis objectives, this chapter substantially contributes to:

1. **Multi-Fidelity Nonlinear Aerodynamic Modelling**
   MF-BayNet reduces the dependence on extensive and expensive HF data. Its fusion of LF, MF, and HF datasets leads to more accurate and computationally feasible models, consistent with the thesis goal of constructing efficient surrogate methods without sacrificing predictive accuracy in nonlinear flight conditions.
2. **Uncertainty Quantification and Robust Design**
   The Bayesian approach provides plausible confidence intervals, thus addressing the thesis goal of robust UQ (Objective O2 of the thesis). Analyses of noise injection at different fidelity levels provide guidelines for balancing regularisation benefits with potential performance degradation, thus informing more pragmatic high-fidelity sample generation.
3. **Flexibility in Test Cases**
   The demonstrated ability to apply MF-BayNet to complex aircraft configurations and transonic flows underscores its adaptability and sets the stage for extensions to multi-physics or multi-point optimisation.

Several avenues remain open for improvement. The proposed framework incorporates epistemic and aleatoric components, but accurately distinguishing their respective contributions remains challenging; the two effects are inevitably interconnected in the learned weight distributions and cannot be easily separated without additional, more advanced techniques [36]. Furthermore, the present implementation assigns identical priors to all fidelity levels and merely ‘freezes’ the number of hidden layers to regularise high-fidelity weights; more nuanced weighting schemes, capable of explicitly privileging data sources according to their credibility or information content, have not yet been implemented and would require a tailored hierarchical prior or fidelity-aware kernel. Finally, the current study is restricted to steady-state operating conditions: the network infers aerodynamic integral quantities but offers no explicit representation of temporal correlations, precluding its direct application to unsteady manoeuvres or aeroelastic transients that dominate many practical design scenarios.

Statement of Contributions

Andrea Vaiuso contributed to the implementation of the Python scripts used for the MF-BayNet model. Oier Coretti generated the dataset for the full-configuration test case.

Chapter 6

Deep Learning for Steady Transonic Flowfields

This chapter presents a deep learning framework for predicting steady-state transonic vector fields across complex 3D geometries, supporting Research Objective O3. Accurate flowfield modelling is essential for computing aerodynamic loads and conducting aeroelastic analyses. The proposed ROM uses freestream Mach number and angle of attack as key independent variables governing the transonic flow. A comparison with a POD-based method highlights the strengths and limitations of the deep learning approach. Its accuracy and generalisation are validated on three benchmark 3D cases: the BSCW, the ONERA M6 wing, and the CRM wing-body configuration. Despite the nonlinear nature of transonic flows, the neural network delivers strong predictive performance throughout. Beyond steady-state prediction, the framework also supports aeroelastic analysis and uncertainty quantification, contributing to Research Objective O2. These additional capabilities highlight the robustness and adaptability of the approach. Overall, the method provides a computationally efficient surrogate to high-fidelity CFD, offering an attractive alternative for preliminary design and parametric exploration in transonic aerodynamic regimes.

6.1 Introduction

Numerical simulations based on CFD are a key part of modern aerodynamic analysis, but remain computationally intensive, often requiring millions of degrees of freedom and extensive high-performance computing. ROMs offer an efficient alternative by compressing high-dimensional flow physics into significantly fewer degrees of freedom while preserving key aerodynamic features. This dimensionality reduction accelerates design iterations, supports real-time flow predictions, and simplifies the interpretation of complex flow structures. Traditional ROMs, such as POD-based methods, project high-dimensional manifolds onto lower-dimensional subspaces and have improved our understanding of flow physics at reduced cost [194, 219].

However, purely linear ROMs can struggle with strongly nonlinear effects (e.g., shocks, viscous phenomena) unless they are heavily tuned or expanded with numerous modes. Although POD, Isomap, and manifold learning have shown promise in predicting steady turbulent flows [116, 115, 289, 415, 416], their simplified formulations limit their accuracy in reproducing abrupt changes in the pressure field caused by shocks.

These limitations resulted in considerable interest in novel, nonlinear ROM strategies, such as deep learning approaches, which introduce nonlinearity through their activation functions. Deep learning has demonstrated impressive success in a number of fluid dynamics applications [41, 47, 119, 189, 337]. In particular, FCNN architecture has been used to predict the dynamic response of nonlinear aerodynamic systems in both subsonic [347] and transonic regimes [330], as well as for aerodynamic load estimation during dynamic stall [367]. Dense network architectures also serve effectively as surrogate models for aircraft design and optimisation [297, 402]. Convolutional models have been widely used for field prediction and surrogate CFD, from early steady flow surrogates on airfoils and bluff bodies [132, 31] to turbulence-aware closures that embed invariance directly in the network architecture [213]. Physics-informed approaches incorporate Navier-Stokes residuals to regularise learning and reduce data demands [226]. For complex geometries, graph-based simulators and mesh graph networks enable message-passing on meshes and particles, improving stability and generalisation in spatio-temporal prediction [302, 267].

Recent hybrid strategies combine established techniques such as POD with neural networks. Zhiwei et al.[411] used POD and domain decomposition to handle transonic flows, using NNs to interpolate POD modes outside the training domain. Du et al.[99] and Espinosa et al.[106] formulated data-driven NN-based aerodynamic frameworks for rapid airfoil shape optimisation, using airfoil geometry and flow conditions as input features. Similarly, Andres et al.[11] emphasized the importance of ML and parameter optimisation for improving the prediction accuracy of aerodynamic data. In turn, Castellanos et al.[55] showed that deep neural networks outperform classical linear methods when the data dimensionality is much larger than the parameter set controlling the flow. More recently, Zhou et al.[412] introduced a flowfield reconstruction technique using a CNN combined with a residual architecture and demonstrated robust handling of shock-induced discontinuities. Related studies comparing conventional ROMs with neural network-based models confirm the effectiveness of the latter in capturing severe nonlinearities such as those found in aerodynamic icing [229].

The present work establishes an optimised deep learning framework for the reconstruction of distributed aerodynamic quantities in the transonic regime. We compare its performance with a POD-based ROM, a well-established linear method, to highlight the benefits in modelling highly nonlinear flow conditions. The approach is tested on a spectrum of physical phenomena and diverse geometric configurations within three established test cases involving 3D wings in the transonic regime. Significantly, this method not only proves highly effective in managing these complexities but also does so at a substantially reduced training cost compared to previously published studies.

The ambition is to demonstrate that the proposed methodology is not only accurate but also accessible and robust. To this end, two applications are shown as examples of the added-value it offers: both examples require many evaluations of the aerodynamic forces and moments and are notoriously impractical for CFD direct solutions. The first application focuses on a static aeroelastic solution, involving the iterative application of loads to the structure and observation of resulting rotations. This strategy proves valuable for quantifying the error introduced in a prediction, as exemplified in a previous study by Da Ronch et al. [79]. The second example pertains to the evaluation of aleatoric uncertainties, showcasing not only effectiveness but also efficiency in handling the propagation of uncertainties.

6.2 Methodology

This section describes the various numerical techniques used to construct the ROMs. The POD formulation, which serves as the baseline for this study, is introduced in the Foundations chapter (Section 2.1).

6.2.1 Deep Learning framework

A fully connected neural network (FCNN) architecture was chosen, which allows the model to predict each surface variable (e.g., local pressure) node by node while taking into account the entire flowfield distribution. Figure 6.1 outlines the overall framework: the ROM takes as input both the flow conditions and the spatial coordinates of each node, with the scalar flow parameters cast at each mesh node. This approach ensures that the flow conditions are consistent across the surface mesh. From there, the deep learning model processes these inputs to provide pointwise predictions of the pressure coefficient and the components of the skin friction coefficient. The ROM was implemented in the deep learning python library `TensorFlow/Keras`.

An important step in improving the predictive capability of the model is the selection of an optimal set of hyperparameters. Given the complexity of the design space, an efficient optimisation algorithm is required to systematically explore and refine these parameters. In this study, Bayesian optimisation is employed for hyperparameter tuning, using a probabilistic framework to balance exploration and exploitation during the

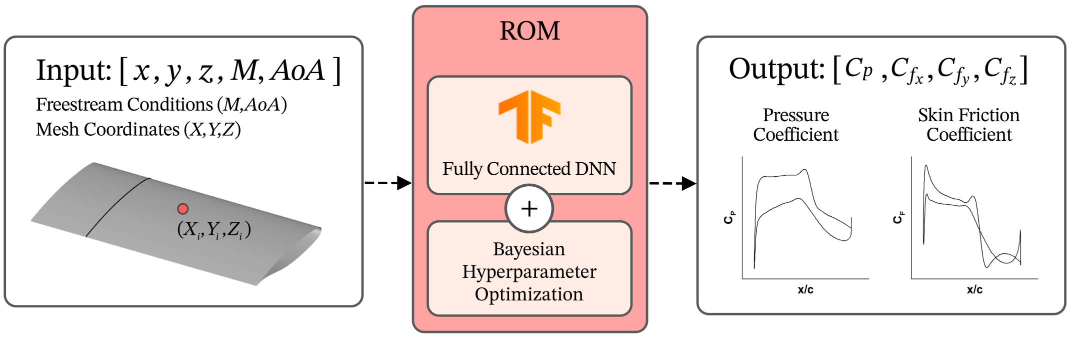

Input: [x, y, z, M, AoA]
Freestream Conditions (M,AoA)
Mesh Coordinates (X,Y,Z)
(X_i,Y_i,Z_i)
ROM
Fully Connected DNN
+
Bayesian Hyperparameter Optimization
Output: [C_p ,C_{f_x},C_{f_y},C_{f_z}]
Pressure Coefficient
Skin Friction Coefficient
C_p
C_f
x/c
x/c

FIGURE 6.1: Schematic of the deep learning framework.

search process (refer to Section 2.5.7 for a detailed overview of the method). The optimisation targets five key hyperparameters: learning rate, batch size, activation function, number of neurons, and number of layers. Among these, learning rate, batch size, number of neurons, and number of layers are continuous or integer-valued variables, while the activation function is a categorical variable, since its options correspond to qualitatively different nonlinear mappings rather than ordered numerical values. In practice, the activation function was handled as a discrete variable in the optimisation procedure. This simplification allowed its inclusion in the search space but does not fully reflect its categorical nature, where no natural ordering or metric exists between choices (e.g. ReLU vs. tanh). Consequently, the optimisation may introduce artificial structure in the search that does not correspond to meaningful similarities between functions. More advanced treatments of categorical variables (e.g. one-hot encodings, specialised kernels, or tree-based search strategies) could address this limitation and may further improve the robustness of the optimisation.

The design space for the hyperparameters, including possible values and step sizes for both continuous and discrete variables, is presented in Table 6.1. The selected ranges were intentionally chosen to be wide enough to allow meaningful optimisation. During the iterative optimisation process, it was observed that the hyperparameters tended to converge to values well within the specified bounds, demonstrating the effectiveness of Bayesian optimisation in efficiently navigating the search space. The hyperparameters of the neural network after the optimisation process for each of the three test cases are reported in Appendix A.4.

Hyperparameter	**Value**	**Step size**
Learning rate	10^{-4} to 10^{-6}	$5 \cdot 10^{-6}$
Number of Hidden Layers	1 to 8	1
Number of Neurons per Hidden Layer	4 to 204	8
Batch size	1 to 8	1
Activation function	TanH - ReLu - LeakyReLu	–

TABLE 6.1: Hyperparameters design space.

6.3 Test cases

In this section, the capability of the ML model to predict distributed aerodynamics quantities in transonic regime with different levels of geometric complexity and physical phenomena was assessed. Specifically, the model was tested on the BSCW and ONERA M6 test cases, followed by the wing-body configuration for NASA CRM (refer to Table 6.2).

This approach was undertaken to enhance the complexity of the geometry, incorporate diverse physics, and accommodate varying grid configurations. The progression ranged from a rectangular–wing to a swept–wing and eventually to a wing–body configuration. Additionally, the grid type changed from structured to unstructured. The physical phenomena also varied, encompassing changes in shock wave geometry, coupling between shock wave and separated boundary layer, and interaction between the fuselage and wing, especially in the junction area, resulting in the generation of complex vortices, pressure changes and boundary layer effects.

By conducting evaluations on these different test cases, the performance of the ML model was thoroughly examined under varying conditions, allowing for a comprehensive assessment of its predictive capabilities in transonic aerodynamics.

Test case	**Configuration**	**Mesh Type**	**Surface Mesh Points**
BSCW	Wing	Structured	130,816
ONERA M6	Wing	Structured	149,700
CRM	Wing–Body	Unstructured	78,829

TABLE 6.2: Main features of each test case.

Angle of attack AoA and Mach number M were chosen as the two independent parameters for the ROMs. They are in the ranges of $[0, 5]$ [deg] and $[0.70, 0.84]$, respectively. This range selection remains consistent across all test cases due to the analogous flow conditions characterizing the geometries. This choice ensures coherence throughout the analysis and guarantees a suitable level of complexity in the physics involved. The transonic regime leads to shock wave formation on the wing of each test case, whereas the high angles of attack to boundary layer separation.

The necessary number of samples, n, for AoA and M are defined through LHS method with a total of 70 points (Figure 6.2). Sixty percent of the samples (40 flight conditions with circles) are selected for training, 20% (15 flight conditions with squares) are selected for validation, and the remaining 20% (15 with diamonds) are left for testing.

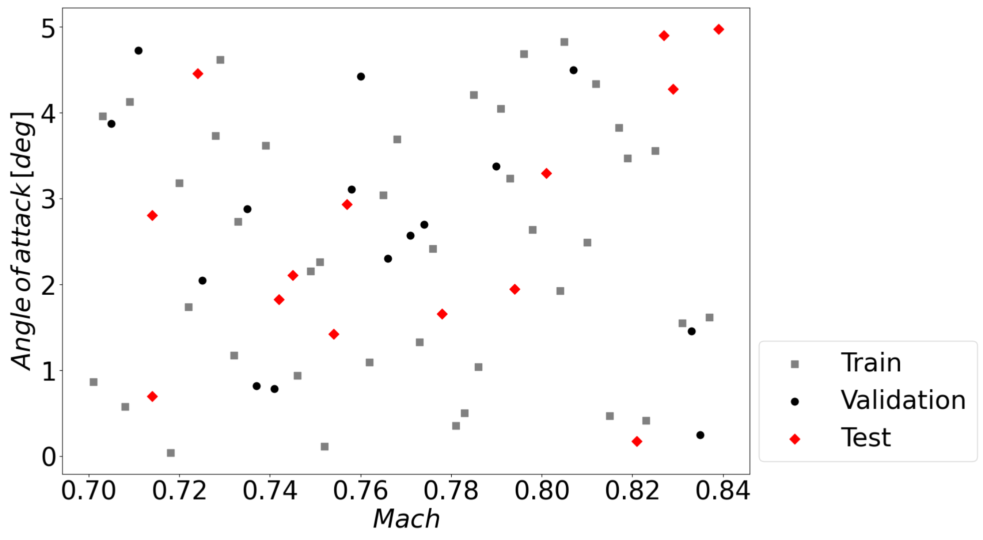


FIGURE 6.2: Training, validation and test samples for Mach number and angle of attack.

6.3.1 Benchmark Super Critical Wing

The first test case is the BSCW, introduced in Section 4.5. Although originally designed for flutter analysis (pitch and plunge DOFs), only the isolated wing is considered here. A structured surface grid with 130,816 surface elements was employed.

Figure 6.3 reports the convergence study on the number of training samples of the neural network in terms of Mean Absolute Percentage Error (`MAPE`) for BSCW test case. `MAPE` was calculated averaging the absolute error of each prediction in the test set. This prediction error is determined weighted averaging the error on each grid point with the cell area. The power fit figure of values outside of training samples range was added as a dotted line. As expected, increasing the number of training samples reduces `MAPE` on test dataset of pressure coefficient. In particular, double the training samples reduces drastically C_P error of almost one order of magnitude. Therefore, 40 training samples are kept for the rest of the study.

Table 6.3 shows `MAPE` on test dataset of pressure coefficient C_P and skin friction coefficient C_F. The error is normalized with respect to the cell area. Neural network behaves better with both quantities, showing its suitability for the complex aerodynamic problem analysed. Insights on the error distribution are achieved through error percentile analysis. This entails reporting the maximum and minimum error values, highlighting the median with an orange line, and using boxes to represent the 25th and 75th percentiles. Notably, the error band of the FCNN model is narrower, showing a conservative tendency.

Figure 6.4 compares pressure coefficient contour of CFD data with the reconstructed surface field employing two different techniques, POD+I and FCNN. The comparison

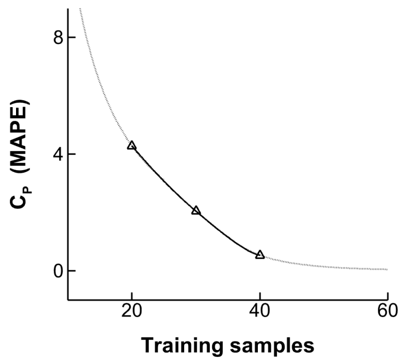
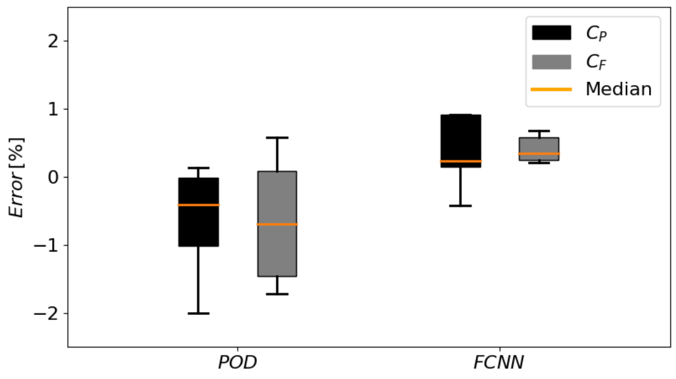


FIGURE 6.3: Training samples convergence study on pressure coefficient of BSCW test dataset. Dotted line denotes the power fit figure of values outside of training samples range.



	MAPE [%]	
	FCNN	POD+I
C_P	0.5382	0.8958
C_F	0.6080	2.2753

TABLE 6.3: Error analysis on test dataset of distributed aerodynamics quantities for BSCW test case.

is conducted on the test sample at $M = 0.801$ and $AoA = 3.295$ [deg], which was identified as one of the most challenging prediction from Righi et al. [288]. Wing root is located on the left side. Incoming flow impinges onto the wing from the leading edge. The FCNN recovers both the location and lateral extent of the shock system with high fidelity, reproducing the high-$|C_P|$ plateau upstream and the steep jump across the shock. By contrast, POD+I exhibits a systematic outward bias in shock position and an overprediction of shock footprint. This behaviour is consistent with its linear superposition hypothesis: sharp, advecting discontinuities (shock fronts) require many modes to localise in space, and small phase errors in the dominant modes translate into large apparent shifts in the reconstructed shock. Moreover, mode truncation tends to smooth sharp gradients, exaggerating the apparent shock thickness and footprint. The same physics is reflected in the skin-friction contour of Figure 6.5. Downstream of the shock, shock-boundary layer interaction produces a local C_F deficit associated with separation or a thickened boundary layer; the violet region marks the separated/near-separated layer bounded by reattachment lines that curve spanwise with shock sweep. FCNN accurately delimits both the onset and lateral extent of this low-C_F region, whereas

POD+I under-resolves the separation area and misplaces its maximum extent. The discrepancy again stems from linear reconstruction smoothing the separation gradients and failing to account for the nonlinear coupling between shock motion, corner flows near the root, and tip-vortex suction near the tip, all of which modulate the surface shear.

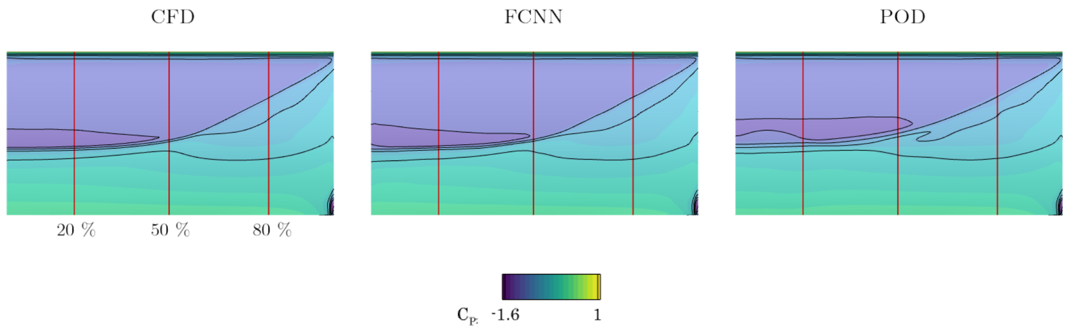


FIGURE 6.4: Prediction of the pressure coefficient contour on the upper surface for BSCW test case at $M = 0.801$ and $AoA = 3.295$ [deg].

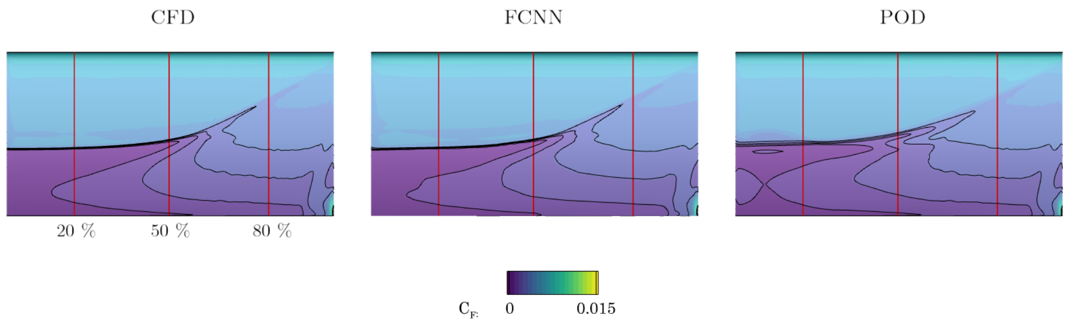


FIGURE 6.5: Prediction of the skin friction magnitude coefficient contour on the upper surface for BSCW test case at $M = 0.801$ and $AoA = 3.295$ [deg].

Figures 6.6 and 6.7 highlight the pressure and skin friction coefficient distribution at three sections along the span of the BSCW wing at $M = 0.801$ and $AoA = 3.295$ [deg]. The POD+I method captures the global trend of the variables but is unable to resolve any localized features, in contrast to the deep learning model, which is able to estimate the distribution with remarkable accuracy. In fact, we can notice a precise prediction of the pressure peaks in proximity of the shock location at 20% of the span which the POD+I approach is not able to capture correctly. This behaviour is likely due to the intrinsic linear formulation of POD+I which does not allow the model to take into account strong nonlinearities.

The better estimation of the flowfield using the FCNN is clearly shown from the reconstruction of the aerodynamic forces. Table 6.4 shows lift coefficient C_L, drag coefficient C_D and pitching moment coefficient C_M obtained with each approach, with a FCNN error less than 1%. C_M has been computed with respect to 30% of the chord, considering the rigid mounting system of the BSCW that induces pitch oscillations around this specific location. Note that aerodynamic forces are calculated by integrating the pressure coefficient distribution and the skin friction coefficient distribution over the wing surface. The contribution of shock wave magnitude predominantly influences the C_D, which both methods predict accurately. As a result, the error in predicting C_D is comparatively lower than that of other coefficients. Conversely, the C_L and C_M are sensitive to the shock wave location. In the POD+I method, there is a misplacement of the shock wave position, evident in the considerably higher error in C_L compared to FCNN method. The disparity is even more pronounced in the case of C_M, which is closely tied to the shock wave position. Neural network, renowned for its capability in pattern recognition, excels in capturing the shock wave position, thereby achieving a remarkably small error in C_M prediction. The relatively larger error in predicting C_L compared to C_M could potentially be attributed to minor distributed errors across the surface.

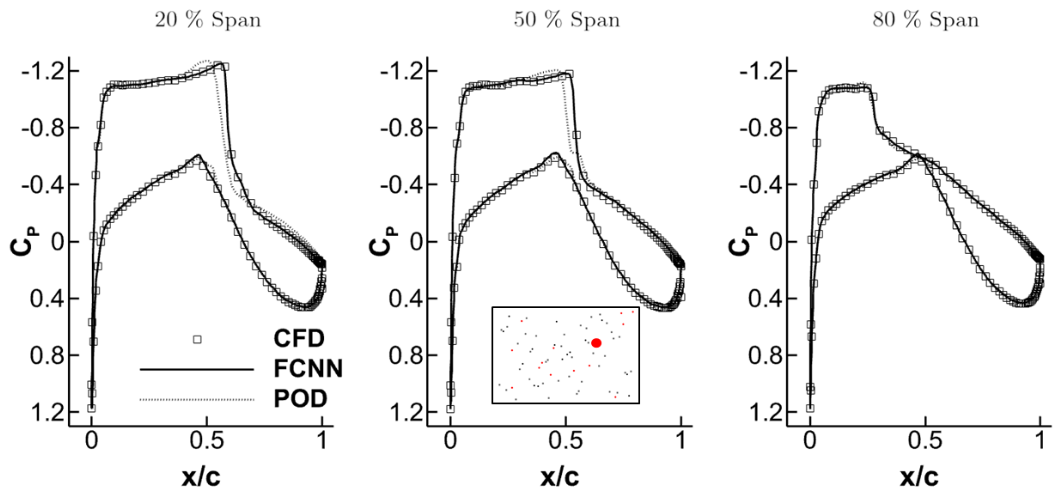


FIGURE 6.6: Pressure coefficient sections of BSCW at $M = 0.801$ and $AoA = 3.295$ [deg].

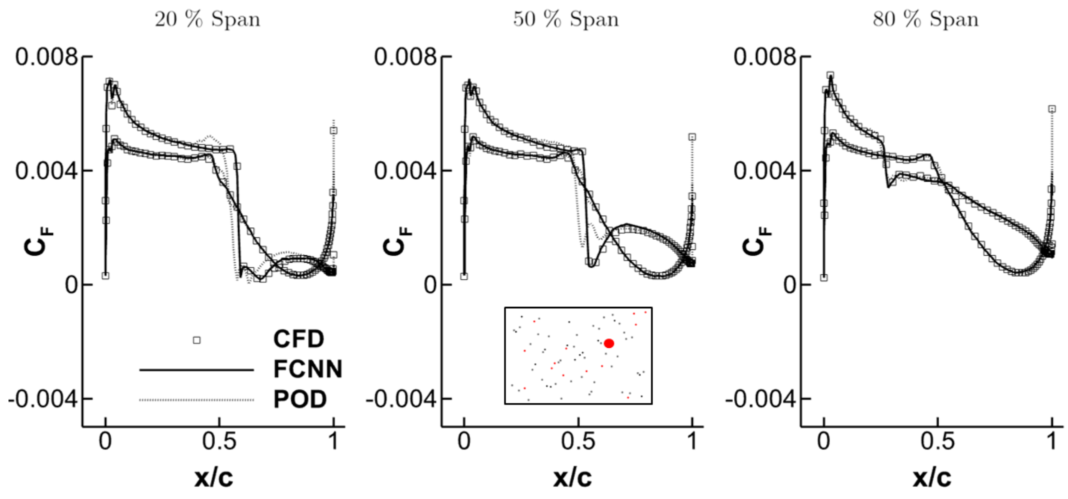


FIGURE 6.7: Skin friction coefficient sections of BSCW at $M = 0.801$ and $AoA = 3.295$ [deg].

	CFD	FCNN		POD+I	
		Value	Error (%)	Value	Error (%)
C_L	0.5256	0.5**186**	1.3318	0.5**122**	2.5495
C_D	0.0423	0.042**2**	0.2364	0.04**17**	1.4184
C_M	-0.0653	-0.06**49**	0.6125	-0.06**29**	3.6753

TABLE 6.4: Aerodynamic coefficients prediction for BSCW test case at $M = 0.801$ and $AoA = 3.295$ [deg].

Figure 6.8 compares the pressure coefficient distribution obtained with wind tunnel measurements [282] in two sections of the wing at $M = 0.74$ and $AoA = 0.2$ [deg] with the prediction of the presented methods. The flow conditions in this case involve the absence of boundary layer separation and a small shock wave on the upper surface of the wing. It is worth noting that we include experimental data in the figures to examine the discrepancies between ROMs and CFD, as well as between CFD and experimental data. This allows for a comparison of the two deviations. Given that ROMs depend on the quality of the data, being inherently data–driven, if CFD proves to be accurate, then this indicates that ROMs have the potential to achieve a similar level of accuracy. FCNN closely follows CFD in both sections, whereas POD+I mispredicts the pressure distribution at 60% of the span close to the leading edge showing small fluctuations.

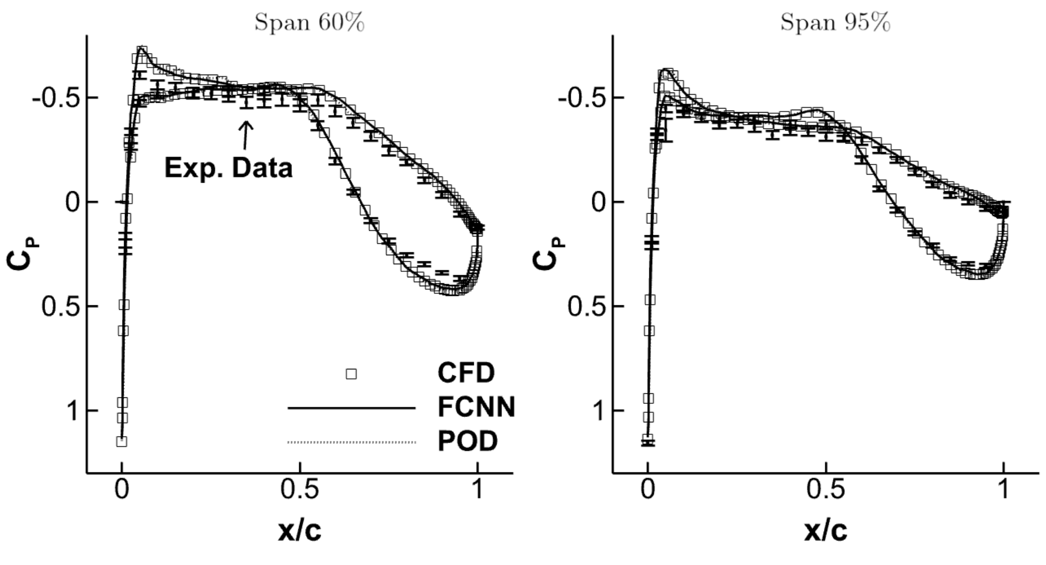


FIGURE 6.8: Comparison of pressure coefficient sections of BSCW at $M = 0.74$ and $AoA = 0.2$ [deg] with experimental data [282].

6.3.2 ONERA M6

The second test case is the ONERA M6 wing, a canonical benchmark that combines relatively simple planform and airfoil geometry with intrinsically complex transonic flow physics. In transonic conditions, the wing exhibits shock-boundary layer interaction, shock-induced separation and reattachment, and a characteristic λ–shaped shock pattern on the suction side.

The λ structure arises from the coupling between a spanwise-curved primary shock and the three-dimensional boundary layer on a swept wing. As the local normal Mach number increases, a near-normal shock forms typically at mid to outboard stations. With growing incidence, the adverse pressure gradient at the shock foot thickens the boundary layer and induces a separation bubble. Two oblique shock legs (the "λ" branches) then bound the separated region: an upstream, weaker branch associated with incipient separation, and a downstream, stronger branch linked to reattachment. The central portion of the pattern, often closer to normal, spans the shock across the separation bubble. Three-dimensional effects, driven by spanwise pressure gradients, crossflow within the boundary layer, and local variations of the normal Mach component with sweep, curve the shock front and cause the two oblique legs to diverge spanwise. At higher lift coefficients, the separated pocket expands and the λ becomes more pronounced; near the tip, geometric taper and reduced chord promote further three-dimensionality, altering both the position and the topology of the shock system.

The onset, strength, and topology of the λ–shock depend sensitively on Mach and Reynolds numbers, as well as on local sweep (through the normal Mach component). Increasing M steepens the adverse pressure gradient and moves the shock forward, while Re controls boundary layer state and separation extent. Their interaction yields sharp, non-monotonic changes in pressure gradients and skin friction, which complicate interpolation across (M, α_0, Re).

A mixed-type grid with $18.8 \cdot 10^6$ elements and 149,700 surface elements was generated, structured on the wing surface and in the first layers of the boundary layer, voxel in the rest of the computational domain. A $y^+ = 1$ is adopted. The computational domain extends 100 chords from the solid wall to the farfield. An impression of the grid can be obtained from Figure 6.9.

The JST central scheme with artificial dissipation is adopted for the discretisation of convective flows. The gradients of the flow variables are calculated using a Green Gauss method. The linear solver biconjugate gradient stabilisation is chosen, with ILU preconditioner applied. A chord of 0.6461 [m] and a scaling area of 0.7532 [m^2] have been considered for the aerodynamic forces calculation.

Table 6.5 shows `MAPE` on test dataset of distributed aerodynamics quantities. The error is normalized with respect to the cell area. Similar to the previous test case, FCNN consistently outperforms the POD+I method, exhibiting an error lower than 0.19% for C_P and 0.53% for C_F compared to the POD+I method. A detailed examination of the error percentile analysis graph reveals a narrower error spread for FCNN, underscoring its higher accuracy on the test dataset in comparison to the POD+I method.

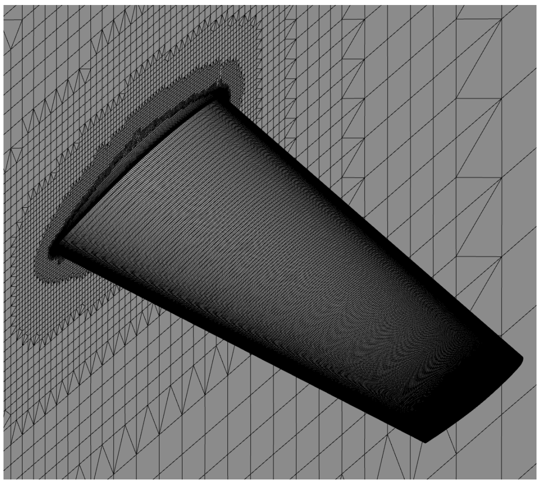
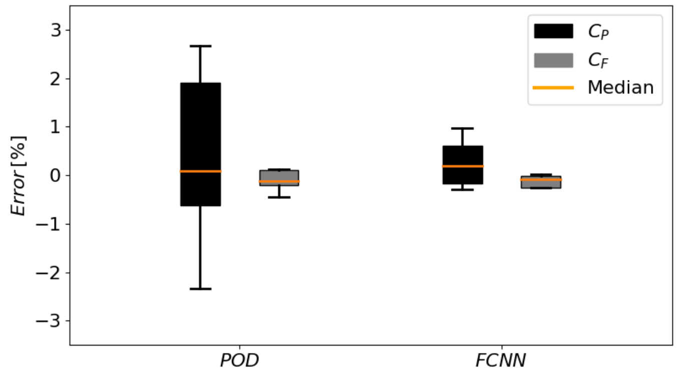


FIGURE 6.9: Impression of the ONERA M6 structured grid.



	MAPE [%]	
	FCNN	**POD+I**
C_P	2.4949	2.6836
C_F	0.8470	1.3745

TABLE 6.5: Error analysis on test dataset of distributed aerodynamics quantities for ONERA M6 test case.

Figure 6.10 shows the surface pressure coefficient field on a new prediction at $M = 0.8395$ and $AoA = 3.06$ [deg] employing two different techniques, POD+I and FCNN. This particular condition was selected due to the availability of experimental data [309]. Consequently, an additional CFD simulation has been computed for reference.

At these transonic conditions the suction side is dominated by a λ-shaped shock system: an attached main front with bifurcated legs and a finite spanwise sweep, arising from the combination of leading-edge sweep, local thickness/curvature distribution (with a minimum curvature near mid-chord), and spanwise pressure communication. Both ROMs capture the overall shock footprint, but FCNN better reproduces the shock curvature and inboard-outboard migration of the front, including the upstream high-$|C_P|$ plateau and the steep jump across the main leg. The improved fidelity is consistent with FCNN ability to encode nonlinear dependencies between (M, AoA) and local shock position, whereas the linear superposition in POD+I induces phase errors that manifest as slight misplacement and broadening of the shock footprint.

The skin-friction reconstruction in Figure 6.11 reflects the same physics. Downstream of the shock, shock boundary-layer interaction produces a local C_F deficit associated with incipient or partial separation; the λ legs delineate the edges of this separated/thickened region. FCNN accurately delineates both the onset and lateral extent of the low-C_F patch, particularly near the tip where the tip-vortex suction lowers pressure, twists the shock front, and amplifies three-dimensionality. POD+I captures the qualitative pattern but underestimates the magnitude of the C_F deficit close to the tip and slightly smooth the reattachment area, a typical consequence of modal truncation and the inability of a linear basis to localise sharp, advecting discontinuities.

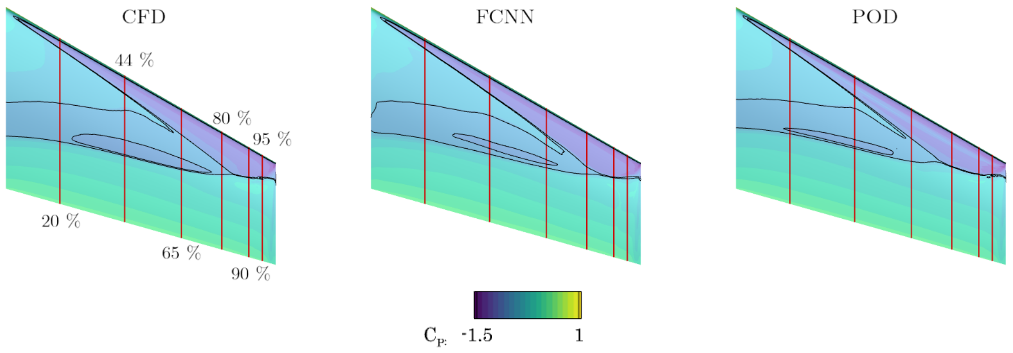


FIGURE 6.10: Prediction of the pressure coefficient contour on the upper surface for ONERA M6 test case at $M = 0.8395$ and $AoA = 3.06$ [deg].

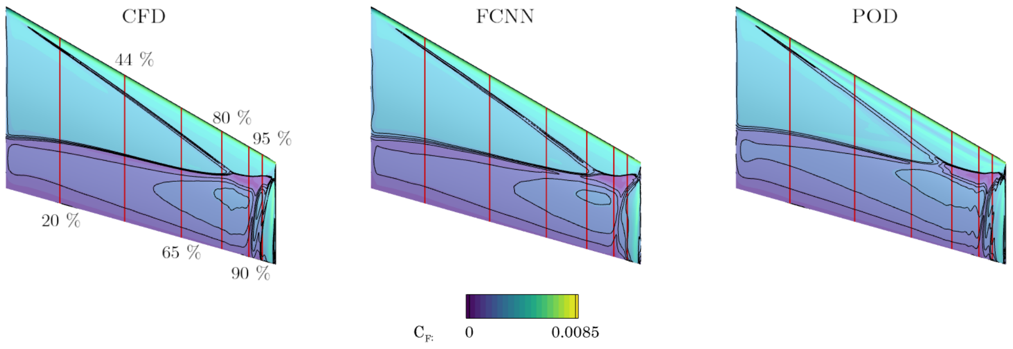


FIGURE 6.11: Prediction of the skin friction magnitude coefficient contour on the upper surface for ONERA M6 test case at $M = 0.8395$ and $AoA = 3.06$ [deg].

The pressure coefficient distribution at several station along the wing span at $M = 0.8395$ and $AoA = 3.06$ [deg] is reported in Figure 6.12, where a comparison with available experimental data [309] takes place. The interest to add experimental data to the figures is to highlight the similar deviation between ROMs and CFD, as well as between CFD and experimental data. This suggests that the ROM could potentially attain a level of accuracy similar to the CFD. In contrast with Figure 6.10, the better behaviour of FCNN is clearly shown. At each span station, the neural network closely follows CFD results, whereas POD+I shows large C_P oscillations in proximity of the shock wave. This non-smooth pressure field results from POD modes with small eigenvalues, which get dramatically magnified throughout the interpolation phase. This is further emphasized by examining the skin friction reconstruction presented in Figure 6.13. The current results have been based on POD modes that encompass 95% of the total kinetic energy. Reducing the number of modes only resulted in poorer outcomes. Conversely, increasing them for accounting 99% of the total energy does not impact significantly on the results. Therefore, we have made the decision to retain a high number of modes in order to achieve the highest level of accuracy in the predictions.

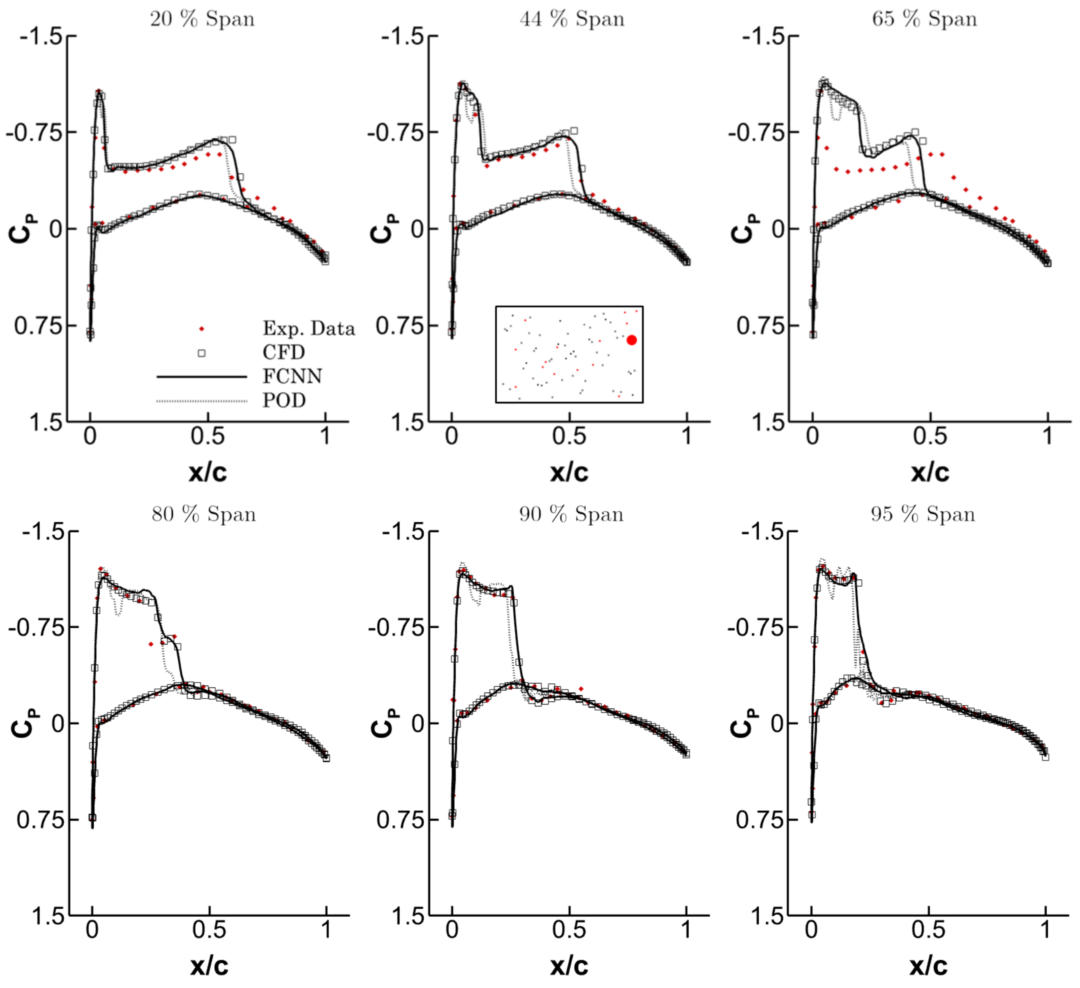


FIGURE 6.12: Comparison of pressure coefficient sections ONERA M6 at $M = 0.8395$ and $AoA = 3.06$ [deg] with experimental data [309].

In fact, the accurate pressure and skin friction estimation of FCNN brings to a better prediction of aerodynamic coefficients, as shown in Table 6.6. Specifically, the FCNN exhibits excellent prediction capabilities for the lift coefficient, improving by almost an order of magnitude over POD+I. Additionally, the error of FCNN in the drag and pitching moment coefficients is less than half of what is achieved with POD+I. The reason behind the lower error of the FCNN on the lift coefficient compared to the other two coefficients can be understood by examining Figure 6.10 and 6.11. These figures reveal that the contour of the C_F exhibits non–smooth isolines in the CFD solution, which suggests a simulation that may not have converged effectively. However, this issue does

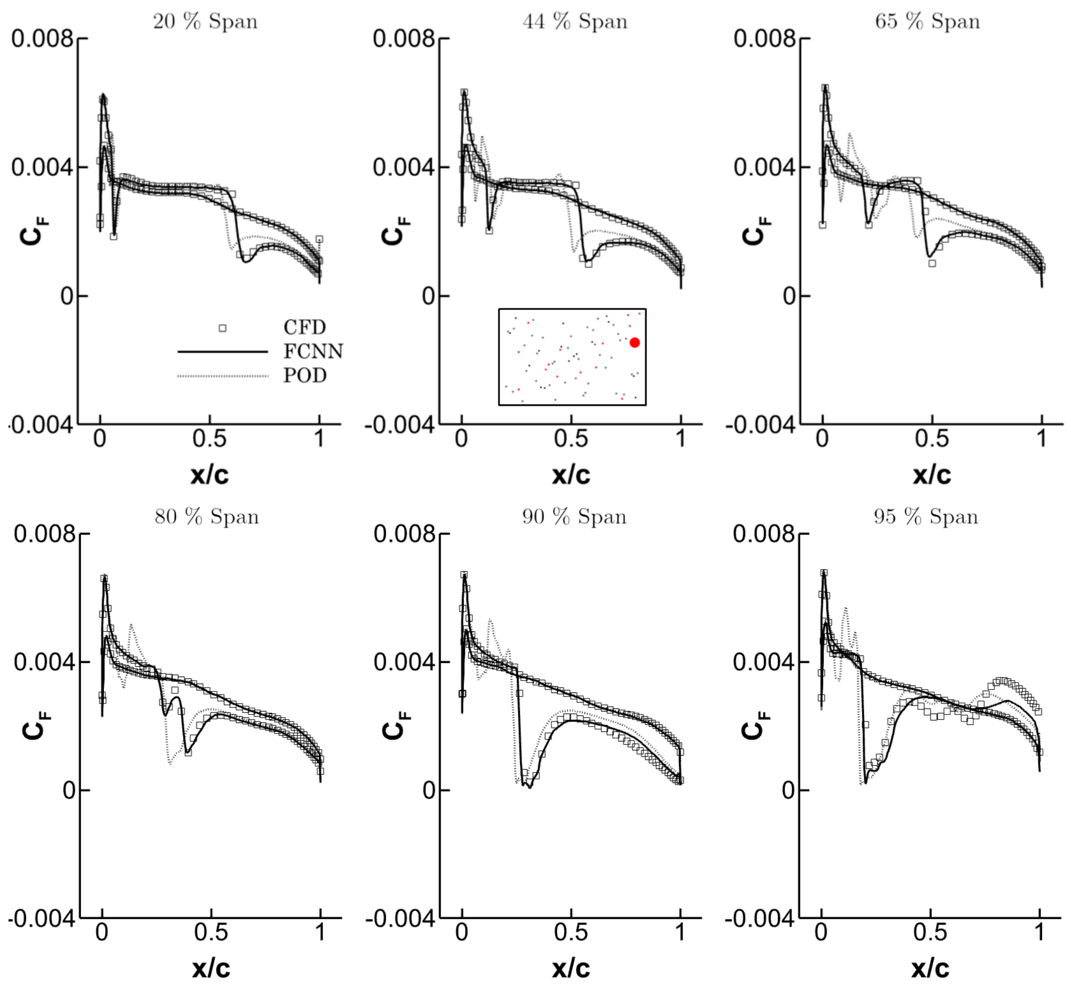


FIGURE 6.13: Skin friction coefficient sections of ONERA M6 at $M = 0.8395$ and $AoA = 3.06$ [deg].

not impact the deep learning algorithm, which is capable of providing accurate solutions even in regions with complex physical phenomena. This highlights the significant advantage of using deep learning to regularise CFD data.

	CFD	**FCNN**		**POD+I**	
		Value	Error (%)	Value	Error (%)
C_L	0.2763	0.276**1**	0.0724	0.2**698**	2.3525
C_D	0.0183	0.01**90**	3.8251	0.0**169**	7.6503
C_M	-0.1281	-0.12**67**	1.0929	-0.1**229**	4.0593

TABLE 6.6: Aerodynamic coefficients prediction for ONERA M6 test case at $M = 0.8395$ and $AoA = 3.06$ [deg].

6.3.3 NASA Common Research Model

The third test case is the NASA Common Research Model (CRM), a transonic wing-body configuration originating from the AIAA CFD Drag Prediction Workshop [356].

The model features a conventional swept, tapered low wing connected to a wide-body fuselage, providing a representative platform for cruise-relevant shock-boundary layer interactions on lifting configurations. The CRM generates a markedly 3D flowfield. At transonic conditions, the primary shock on the suction side coexists with tip-vortex roll-up and a persistent juncture (horseshoe) vortex at the wing-fuselage junction. Viscous effects concentrate in regions of strong adverse pressure gradient and geometric curvature: (i) trailing-edge zones where the boundary layer is thin yet highly accelerated, promoting separation lines and strong wake shear; and (ii) the wing-fuselage junction, where the corner flow experiences coupled streamwise and lateral pressure gradients, giving rise to a junction vortex system and localised separation. These structures imprint distinct signatures on the skin-friction distribution and can reorganise separation/reattachment lines across the trailing portion of the wing and the inboard root fillet.

The CRM response also exhibits a nonlinear dependence on freestream parameters. Increasing Mach number strengthens the shock and moves it forward, intensifying the adverse pressure gradient; changes in angle of attack modify both the loading and the local normal Mach component, thereby shifting shock position and separation extent.

The computational grid was derived from the DLR one created for the AIAA Drag Prediction Workshop [355]. The unstructured grid has $8.8 \cdot 10^6$ elements and 157,374 surface elements. The computational domain extends 100 chords from the fuselage to the farfield. A $y^+ = 1$ is adopted. An impression of the grid can be obtained from Figure 6.14.

The Lax–Friedrichs central scheme with low artificial dissipation set to 0.02 is adopted for the discretisation of convective flows. The gradients of the flow variables are calculated using a Green Gauss method. The linear solver biconjugate gradient stabilisation is chosen, with ILU preconditioner applied. A chord of 0.1412 [m] and a scaling area of 0.1266 [m^2] were considered for aerodynamic coefficients calculation.

This test case represents the most challenging prediction for our deep learning framework due to the complex geometry, physics and grid configuration. It differs from the others as it has a remarkable amount of surface points in regions with mostly linear flow behaviour, such as vast areas of the fuselage, which is advantageous to the linear POD+I technique. Consequently, we chose to quantify the `MAPE` exclusively on the wing, which exhibits high nonlinearity. The results in Table 6.7 reveal that FCNN outperforms POD+I in both coefficients for the wing. Additionally, when examining the error percentile analysis graph, it becomes evident that FCNN exhibits a lower error spread compared to POD+I. This underscores the superior performance and accuracy of the developed deep learning framework in capturing the nonlinearities.

Surface pressure contour predictions are shown in Figure 6.15 on the test sample at $M = 0.827$ and $AoA = 4.898$ [deg]. This specific condition has been selected due to

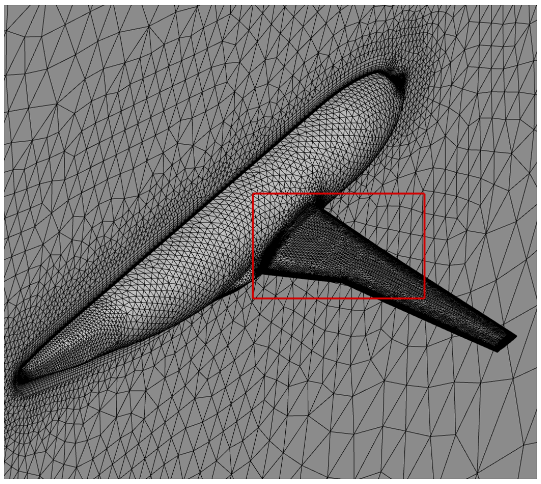
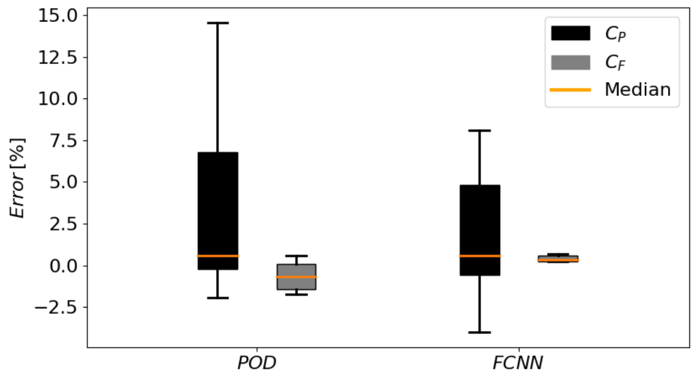


FIGURE 6.14: Impression of the CRM unstructured grid. More focus on the red rectangle in Figure 6.15 and 6.16.



	MAPE [%]	
	FCNN	**POD+I**
C_P	1.7385	3.7136
C_F	1.1942	1.2142

TABLE 6.7: Error analysis on test dataset of distributed aerodynamics quantities for CRM test case.

the complex physics involved, in terms of boundary layer detachment and shock wave formation on the upper wing surface. These phenomena interplay with the vortex generated at the wing-fuselage junction, making the prediction particularly challenging.

In regions dominated by attached flow and gradual pressure recovery, both ROMs provide accurate surface pressure reconstructions, which explains the negligible prediction error on the fuselage outside the junction region. However, Figures 6.15 and 6.17 reveal discrepancies near the wing-fuselage junction and in the kink area of the wing, where strong three-dimensionality and vortex-shock interaction occur. These nonlinear features expose the limitations of POD+I: the linear modal basis cannot adapt to sharp localised gradients or advecting structures, and therefore fails to reproduce the suction peak and shock position with sufficient accuracy. FCNN, by contrast, recovers the suction distribution and shock-induced compression more faithfully, reflecting its ability to interpolate nonlinear dependencies between global parameters and localised flow features.

The skin-friction distributions in Figures 6.16 and 6.18 reinforce this interpretation. Downstream of the shock, shock-induced separation thickens the boundary layer and lowers C_F; in addition, the fuselage vortex impinges on the inboard wing, producing strong streamwise gradients and modulating the shock front. FCNN provides smoother yet physically consistent reconstructions in these regions, filtering numerical noise while retaining the footprint of separation and reattachment near the trailing edge. POD+I underpredicts the reduction in C_F, particularly where the vortex interacts with the shock, confirming its inability to encode the nonlinear coupling between vortex-induced upwash and shock boundary-layer interaction.

Spanwise sections in Figure 6.17 further illustrate the physics. At 20% span the flow remains attached with no shock; moving outboard, a distinct shock appears and intensifies toward mid- and outboard stations. FCNN tracks this progression with good fidelity, whereas POD+I fails to reproduce the suction peak amplitude and shock location, leading to phase errors in the reconstructed pressure distribution. These shortcomings explain the higher overall error in aerodynamic coefficients for POD+I (Table 6.8), while FCNN yields lower C_L errors by better capturing the nonlinear redistribution of pressure.

The CRM case highlights the necessity of nonlinear mapping strategies in regions where multiple 3D mechanisms coexist: vortex formation at the fuselage junction, kink-induced spanwise redistribution of lift, and transonic shock motion. FCNN provides a clear improvement by resolving these coupled effects, though residual discrepancies remain near the junction and kink, where irregular and non-uniform data distributions can challenge the network [233]. The next chapter with more complex network architectures will further improve accuracy in these highly nonlinear flow regions.

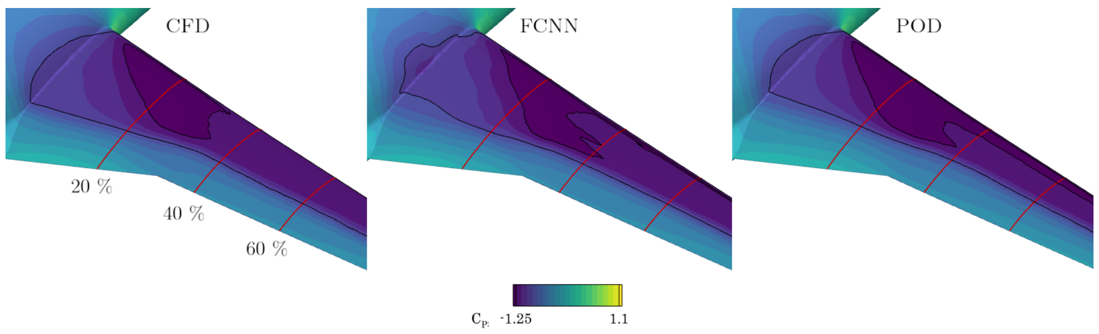


FIGURE 6.15: Prediction of the pressure coefficient contour on the upper surface for CRM test case at $M = 0.827$ and $AoA = 4.898$ [deg].

6.3.4 Computational Cost Analysis

A comprehensive computational cost analysis was conducted to assess the efficiency of reduced-order methods compared to the high-order approach, as presented in Table 6.9. In CFD simulations, it was found that an average of 900 CPU hours is required

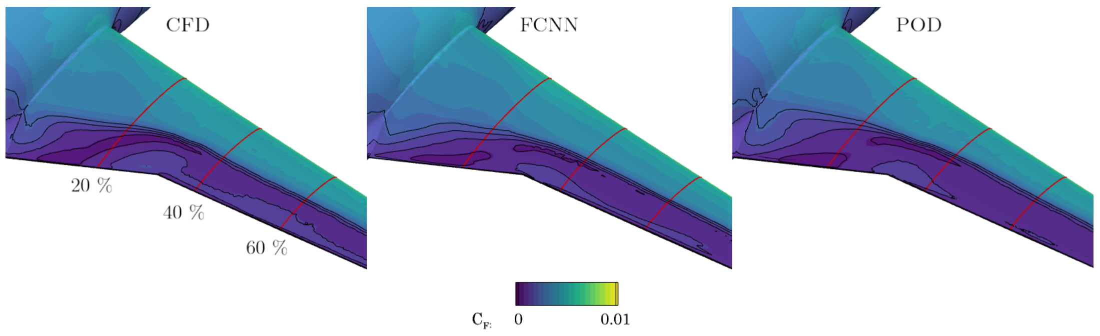


FIGURE 6.16: Prediction of the skin friction magnitude coefficient contour on the upper surface for CRM test case at $M = 0.827$ and $AoA = 4.898$ [deg].

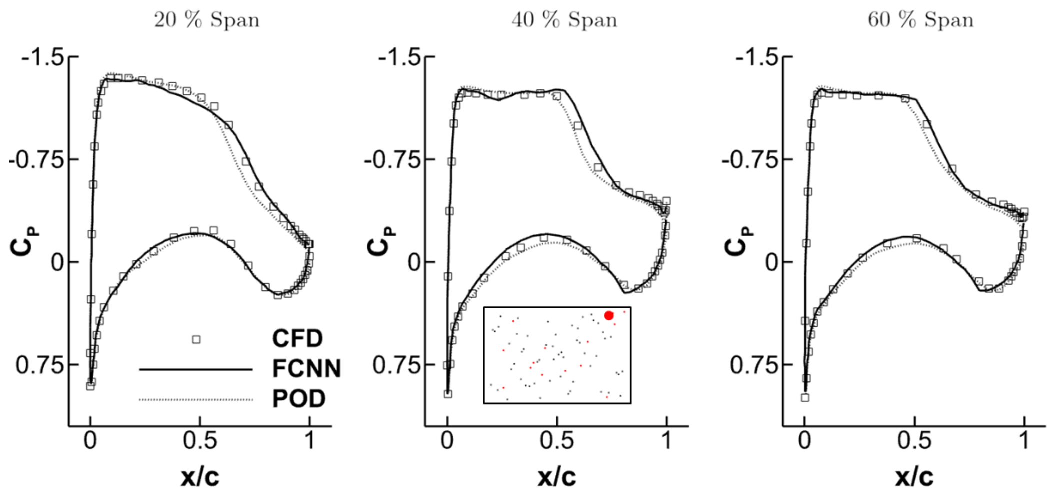


FIGURE 6.17: Pressure coefficient sections of CRM at $M = 0.827$ and $AoA = 4.898$ [deg].

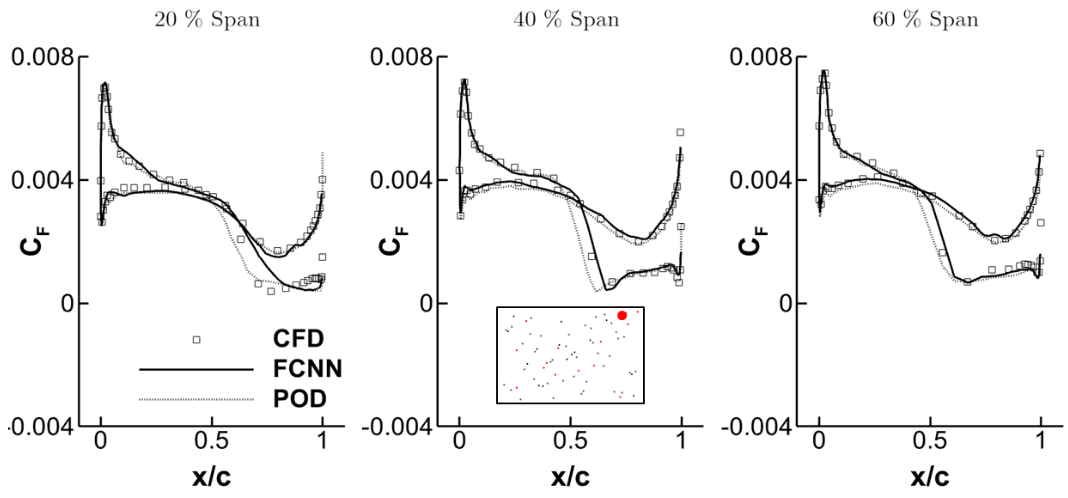


FIGURE 6.18: Skin friction coefficient sections of CRM at $M = 0.827$ and $AoA = 4.898$ [deg].

for a single run, while generating the entire dataset demands approximately 65,000 CPU hours. The prediction time for a single sample is comparable between the two ROMs. However, the FCNN, which delivers highly accurate predictions, necessitates

	CFD	**FCNN**		**POD+I**	
		Value	Error (%)	Value	Error (%)
C_L	0.4927	0.4**860**	1.3598	0.4**874**	1.0807
C_D	0.0821	0.0**787**	4.1413	0.0**773**	5.8738
C_M	-0.0349	-0.03**48**	0.2865	-0.0**238**	31.7103

TABLE 6.8: Aerodynamic coefficients prediction for CRM at $M = 0.827$ and $AoA = 4.898$ [deg].

computationally demanding optimisation and training phases. As a result, while the POD+I method may suffice during the initial design stages, when precise and reliable results are paramount, employing neural network becomes an extremely valuable approach despite its greater computational requirements.

One of the primary obstacles we faced throughout this study was the considerable computational cost associated with each high-fidelity simulation. Therefore, it is important to adopt a philosophy aimed at minimizing the amount of training data required to develop an accurate model. Neglecting the costs associated with data generation can be a potential risk, as it may compromise the computational benefits of using a ROM. Hence, striking a balance between model accuracy and computational efficiency is essential for successful implementation.

Test case	**CFD (CPU hours)**		**FCNN (GPU hours)**			**POD+I (CPU hours)**	
	Simulation		optimisation	Training	Prediction	Training	Prediction
	(70 runs)	(1 run)	(100 trials)	(1 model)	(1 sample)	(1 model)	(1 sample)
BSCW	42,000	600	10	0.6	0.0001	0.002	0.0002
ONERA M6	126,000	1,800	11	1.6	0.0001	0.002	0.0002
NASA CRM	28,000	400	7	0.4	0.0001	0.002	0.0002

TABLE 6.9: Computing cost comparison between ROMs and CFD for each test case.

6.4 Applications of the Framework

To evaluate the feasibility of the proposed method, two applications are presented. The applications require several repeated evaluations of the aerodynamic forces and moments, which are performed with the deep learning framework. The test case is the BSCW wing. The first application concerns the static aeroelastic response in a flow regime with a nonlinear relationship between aerodynamic loads and structural displacements. The second application deals with the propagation of aleatory uncertainties of Mach number and angle of attack on the aerodynamic forces. Here, we used a neural network to generate a large number of samples in a short time scale and employed the Polynomial Chaos Expansion (PCE) to investigate statistical quantities. We demonstrated the efficiency of the method and, importantly, verified that the results are consistent with physical principles. By evaluating the performance and robustness of the framework in these applications, we gain valuable insights into the potential of the method and its ability to address complex challenges encountered in practical settings. Indeed, the applications show that the proposed deep learning framework is both effective (it provides accurate data), efficient (at low cost), and robust (it reduces the dependence on data quality in each sample).

6.4.1 Static Aeroelastic Response of BSCW Wing

To evaluate the static aeroelastic behaviour of the BSCW wing, the proposed deep learning framework is used to predict the surface pressure and skin friction distributions under varying freestream conditions. These vector fields are then integrated to compute the aerodynamic pitching moment M, which is passed to a structural model to resolve the elastic twist angle θ. The latter is obtained from the linear relationship $K \cdot \theta = M$, where K is the structural stiffness matrix. Because the moment depends on the twist angle via changes in the local angle of attack, a fixed-point iterative procedure is employed. At each iteration, the updated twist modifies the angle of attack, which is re-evaluated by the model until convergence is achieved. The structural properties and geometric layout of the BSCW configuration are detailed in Chapter 4.

The influence of increasing dynamic pressure on aeroelastic behaviour is noteworthy as it enhances the impact of pressure in relation to structural stiffness. Figure 6.19 illustrates this effect by presenting the incremental elastic twist angle θ [deg] predicted by the deep learning framework as a function of dynamic pressure for rigid angles of attack ranging from 0 to 5 [deg] at $M = 0.74$. The circular symbols in the figure represent CFD results obtained from Heeg and Chwalowski [142].

In static aeroelastic solutions with rigid angles of attack equal to or less than 1.15 [deg], the dynamic pressure induces a negative twist, resulting in a reduction of the angle of attack (nose down). However, for angles of attack greater than 1.15 [deg], the induced elastic twist angle becomes positive (nose up) and increases with dynamic pressure. This phenomenon can be attributed to a shift in the centre of pressure, resulting in a change of sign for the pitching moment. This, in turn, leads to an increase in the aerodynamic load as the angle of attack rises. A remarkable agreement might be notice between the presented results and computational ones from Heeg and Chwalowski [142].

To conclude, we want to point out that a static aeroelastic solution has been computed within the presented framework in 0.002 GPU hours. The problem is nonlinear and necessitates iterative resolution. This stands in stark contrast to the method proposed by Heeg and Chwalowski [142], which required about 1,000 CPU hours with a gradient method. Unlike traditional method that would necessitate running a CFD simulation at each step, this approach demonstrates efficiency and effectiveness in handling aeroelastic simulations.

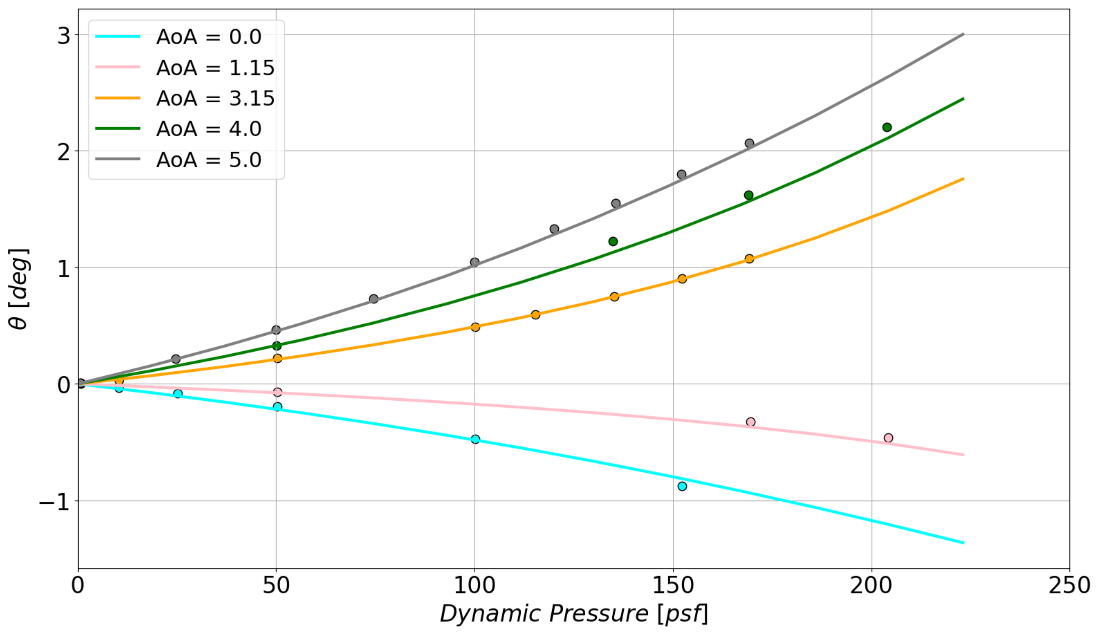


FIGURE 6.19: Static elastic twist angles of BSCW at $M = 0.74$ for different initial AoAs.

6.4.2 Propagation of Uncertainties of BSCW Wing

The aim of this study is to effectively address the uncertainties associated with flow parameters. This research is motivated by the significant variability observed in the results of the Second Aeroelastic Prediction Workshop, particularly the substantial scatter in the test case results [144]. This study leverages the developed deep learning framework to efficiently generate a high number of samples, and Non-Intrusive Polynomial Chaos Expansion (NIPCE) technique [122, 329, 372, 386] to approximate the probability distribution associated with the random variables. This technique has been successfully applied to similar problems, as documented in the literature [155, 327, 328]. In the expansion, the variable of interest Y, which depends on a vector of random variables ξ, is expressed as a series expansion:

$$Y(\xi) = \sum_{i=o}^{p} \alpha_i \Psi(\xi) \tag{6.1}$$

Here, Ψ represents multivariate basis functions, often Hermite polynomials for normally distributed random variables [329, 277], and α_i are the spectral coefficients. The number of terms in the expansion, denoted by p, is determined by the number of random variables n and the chosen order of the NIPCE s, and can be calculated using $p = \frac{(n+s)!}{n!s!}$.

To determine the spectral coefficients α_i, a minimum of m experiments are required, assuming oversampling with $m > n$. The m values of Y are sampled using the LHS method. In this study, the random variables Mach number and angle of attack have been selected for analysis (refer to Table 6.10). It is assumed that the random variables follow a normal distribution with given standard deviation, 0.01 for M and 0.1 for AoA. These values are chosen to align with the order of magnitude of the uncertainty typically encountered during standard wind tunnel campaigns [5, 174, 256]. The NIPCE is constructed using Hermite functions.

The proposed approach facilitates the generation of a diverse set of samples, ensuring comprehensive exploration of aleatory uncertainties. NIPCE technique provides a systematic and computationally efficient means to characterize the probability distribution of the flow parameters, and the integration of the deep learning framework permits to efficiently explore the parameter space, capturing complex patterns and relationships among the flow variables.

RV	**Distribution**	**Std deviation**
Mach	Normal	0.01
AoA	Normal	0.1

TABLE 6.10: Freestream Random Variables, their distribution and standard deviation values used to quantify uncertainty.

Figure 6.20 displays the Probability Density Function (PDF) of the lift coefficient at $M = 0.80$ and $AoA = 3.0$ [deg], which was identified as a condition with significant variability by Righi et al. [288]. It is important to note that the PDF retains its shape and characteristics as the NIPCE order increases. Moreover, as the NIPCE order is augmented, the PDF reveals pronounced nonlinearities that become more prominent.

A comparison of the PDFs at various Mach numbers and AoAs is showcased in Figure 6.21. In the low M-AoA region, the PDF exhibits a linear behaviour, as evidenced by their symmetric distribution. However, as the Mach number and AoA increase, the PDF displays pronounced nonlinearity, characterized by the observed asymmetry in the curves. Notably, at $M = 0.84$ and $AoA = 5$ [deg], this asymmetry is less pronounced. This phenomenon can be attributed to the complete separation of the boundary layer and the relatively less intense interaction with the shock wave, lending support to an explanation grounded in linear theory.

Subsequently, we conduct a sensitivity analysis utilizing Sobol indices to measure the contribution of individual input variables and their interactions to the overall variability in model output. These indices assist in discerning the relative importance of each factor. The Sobol indices, depicted in Figure 6.22, reveal the impact of Mach variations. It is evident that as the Mach number increases, the influence of Mach, treated as a random variable, becomes more prominent, primarily due to the intensified compressibility effects. This dominance of Mach is particularly significant at high Mach numbers, where the flow experiences separation, diminishing the influence of the AoA in comparison. These findings align well with previously published results by Heeg et al. [144], lending credibility to our observations.

This study demonstrated that our model can generate the PDF at each condition in only 0.01 GPU hours, highlighting the efficiency and resource–saving benefits of our approach within this context. Crucially, the results consistently align with physical principles, underscoring the reliability of the approach. Moreover, it is pertinent to acknowledge that, although our present focus did not encompass the incorporation of uncertainty into the neural network, this remains a potential avenue for future exploration and refinement of the proposed methodology.

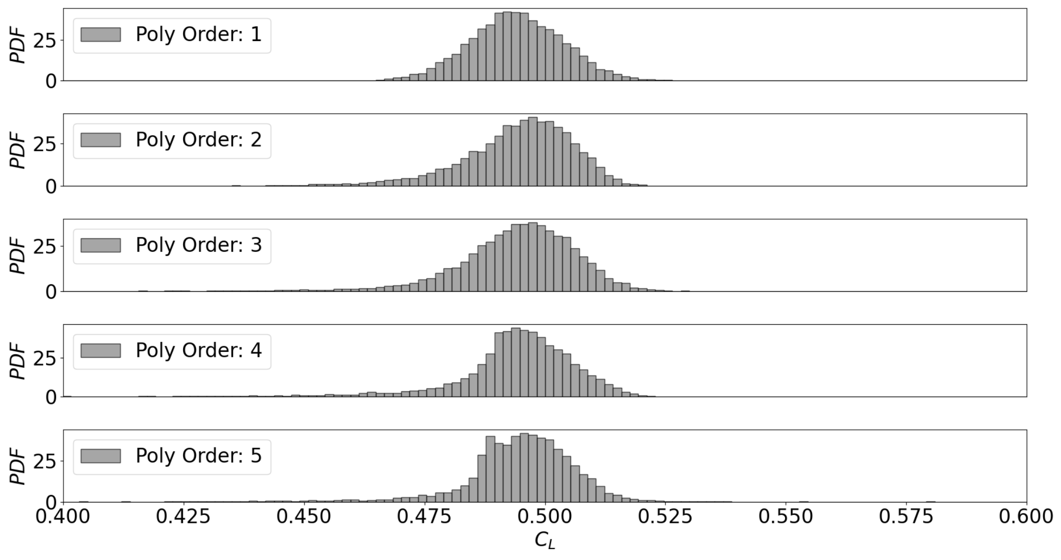


FIGURE 6.20: Probability Density Function of lift coefficient at $M = 0.80$ and $AoA = 3.0$ [deg], polynomials are consistent up to $p = 5$, 10,000 samples, 2 Random Variables.

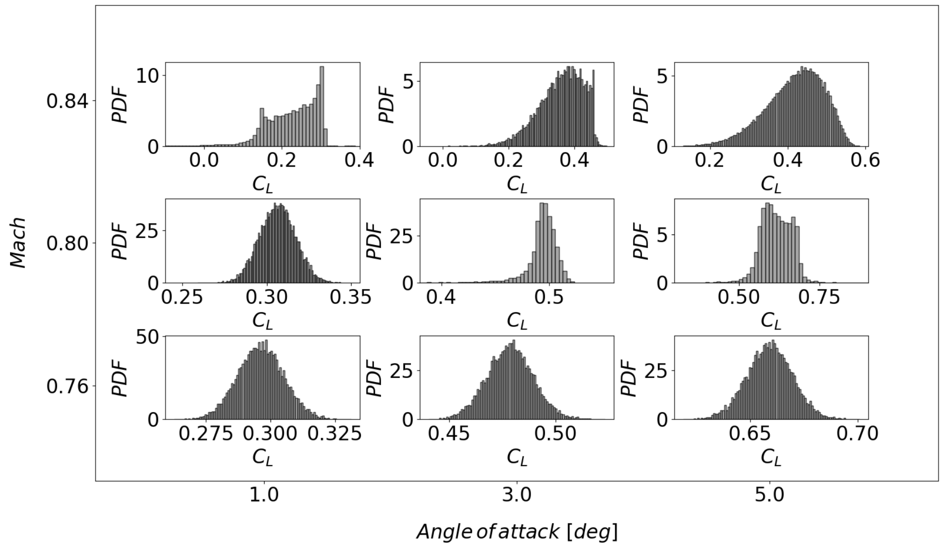


FIGURE 6.21: Probability Density Function of lift coefficient, $p = 4$, 10,000 samples, 2 Random Variables.

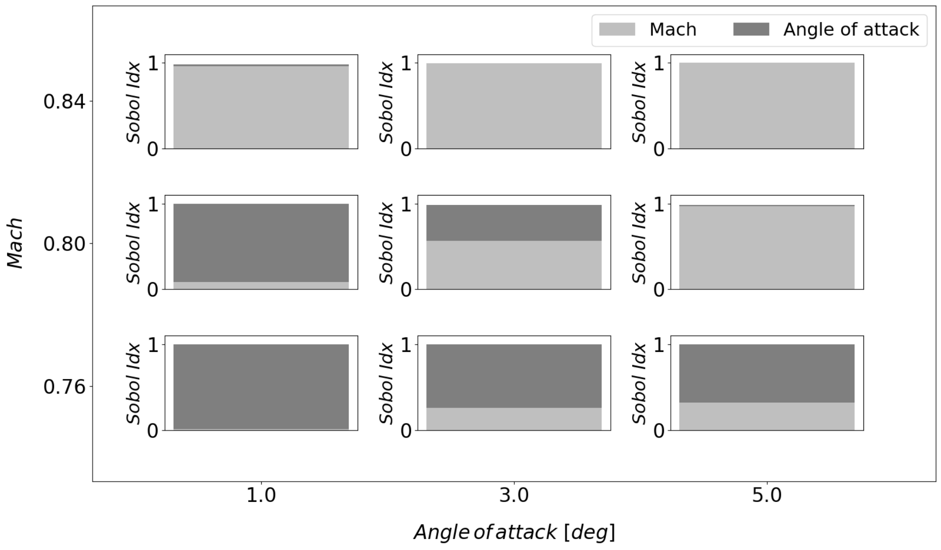


FIGURE 6.22: Sobol indices of the two random variables for lift coefficient.

6.5 Conclusions

This chapter presented a deep learning framework for modelling nonlinear aerodynamic phenomena in the transonic regime, with a particular focus on predicting 3D distributed flow quantities essential for aerodynamic load estimation and aeroelastic analysis. Evaluated on test cases involving shock waves and boundary layer separation, the framework showed strong predictive performance for wing-only configurations, but revealed limitations when dealing with more complex wing-body geometries. These results highlight that structured, ordered meshes can enhance the predictive accuracy of the proposed model, while non-homogeneous unstructured meshes remain a challenge.

The method also demonstrated considerable capabilities in efficiently estimating the static twist deformation for the BSCW configuration. Its ability to significantly reduce the computational effort compared to traditional aeroelastic simulations provides deeper insight into the structural behaviour and offers promising avenues for improving wing design. In addition, the incorporation of NIPCE allowed the quantification of uncertainties in the freestream variables, showing that strongly non-Gaussian distributions can indeed account for large variations observed in previous Aeroelastic Prediction Workshops.

The effectiveness of the proposed approach is the result of two main factors. First, optimizing the NN architecture is essential for both faster hyperparameter tuning and improved predictive accuracy. Second, regularisation of the data is a considerable advantage of the presented algorithm. In fact, in regions where convergence is hard to meet, the model showcases its ability to deliver a smooth solution, effectively regularizing the data.

These findings address several research objectives:

1. **Efficient Handling of Transonic Nonlinearities**
   By extending deep learning techniques to 3D wing configurations, the chapter underscores the feasibility of rapid aerodynamic predictions in regimes dominated by shocks and boundary layer separation, supporting Research Objective O3 of spatial modelling.

2. **Uncertainty Quantification**
   The NIPCE analysis clearly connects the variability of input conditions to the observed scatter in aerodynamic outputs, consistent with the thesis emphasis on robust design through quantified confidence (Research Objective O2).

3. **Towards More Complex Configurations**
   While FCNN-based approaches performed well on structured grids, their limitations on non-homogeneous unstructured grids emphasize the need for advanced methods that capture geometric relationships in irregular meshes more effectively, motivating Research Objective O3.

Therefore, the concluding part of this chapter naturally transitions to a geometric deep learning strategy that aims to overcome the shortcomings identified in current FCNN approaches. This graph-based strategy is explored in Chapter 7, and offers a more powerful architecture for handling irregular node distributions and capturing the critical transonic flow structures that arise in realistic, unstructured aerodynamic grids.

Chapter 7

Autoencoder Graph Convolutional Networks for Steady Transonic Flowfields on Unstructured Grids

This chapter addresses the challenges of non-homogeneous unstructured grids, widely used in CFD. Their prevalence in fluid dynamics has motivated the development of advanced ROM strategies. Our approach leverages geometric deep learning through an autoencoder built on graph convolutional networks, which improves prediction accuracy by propagating information across nodes and highlighting influential regions. Addressing Research Objective O3, this chapter extends deep learning-based aerodynamic modelling to unstructured domains via graph-based architectures. Unlike conventional CNNs limited to structured meshes, the graph convolutional autoencoder enables mesh-consistent aerodynamic predictions while maintaining geometric fidelity. Key innovations include pressure-gradient-based dimensionality reduction, rapid connectivity reconstruction using Mahalanobis distance, network architecture optimisation, and a physics-informed loss function derived from aerodynamic coefficients. These enhancements yield a robust and accurate model with lower prediction errors compared to prior graph-based methods. The framework is validated on two test cases, wing-only and wing-body, demonstrating accurate steady-state flowfield reconstruction across a 2D parametric space. This work builds on the structured-grid models introduced earlier and sets the stage for the spatio-temporal extensions developed in the next chapter.

7.1 Introduction

In recent years, addressing problems characterized by non-homogeneous and unstructured grids has become a central topic of research in the field of aerospace engineering.

A pertinent example lies within the CFD field, where the initial step involves mesh generation, entailing the discretisation of the fluid domain through the finite volume method. This mesh serves as a computational grid that enables the simulation of fluid flow and related phenomena within a defined space. A non-homogeneous unstructured grid is characterized by irregularly shaped elements (such as triangles or tetrahedras) connected in a non-regular pattern. The spacing between grid points varies across the domain, providing greater resolution in areas of interest, such as regions with complex geometries or flow features, while optimizing computational resources in less critical areas.

These geometrical complexities, coupled with the need to accurately capture nonlinear fluid dynamics, motivate novel ROM strategies. Traditional numerical methods, including POD and manifold learning [116, 115, 289, 416], have been extensively applied. Yet, as grid complexity increased, these approaches struggled to capture strong nonlinearities, handle irregular geometries, and scale to high-dimensional datasets.

ML offers a way to address these challenges by learning from complex, non-traditional data structures. Early studies using deep neural networks showed promise in modelling complex patterns in fluid dynamics [55, 160, 297], but non-homogeneous, unstructured grids remain particularly challenging. This need for advanced architectures led to the concept of geometric deep learning, which emerged around 2017 [45]. This approach introduced the use of graph-structured data prediction through GNN architectures [128], specifically designed for applications involving interconnected entities. GNNs excel in capturing intricate relationships and dependencies within graph nodes and their connections [383, 407, 413]. The ability of GNNs to consider both local and global context through neighbourhood aggregation mechanisms makes them well-suited for tasks where topological information is critical. These versatile networks have found extensive application as a foundation for solving classical artificial intelligence tasks and addressing various challenges in data science and analysis [68, 178, 252, 275, 383, 417]. Notably, it has been shown that GNNs outperform traditional approaches in handling local nonlinearities [149]. They have demonstrated precise predictions for aerodynamic performances [173] and flowfield properties [253]. Additionally, they are effective in addressing complex time-dependent problems [233] and have proven successful in diverse aerospace applications, including data fusion tasks [203], uncertainty quantification [204], and multi-objective optimisation [208]. On the other hand, implicit neural representations (INRs, or neural fields) parameterise a continuous function with a coordinate-based MLP, offering resolution-agnostic sampling and compact storage. INRs have been applied to external aerodynamics, delivering rapid steady predictions once trained, and are part of a broader family of operator-learning approaches (e.g. neural operators) that learn mappings between function spaces rather than discretised arrays [56, 170]. While attractive for their mesh-independence and continuous evaluation, INRs and operator learners often require careful encodings to represent sharp gradients and may demand substantial data to cover geometry/flow variability. Neural operators share similar goals but many formulations assume regular lattices, implying remeshing for complex aircraft geometries [170].

In parallel, CNNs have excelled in many fields [169, 118, 255, 264, 294], leveraging properties of sparse connection, parameter sharing, and translation invariance. However, CNN-based methods assume a regular grid structure [409], which precludes direct application to irregular meshes. Consequently, GCNs[103] emerge as a powerful alternative, extending convolutional operations to non-homogeneous unstructured grids. By using one-element filters that sweep connected nodes and re-weight outputs via the corresponding edge weights, GCNs allow for localized, repeated filtering without the Cartesian grid assumption. This approach allows direct input of raw 3D model mesh data avoiding unnecessary pre–computation or feature extraction methods that may introduce bias or loss of information.

Moreover, computational costs remain significant for large unstructured grids. High-dimensional data can lead to overfitting and poor generalisation, while the autoencoder architecture can help to capture the most important features, enhancing the model performance. Conventional autoencoders tailored to regular data require further adaptation for unstructured meshes [31, 120, 132, 137, 294], which may involve additional interpolation or network regularisation steps.

With the challenges outlined before, our architecture has been specifically designed to address the limitations of traditional methods. We adopt a GCN autoencoder operating directly on the CFD mesh. The choice is driven by computational constraints and data topology. First-order GCN layers reduce to sparse matrix-vector operations with cost linear in the number of edges, exploiting mesh sparsity; this is markedly cheaper than attention-based graph transformers whose complexity typically scales quadratically with node count [357, 68]. Message passing over physical adjacency captures anisotropic, mesh-induced couplings without Cartesian assumptions, avoiding pre-interpolation steps that can blur shocks and boundary layers [253]. Pooling/unpooling extends the receptive field at controlled cost.

We refined our methodology by building on the architecture introduced by Massegur et al. [232], incorporating more complex approaches to improve model accuracy. Our model employs an autoencoder GCN architecture and stands out from other graph-based approaches by introducing several innovations: a dimensionality reduction module based on pressure-gradient values, fast connectivity reconstruction using Mahalanobis distance, Bayesian optimisation of the network architecture, and a physics-informed loss function that includes a penalty term for the pitching moment coefficient. Collectively, these enhancements lead to a systematically lower error compared to previous graph-based studies [232] and traditional technique, while remaining tractable on

GPU limits. Two test cases characterized by distinct physical phenomena are proposed to validate the developed methodology: wing–only model and wing–body configuration.

7.2 Methodology

This section explains the methodology that guided the development of the present model. First, the general architecture of the autoencoder graph convolutional network is presented, followed by a detailed explanation of each component that constitutes each module of the model.

7.2.1 Graph Autoencoder Architecture

The steady–state prediction ROM developed in this work uses freestream conditions and mesh coordinates as input, and is designed to predict specific values for each point in the graph. Scalar freestream conditions are assigned to each node of the surface alongside their respective coordinates. A Gradient-Based Autoencoder GCN model (GB-AE-GCN) with two custom levels of dimensional reduction/expansion was implemented. The output of the model is generated by four parallel GCN layers. The whole architecture is finally trained for the pointwise prediction of the four desired output C_p, C_{f_x}, C_{f_y} and C_{f_z}. A schematic of the model architecture is illustrated in Figure 7.1.

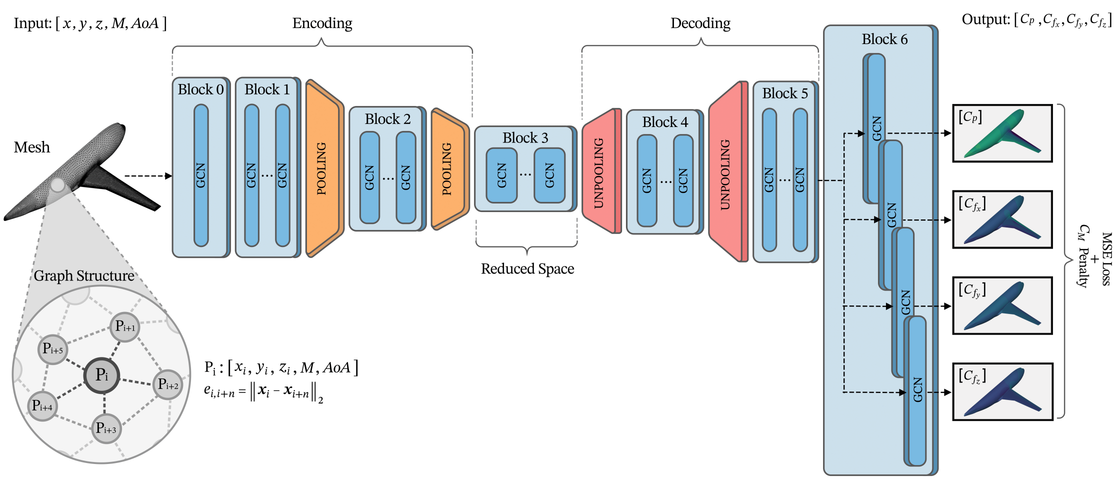


FIGURE 7.1: Schematic of the graph autoencoder architecture.

The use of an Encoder-Decoder based architecture aims to reduce the computational effort by reducing the size of the data during the prediction, increasing the scalability of the system, and also allows the model to consider the connection between more distant points of the mesh, which are not directly connected initially. This step has been taken in order to reproduce the CNN behaviour used in AI-based computer vision tasks [130, 209], with the addition of the information about the distances and connections between the points given by the graph structure.

The pooling module implemented in our approach is a gradient–based point selection and connection reconstruction. The pressure gradient–based point selection task involves two key steps. Firstly, we compute the gradients for each sample and subsequently identify the regions of interest across all samples. This process enables us to pinpoint areas where pressure gradients exhibit significant disparities, thereby identifying points characterized by heightened nonlinearity. Once these critical points are identified, we implement a Moving Weighted Least Squares Interpolation (MWLSI) algorithm [172, 276] to seamlessly interpolate values from the source points (fine grid) to the destination points (coarse grid). To reconstruct connections and calculate Euclidean distances between the remaining points, a Mahalanobis distance [88] based method was implemented. This method re-establishes connections of each point with its 5 neighbours in the destination space based on Mahalanobis distances calculated in the original space.

The aim of the unpooling module is to reconstruct the original structure of the input to generate an output with the same dimension of the input, but this operation requires a new interpolation matrix computed with the MWLSI algorithm (refer to Section 7.2.2 for details) to calculate the missing data of the new nodes, moving from a coarser grid back to the finer one. The pooling and unpooling modules are pre-computed in order to save computational resources. An on demand version could be implemented for adding learnable capabilities of space reduction/expansion, especially on time–variant problems.

To enhance the predictive capacity of the model, we adopted two strategies: a Bayesian optimisation and a custom loss function. The Bayesian approach has been employed for optimizing the neural network hyperparameters, such as number of layers per block, units per layers and compression ratio of encoding/decoding operations. By leveraging Bayesian optimisation, the model systematically explores and adapts these hyperparameters to maximize performance and predictive accuracy. The custom loss function aims to optimize the distribution of C_P and C_F components across the grid by minimizing the mean squared error (`MSE`) between the model predictions and the ground truth. Factors like shock waves and boundary layer separation introduce complexity to predictions, affecting force resultant and, therefore, moment calculation. Incorporating physics-based penalty terms in loss functions has proven effective in improving aerodynamic prediction accuracy [285]. Therefore, a penalty term for the pitching moment coefficient C_{M_y} has been introduced into the `MSE` loss function. This addition, represented as $Loss = \text{MSE} + \lambda \cdot C_{M_y}$, with $\lambda = 0.01$ for dimensional consistency, guides the model towards more precise predictions, particularly in terms of shock wave positioning.

Bayesian optimisation for Hyperparameters Tuning

To enhance the predictive accuracy of our model, we employed Bayesian optimisation to fine-tune the hyperparameters, as detailed in Section 2.5.7 of the Foundations chapter. Bayesian optimisation iteratively refines the hyperparameters by balancing exploration and exploitation through probabilistic modelling, thus efficiently navigating large design spaces.

We utilized the `Optuna` library [3], which integrates seamlessly with the `PyTorch` framework, to perform this optimisation. The primary hyperparameters targeted in this process were the number of layers per block, the number of units per layer, and the dimensionality compression/expansion value. The number of layers per block defines a group of layers before or after a spatial reduction operation in the encoding module, with the decoding module mirroring this structure to save computational resources. The number of units per layer refers to the number of neurons in a single GCN module, optimised initially for the encoding phase and then mirrored for the decoding phase, ensuring both dimensional compatibility and minimized computational cost. Lastly, the dimensionality compression/expansion value controls the ratio between the number of points in coarser and finer meshes during the compression phase, and vice versa during expansion.

The design space for these hyperparameters is presented in Table 7.1, with ranges carefully selected to allow sufficient exploration while ensuring convergence to optimal values.

Hyperparameter	Value	Step size
Compression ratio	1/4 - 1/3 - 1/2	–
Number of Hidden Layers per Block	1 to 3	1
Number of Neurons per Hidden Layer	32 to 512	16

TABLE 7.1: Hyperparameters design space.

To ensure sufficient convergence towards the optimal set of hyperparameters, we conducted 30 trials, each limited to 500 epochs to manage computational costs. Upon completion of the optimisation phase, we executed the training procedure for the refined encoder-decoder architecture that minimizes the loss function for 2000 epochs. For a detailed overview of the final optimised architectures, please refer to Appendix A.5.

7.2.2 Dimensionality Reduction/Expansion

The core idea behind the use of space reduction/expansion operations is to minimize non–influential information from nodes that do not contribute to the nonlinearity of the system. The aim is to streamline the complexity of hidden layer operations and eliminate redundant information that could potentially mislead the model. The pooling and unpooling modules entail different concepts, which are herein explained. An overview is presented in Figure 7.2, where it is possible to distinguish all the processes used for construct and reconstruct hidden spaces. During encoding, we select points based on pressure gradients, creating a reduced–point cloud. We then use a Mahalanobis distance–based method to reconstruct connectivity, resulting in a connected reduced graph. Node values are computed through grid interpolation using the moving weighted least squares method. In decoding, we interpolate on the original fine point map and connectivity using the same method with a new interpolation matrix.

Pressure gradient–based Point Selection

The goal of gradient–based point selection is to find the optimal approach for implementing a pooling phase. During this phase, points are chosen for removal from the mesh graph in the space reduction operation. The general idea is to employ a more advanced point selection method instead of relying solely on the simplistic density–based approach [232]. By doing so, the pooling phase can more effectively consider the primary region where nonlinear phenomena occur.

This method entails two fundamental steps. Initially, gradients on pressure value are computed for each sample. Then, the value of gradient for each example is used for the identification of regions of interest across the entire dataset. This approach facilitates the detection of areas where pressure gradients display notable differences on pressure, thereby identifying points characterized by heightened nonlinearity.

Spatial gradients are computed for each point by considering the pressure value at each node of the graph. To calculate gradients in unstructured grids, it is assumed that the pressure variable varies linearly in all dimensions, yielding:

$$p - p_0 = \Delta p = \Delta x p_x + \Delta y p_y + \Delta z p_z \tag{7.1}$$

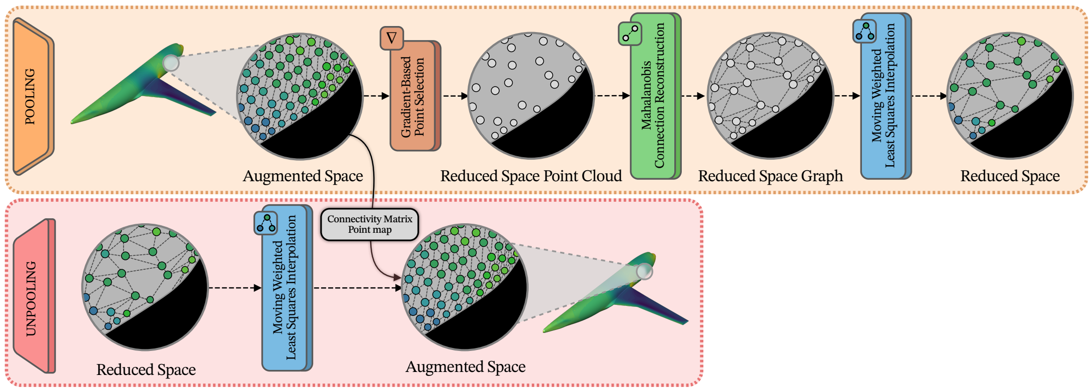

POOLING
Augmented Space
∇
Gradient-Based Point Selection
Reduced Space Point Cloud
Mahalanobis Connection Reconstruction
Reduced Space Graph
Moving Weighted Least Squares Interpolation
Reduced Space
UNPOOLING
Reduced Space
Moving Weighted Least Squares Interpolation
Connectivity Matrix Point map
Augmented Space

FIGURE 7.2: Pooling and unpooling modules architecture.

Where p_0 is the pressure in the node. Then, a matrix equation is constructed using the pressure differences among all nodes neighboring the current node. With five connections, the matrix equation results in:

$$\begin{bmatrix} \Delta x_1 & \Delta y_1 & \Delta z_1 \\ \Delta x_2 & \Delta y_2 & \Delta z_2 \\ \Delta x_3 & \Delta y_3 & \Delta z_3 \\ \Delta x_4 & \Delta y_4 & \Delta z_4 \\ \Delta x_5 & \Delta y_5 & \Delta z_5 \end{bmatrix} \begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix} = \begin{bmatrix} \Delta p_1 \\ \Delta p_2 \\ \Delta p_3 \\ \Delta p_4 \\ \Delta p_5 \end{bmatrix} \tag{7.2}$$

Equation (7.2) is then inverted via the least-squares method to compute the gradient vector.

Starting form the value of the gradients calculated for all the points in the graph, a suitable probability distribution has been employed to determine the number of points retained in the reduced space. The challenge arises in regions of the original mesh with low gradients, potentially resulting in an inadequate number of nodes at the coarsened level and leading to an irreversible loss of information. Conversely, excessive node removal in regions of originally high gradients may result in insufficient accuracy reconstruction of complex physics phenomena. Thus, an appropriate node selection strategy is essential to ensure the proper representation of both high and low gradient regions in the coarsened domain. This is obtained using a probability function applied on the gradient value of each mesh element (node):

$$Pr(i) = 1 + \frac{1 - e^{-2i/n}}{1 - e^{-2}}(Pr_1 - Pr_n) + Pr_1 \quad \text{for} \quad i = 1, \ldots, n \tag{7.3}$$

Here, i represents the mesh node index, sorted by pressure gradient value in descending order, and n is the total number of nodes. The probabilities Pr_1 and Pr_n denote the choices for the highest and lowest gradients, respectively, set to 0.2 and 1.

After each space reduction, an unconnected point cloud is obtained, therefore it is essential to restore the connectivity between neighbours.

Mahalanobis connection reconstruction

To identify the neighbours of each node in the point cloud after the reduction process and thereby restore connectivity, we use a reconstruction method based on the Mahalanobis distance [88], that is widely used in clustering problems and other statistical classification techniques [384, 123]. The Mahalanobis distance is a measure of the distance between points in a distribution. Unlike the simple Euclidean distance, the Mahalanobis distance takes into account the spread of points in different directions through the covariance matrix of the distribution of points. Using this type of distance, it is possible to connect each point to its neighbours by following the distribution of points in the finer mesh by using the covariance matrix calculated in the original space. This method minimizes false connections between opposite faces of the mesh which are considered close according to the simple Euclidean distance. Therefore, the distance between points is calculated using the following equation:

$$D_M(x, y) = \sqrt{(x - y)^T S^{-1} (x - y)} \tag{7.4}$$

Where x and y are two points of the reduced space and S is the covariance matrix of the distribution of the points in the finer mesh. Additionally, to reduce the searching field of nearest neighbours on the reduced space, we used the K-d tree algorithm [117] to determine for each point a subset of 250 elements using Euclideian distance, and then selected the nearest neighbours by following the Mahalanobis distance calculated only in that subset.

Building on this process, we tested the accuracy of connection reconstruction by removing all connections from the fine grid and using Mahalanobis and Euclidean distances to identify the nearest neighbours for each node, creating new connections. These reconstructed connections were compared to the original grid and the percentage of incorrect or false connections was calculated for each node. The average error across all nodes assessed overall accuracy. For the BSCW model, we observed that the Mahalanobis distance resulted in only 0.4% false connections, compared to 48.6% when using Euclidean distance. Similarly, for the CRM model, Mahalanobis distance led to 1.5% false connections, while Euclidean distance produced 58.4%. These false connections can significantly impact the model performance by introducing noise into the learning process, linking unrelated nodes, and misguiding the model, which reduces its capacity to accurately capture the physical phenomena accurately. This can result in poorer generalisation and lower prediction accuracy, particularly in regions with complex flow dynamics.

Moving Weighted Least Squares for Grid Interpolation

Efficient information transfer between grids is a critical aspect in the proposed methodology. While one option involves using a neural network with learnable weights, this approach could significantly escalate computational requirements. On the contrary, traditional interpolation techniques may yield inaccuracies that are not suitable for our purposes [232]. Consequently, we opted for the Moving Weighted Least Squares (MWLS) technique [276, 172], which has proven to deliver accurate results in similar problems [232]. This decision aims to strike a balance between accuracy and computational efficiency, while also ensuring the conservation of the integrated quantity across both grids and maintaining continuity across the domain [276]. MWLS assigns varying weights to neighbouring data points based on their proximity to the interpolation point, allowing for a more adaptive and accurate representation of the underlying data. The approach involves fitting a local polynomial to a subset of nearby points, with the influence of each point weighted according to its distance. This adaptability ensures that closer points have a more significant impact on the interpolated value, while those farther away contribute less.

The core idea of MWLS is to generate an interpolation matrix $I_{S_s \to S_d}$ that maps features $\mathbf{y}_i$ from the source grid S_s with n_s nodes to the destination grid S_d with n_d nodes:

$$\mathbf{y}_j = I_{S_s \to S_d} \mathbf{y}_i \quad \forall j \in S_d, \quad \forall i \in S_s$$

To accomplish this, we construct a shape function $u(\mathbf{x}) = \mathbf{m}^T(\mathbf{x})\,\mathbf{a}$ that approximates the grid data y_i by minimizing the least squares error:

$$\min L = \sum_{i \in S_s} \left(\mathbf{m}^T(\mathbf{x_i})\,\mathbf{a} - y_i \right)^2 w(\mathbf{x}_i)$$

where $w(\mathbf{x}_i) = e^{-\|\mathbf{x}-\mathbf{x}_i\|_2}$ is the Gaussian weight function that ensures nodes closer to the interpolation point have a greater influence, $\mathbf{m}(\mathbf{x})$ is a second-order polynomial basis function, and $\mathbf{a}$ is the vector of coefficients. The coefficients $\mathbf{\Phi}(\mathbf{x}_j)$ for each destination node are calculated by:

$$\mathbf{\Phi}(\mathbf{x}_j) = \mathbf{m}^T(\mathbf{x}_j)(\mathbf{M}^T\mathbf{W}\mathbf{M})^{-1}\mathbf{M}^T\mathbf{W}$$

The design matrix $\mathbf{M}$ and the weight matrix $\mathbf{W}$ are formed based on the source nodes, where $\mathbf{W}$ is diagonal with Gaussian weights. The interpolation matrix $I_{S_s \to S_d}$ is then constructed as:

$$I_{S_s \to S_d} = \begin{bmatrix} \mathbf{\Phi}(\mathbf{x}_1) \\ \mathbf{\Phi}(\mathbf{x}_2) \\ \vdots \\ \mathbf{\Phi}(\mathbf{x}_{n_d}) \end{bmatrix}$$

To reduce computational complexity, a local interpolation is applied by considering only the k_n nearest neighbors for each destination node. The optimal number of neighbors was found to be $k_n = 10$, striking a balance between minimizing reconstruction errors and managing computational requirements efficiently.

It is worth remarking that this interpolated matrix is of non-square size $n_s \times n_d$ and largely sparse, with only k_n non-zero values in each row. Consequently, with regards to executing the inverse interpolation in the decoder phase, this matrix is not invertible. Thus, it is necessary to compute two independent interpolation matrices: $I_{S_s \to S_d}$ and $I_{S_d \to S_s}$. The entire interpolation process, from fine grid to coarse grid and back, yields an average error of 1.9%. However, this error does not accumulate during these steps, as the subsequent network layers learn to mitigate any potential accumulation, preserving accuracy throughout the process. Introducing errors during training can also act as a regularisation technique that enhances the model generalisation capabilities, and prevents overfitting to specific patterns.

7.3 Test Cases

Two test cases of increasing physical complexity are examined using the same sampling strategy and CFD setup as discussed in Chapter 6. Briefly, Mach number M and angle of attack (AoA) vary over $[0.70, 0.84]$ and $[0, 5]$ [deg], respectively, covering transonic conditions. The 70 samples were divided into 60% for training, 20% for validation, and 20% for testing.

Following dataset generation, preprocessing and normalisation to the range $[-1, 1]$ were performed before inputting the data into the GB-AE-GCN model. The rest of the section explores the model predictive capabilities for distributed quantities and integral loads across different test cases. optimised architectures are detailed in A.5.

7.3.1 Wing–only Model

In this first test case, the BSCW is considered (refer to Chapter 4.5). Unlike the structured-mesh configuration described in Chapter 6, here an unstructured grid is employed to challenge the FCNN model while reflecting practical aerospace applications. Specifically, a grid comprising approximately $8.4 \cdot 10^6$ elements and 86,840 surface elements was generated, with a target $y^+ = 1$ to ensure sufficient resolution of both the boundary layer and the shock wave. The computational domain extends 100 chord lengths from the wing to the farfield. An impression of the grid can be obtained from Figure 7.3.

The performance of the GB-AE-GCN model on the BSCW test case is shown through the `MAE` calculated for C_P and C_F at each surface point in the test set mesh. As depicted in Figure 7.4, the errors for both predictions are considerably small, with the selected ranges serving solely to offer a visual depiction of areas where the model faces challenges in prediction. The errors are minimal across the entire surface, except for a localized region near the shock wave.

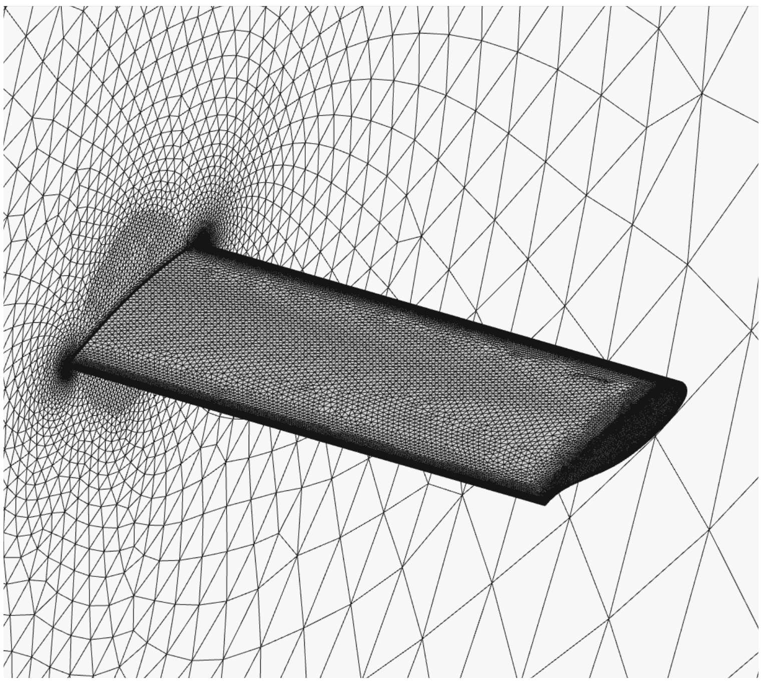


FIGURE 7.3: Impression of the BSCW unstructured grid.

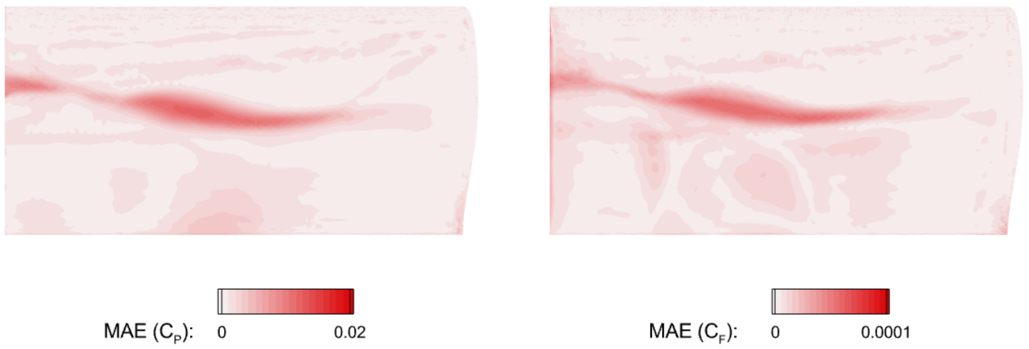


FIGURE 7.4: Mean absolute error (MAE) of C_P and C_F computed across every point in the mesh of the test set for BSCW test case using the GB-AE-GCN model.

Figure 7.5 illustrates the percentage errors in $[C_L, C_D, C_M]$ across different Mach numbers and angles of attacks. Remarkably, the GB-AE-GCN model predictions exhibit high accuracy for all coefficients, even for data points located far from the training set. This indicates the model robustness in extrapolating beyond the provided data points. Aerodynamic coefficients were calculated using a reference chord length of 0.4064 [m] and surface of 0.3303 [m^2], and derived by integrating the pressure coefficient distribution and the skin friction coefficient distribution over the entire wing surface. C_M was calculated with respect to 30% of the chord, accounting for the rigid mounting system of the BSCW, which induces pitch oscillations around this specific location.

The comparison of pressure coefficient contours between the CFD data and the GB-AE-GCN predictions, along with the FCNN results, is presented in Figure 7.6. This comparison is for the test case with the highest error of the GB-AE-GCN model at $M = 0.714$

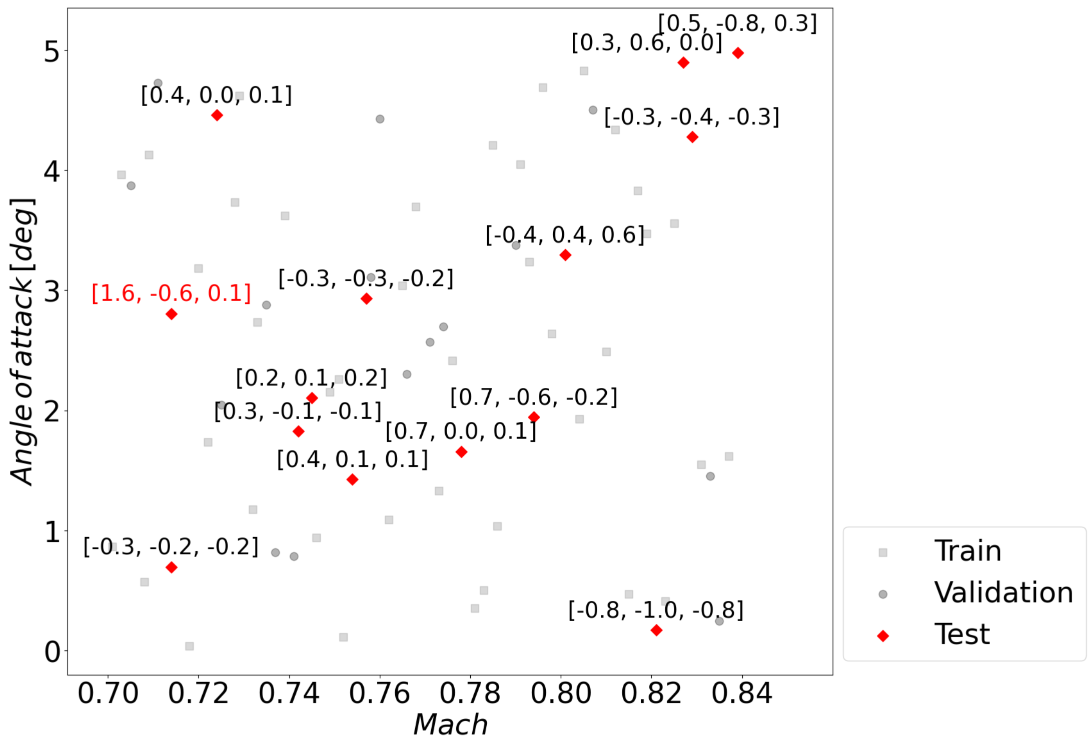


FIGURE 7.5: Errors % in $[C_L, C_D, C_M]$ on the test samples across varying Mach numbers and angle of attacks for BSCW test case using the GB-AE-GCN model.

and $AoA = 2.807$ [deg]. At these transonic conditions the suction side exhibits a swept shock, whose curvature and chordwise position vary with span due to local thickness, sweep and boundary layer state. GB-AE-GCN reproduces both shock location and intensity with high fidelity, including the upstream high-C_P plateau and the steep jump across the wavefront. By contrast, FCNN exhibits mild phase errors and a broader footprint of the shock, consistent with its tendency to smooth sharp gradients. The advantage of GB-AE-GCN is attributable to its mesh-aware latent representation: graph convolutions preserve neighbourhood relations on the unstructured surface, allowing localised features (shock fronts, corner flows near the root) to be encoded effectively.

A similar picture emerges from the skin-friction countour in Figure 7.7. Downstream of the shock, shock-boundary layer interaction produces a C_F reduction associated with separation or strong thickening, bounded by a reattachment ridge that curves spanwise with shock sweep. GB-AE-GCN delineates both the onset and lateral extent of this low-C_F patch more sharply than FCNN, particularly where spanwise pressure communication intensifies near the inboard region. The improved fidelity suggests that the gradient-based encoder/decoder helps recover fine-scale shear features, while the multi-resolution graph hierarchy retains long-range coherence imposed by the finite wing.

Sectional cuts in Figures 7.8 and 7.9 reinforce these observations. Around 20% span, where the shock is strongest, the GB-AE-GCN captures the sharp C_P peak at the shock

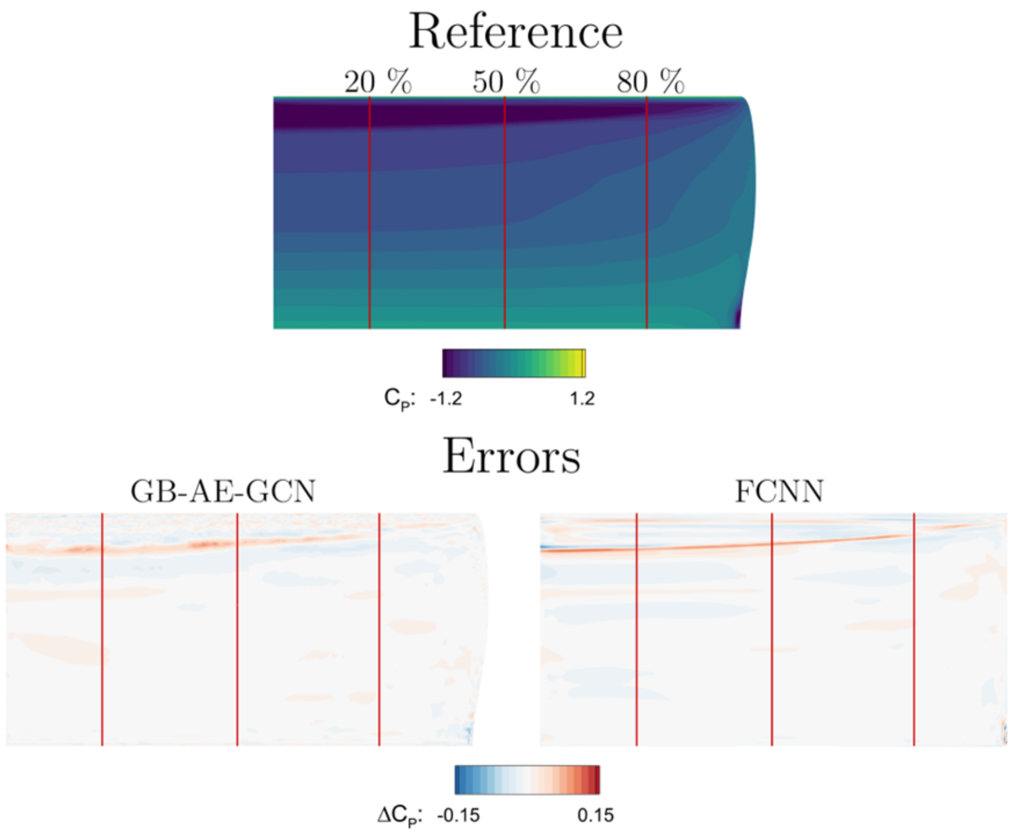


FIGURE 7.6: Prediction of the pressure coefficient contour on the upper surface for BSCW test case at $M = 0.714$ and $AoA = 2.807$ [deg].

foot and the downstream recovery, together with the C_F drop and subsequent rise across reattachment. FCNN follows the global trend but underestimates the peak amplitude and slightly misplaces the jump, indicative of phase error in regions of rapid chordwise variation. Moving outboard, tip-induced upwash perturbs the shock curvature and modulates both pressure and shear; GB-AE-GCN tracks this spanwise evolution more consistently, whereas FCNN field remains overly smoothed.

To provide a quantitative evaluation of the models, we computed MAPE and $R2$ scores on the test set, which are reported in Table 7.2 for the GB-AE-GCN and FCNN models. The MAPE was computed by averaging the absolute error of each prediction calculated by our GB-AE-GCN architecture within the test set. This prediction error was determined by weighted averaging the errors at each grid point, considering the corresponding cell area and normalizing with respect to it. The results show the good performance of the GB-AE-GCN model, with low MAPE values of 0.7712 for C_P and 0.3828 for C_F. In comparison, the FCNN method produced higher errors of 0.9382 for C_P and 1.1091 for C_F. The high $R2$ scores for GB-AE-GCN (0.9728 for C_P and 0.9745 for C_F) further confirm its accuracy in predicting flow features on unstructured grids. This superior performance is attributed to the capability of the GB-AE-GCN to capture nonlinearities in complex flowfields more effectively than the simpler FCNN method, thanks to its ability to incorporate information from neighboring nodes (with GCN layers) and nearby regions

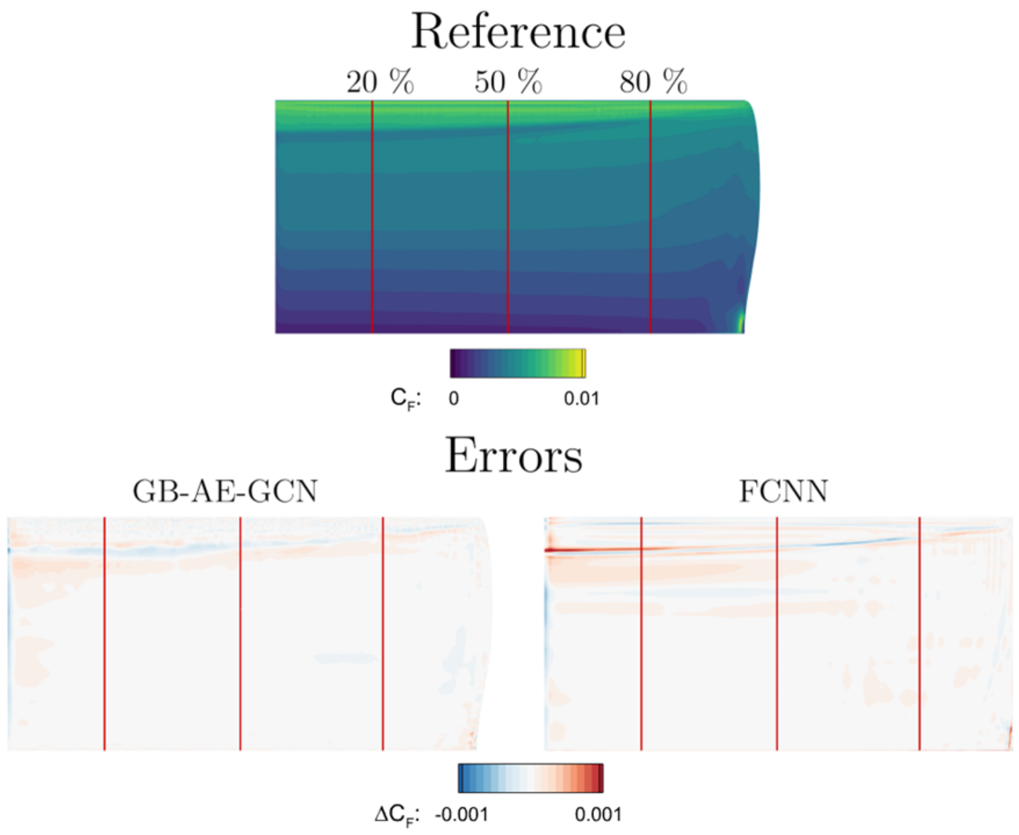


FIGURE 7.7: Prediction of the skin friction magnitude coefficient contour on the upper surface for BSCW test case at $M = 0.714$ and $AoA = 2.807$ [deg].

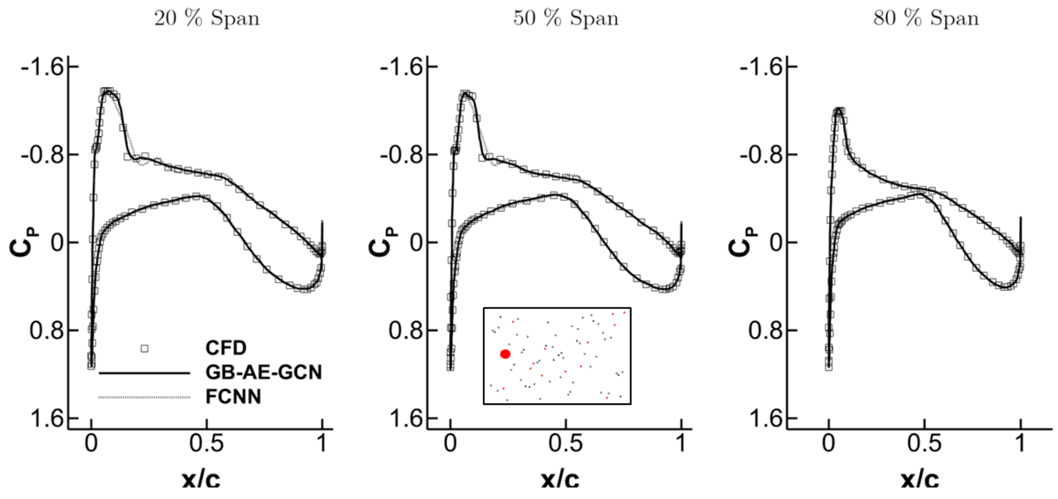


FIGURE 7.8: Pressure coefficient sections of BSCW at $M = 0.714$ and $AoA = 2.807$ [deg].

(with pooling operations).

To ensure the robustness of the results, we studied the influence of random initialisation of the network weights on the variance of `MAPE` and $R2$ scores. The GB-AE-GCN model was trained 30 times with the same hyperparameters, and the resulting variance was minimal, with 0.011 for `MAPE` on C_P and 0.008 for C_F. The variance for $R2$ was

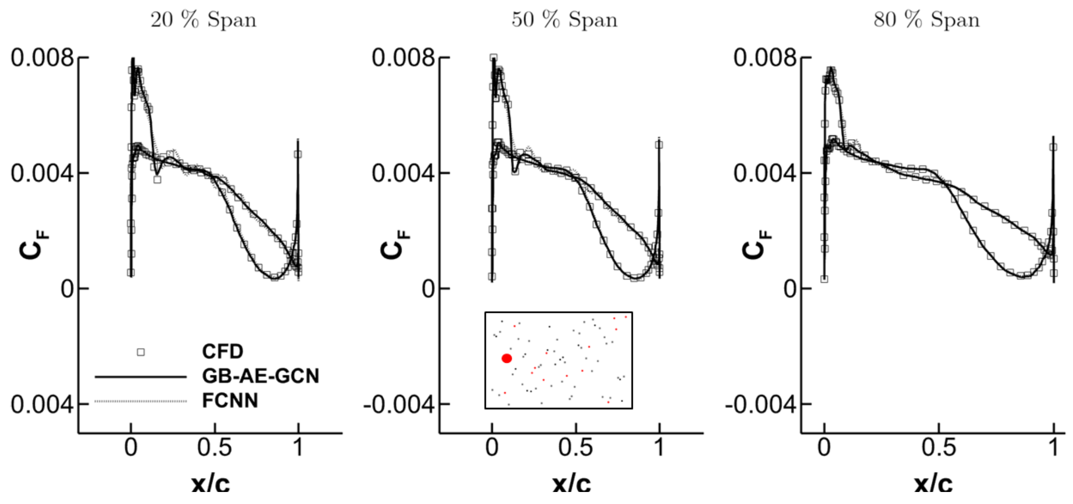


FIGURE 7.9: Skin friction coefficient sections of BSCW at $M = 0.714$ and $AoA = 2.807$ [deg].

similarly low, at 0.0039 for C_P and 0.0031 for C_F. These results indicate that the model performance is stable and consistent across different initialisations.

	GB-AE-GCN		**FCNN**	
	MAPE	$R2$	**MAPE**	$R2$
C_P	0.7712 ± 0.011	0.9728 ± 0.0039	0.9382	0.9304
C_F	0.3828 ± 0.008	0.9745 ± 0.0031	1.1091	0.9185

TABLE 7.2: MAPE [%] and $R2$ on the test set for wing-only model.

7.3.2 Wing–fuselage Model

The second test case is the NASA CRM, introduced in Section 6.3.3. This model includes a conventional low-wing configuration and a fuselage typical of wide-body commercial aircraft. The computational mesh used corresponds to that presented in Chapter 6, with an unstructured layout of 78,829 surface elements.

This test case poses a complex challenge for our GB-AE-GCN model due to the complex geometry, physics and grid configuration. The `MAE` of C_P and C_F computed across every point in the mesh of the test set is depicted in Figure 7.10. This visualisation provides insight into the regions where the model struggles most to accurately represent the flow physical behaviour. Interestingly, the errors are generally minimal across the entire surface, except for a localized region near the wing-fuselage junction and between the kink of the wing and its tip. Nonetheless, the broadly distributed small errors suggest that the model effectively captures the nonlinearities inherent in the system.

Figure 7.11 shows the percentage errors in $[C_L, C_D, C_M]$ on the test samples across varying Mach numbers and angle of attacks. Notably, the predictions demonstrate overall accuracy across all coefficients, even for points distant from the training samples. This

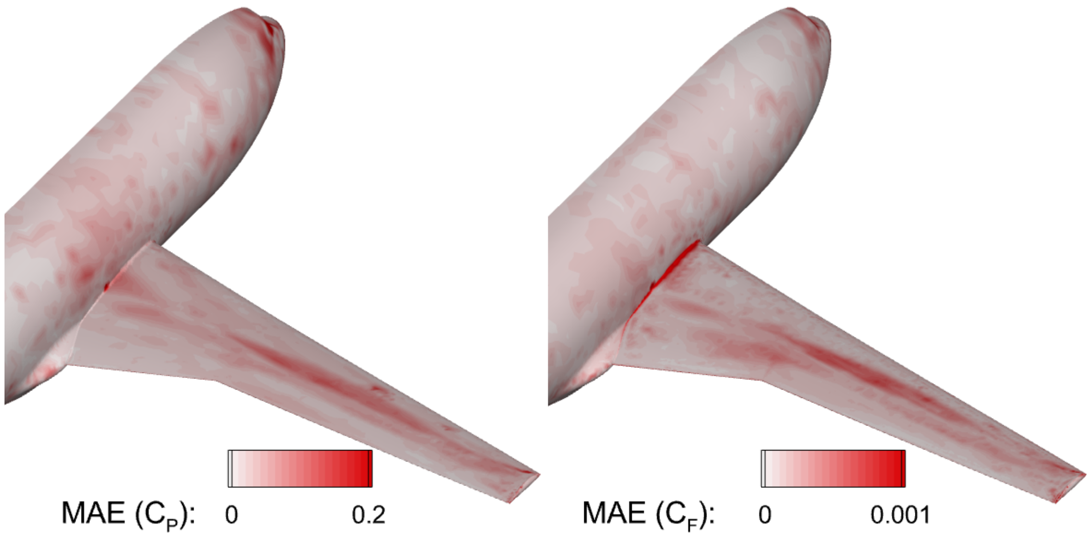


FIGURE 7.10: MAE of C_P and C_F computed across every point in the mesh of the test set for CRM test case using the GB-AE-GCN model.

suggests a robust performance of the model in extrapolating beyond the known data points. A chord of 0.1412 [m] and a scaling area of 0.1266 [m^2] were considered for aerodynamic coefficients calculation. C_M was computed with respect to 25% of the wing mean aerodynamic chord.

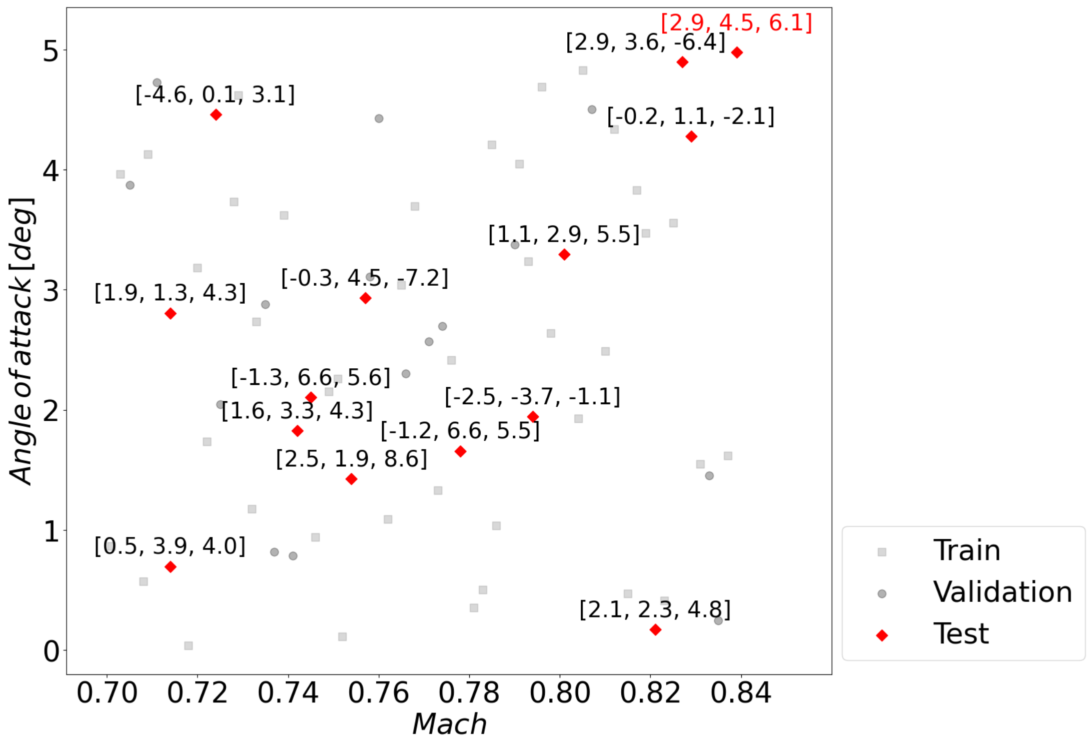


FIGURE 7.11: Errors % in $[C_L, C_D, C_M]$ on the test samples across varying Mach numbers and angle of attacks for CRM test case.

For reference, we also include results from a simpler graph-based method, AE-MM-GCN [233]. This model combines an autoencoder GCN with multi-mesh pooling/unpooling layers, but relies on a single-step grid-density pooling strategy and a standard reconstruction loss, without incorporating physics-based constraints, gradient-aware pooling or hyperparameter optimisation as in our GB-AE-GCN. As a result, AE-MM-GCN provides a useful intermediate comparison: it improves upon fully connected surrogates by directly handling unstructured meshes, yet it remains less accurate than GB-AE-GCN, especially in high-gradient regions.

To provide a comparative analysis, Figures 7.12 and 7.13 present the surface pressure and skin friction contours for the test set with the highest prediction error in the GB-AE-GCN model at $M = 0.839$ and $AoA = 4.975$ [deg]. Despite being the worst performing case, the GB-AE-GCN model still shows excellent agreement with the CFD data, capturing the flow characteristics across the aircraft with high accuracy.

The principal flow mechanisms at $M \approx 0.84$, moderate incidence are: (i) a swept shock on the upper wing with spanwise curvature; (ii) shock-boundary layer interaction leading to local separation and a reattachment ridge; and (iii) a strong junction vortex originating from the wing-fuselage connection that perturbs the inboard pressure field and modulates the shock. These coupled effects generate sharp pressure and shear gradients and spanwise variability that are intrinsically difficult for purely data-driven surrogates. In regions dominated by attached flow, the ROMs provide accurate reconstructions, but Figures 7.12 and 7.14 reveal discrepancies near the wing-fuselage junction and the planform kink, where vortex-shock coupling drives rapid spatial phase shifts. Here, FCNN shows broadened shock footprints and underpredicted suction peaks, while the AE-MM-GCN [233] reduces these errors relative to FCNN yet still lags GB-AE-GCN in localising the shock and matching its curvature. By contrast, the GB-AE-GCN resolves steep chordwise variations while maintaining spanwise coherence, yielding improved prediction of the suction peak, shock position and curvature evident in the sectional comparisons of Figure 7.14.

The skin-friction distributions (Figures 7.13 and 7.15) reinforce this interpretation. Downstream of the shock, shock-boundary layer interaction thickens the boundary layer and lowers C_F; additionally, the junction vortex impinges on the inboard wing, producing strong streamwise gradients and modulating the reattachment point. GB-AE-GCN accurately delineates the low-C_F patch and the reattachment line, whereas FCNN underestimates the C_F reduction and slightly mispredicts the point.

Building on this detailed analysis of the performance of GB-AE-GCN model under its most challenging prediction, we now provide an overview of the performance across the entire test set. To evaluate the models comprehensively, we computed `MAPE` and $R2$ scores only for the wing region, where the flow exhibits strong nonlinearities. The CRM model includes large areas of mostly linear flow, especially over the fuselage, which we

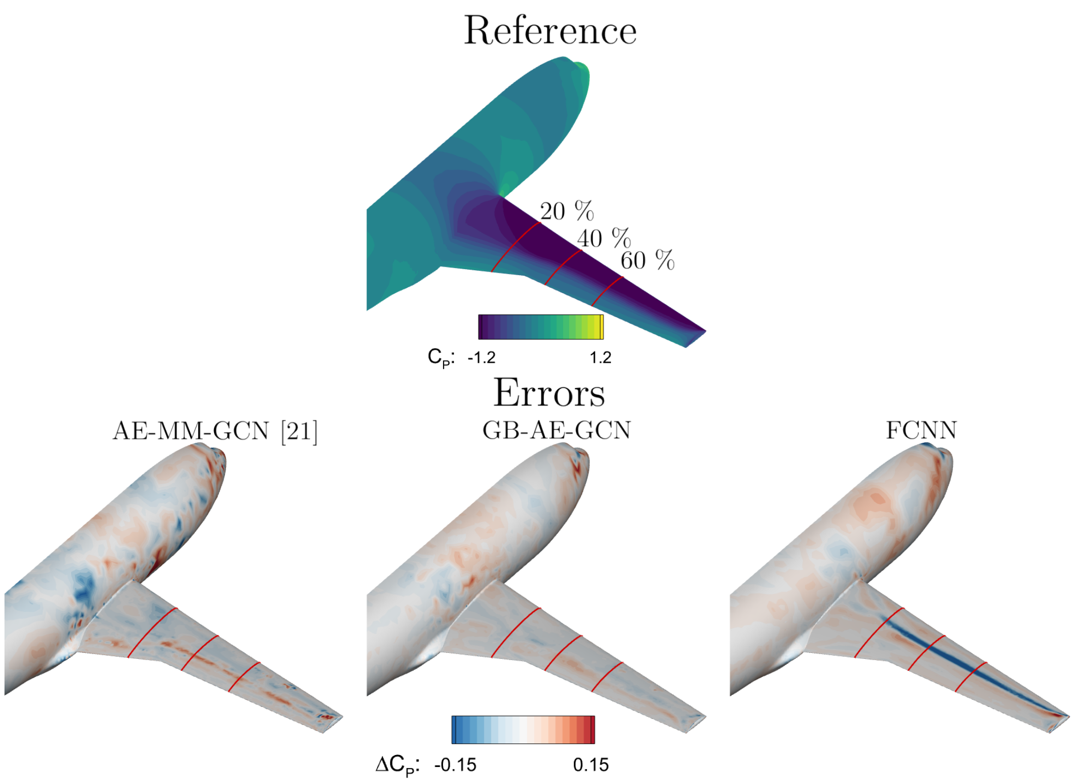


FIGURE 7.12: Prediction of the pressure coefficient contour on the upper surface for CRM test case at $M = 0.839$ and $AoA = 4.975$ [deg].

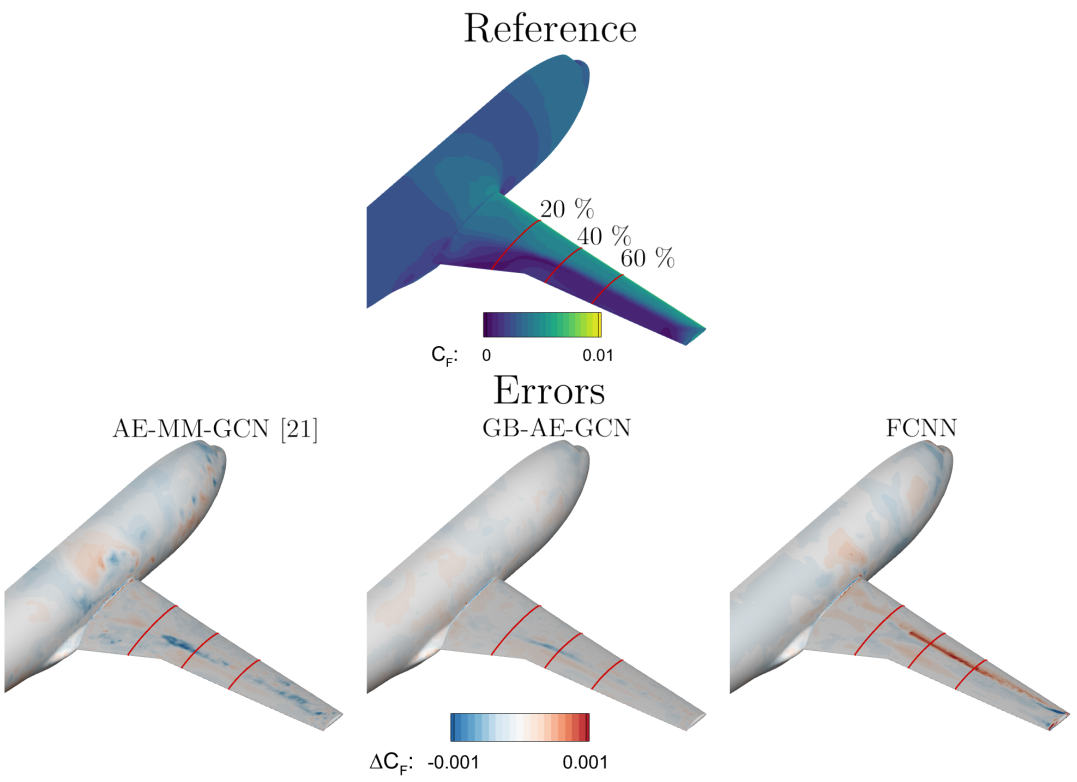


FIGURE 7.13: Prediction of the skin friction magnitude coefficient contour on the upper surface for CRM test case at $M = 0.839$ and $AoA = 4.975$ [deg].

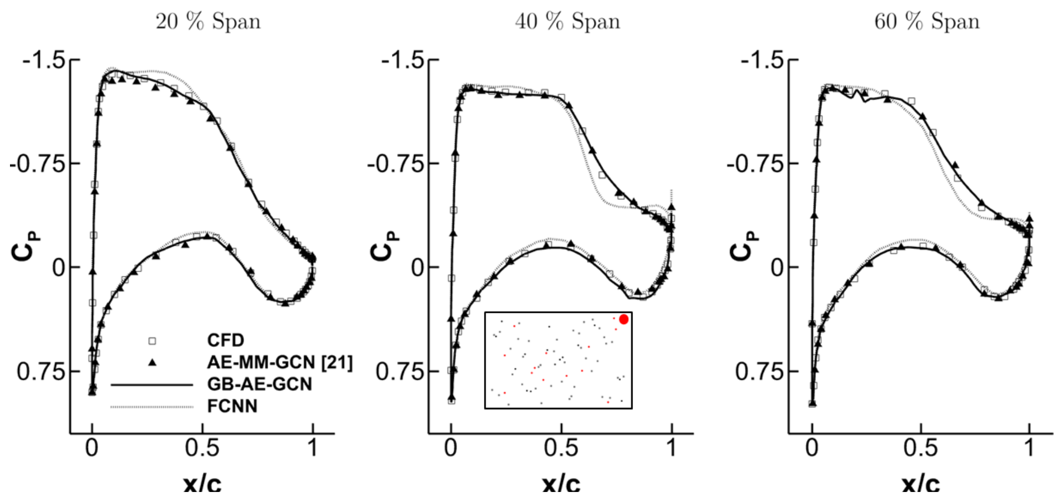


FIGURE 7.14: Pressure coefficient sections of CRM at $M = 0.839$ and $AoA = 4.975$ [deg].

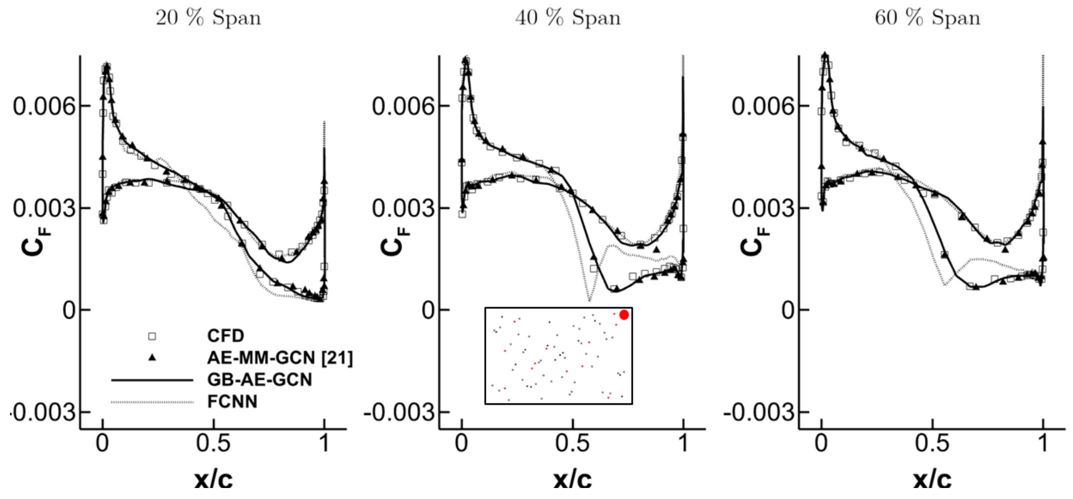


FIGURE 7.15: Skin friction coefficient sections of CRM at $M = 0.839$ and $AoA = 4.975$ [deg].

excluded from the analysis. Table 7.2 presents the MAPE and $R2$ results for the GB-AE-GCN, FCNN, and AE-MM-GCN [233] models. The GB-AE-GCN model achieved a MAPE of 0.8876 for C_P and 0.2402 for C_F, outperforming both the FCNN and AE-MM-GCN [233] models. The lower MAPE for C_F is particularly significant, indicating the model ability to capture skin friction details more accurately, especially in regions with complex flow gradients. The $R2$ values further confirm this performance, with the GB-AE-GCN achieving $R2$ values of 0.9353 for C_P and 0.9650 for C_F, reflecting its accuracy in predicting aerodynamic features. In comparison, the FCNN and AE-MM-GCN [233] models show lower $R2$ values, highlighting their difficulties in capturing sharp flow features.

To further assess the robustness of the GB-AE-GCN model, we evaluated how random initialisation affects the variance in MAPE and $R2$ scores. After training the model 30 times with the same hyperparameters, we observed a variance of 0.07 in MAPE for C_P and 0.05 for C_F. For $R2$, the variance was 0.014 for C_P and 0.019 for C_F. These results indicate that, despite the inherent randomness in initialisation, the model consistently produces reliable and accurate predictions, reinforcing its robustness across different training runs.

	GB-AE-GCN		**GCN-MM-AE [233]**		**FCNN**	
	MAPE	$R2$	**MAPE**	$R2$	**MAPE**	$R2$
C_P	0.8876 ± 0.07	0.9674 ± 0.014	1.4051	0.9353	1.7385	0.8942
C_F	0.2402 ± 0.05	0.9737 ± 0.019	0.5737	0.9550	1.1942	0.9005

TABLE 7.3: MAPE [%] and $R2$ on the test set for wing-body model.

7.4 Computational Cost Analysis

A detailed computational cost analysis was conducted to evaluate the efficiency of the implemented GB-AE-GCN model in comparison to the high–order approach, as outlined in Table 7.4. Each CFD simulation was performed on a high-performance computing system with an Intel Skylake-based architecture utilizing 3 nodes with 40 CPU cores each, typically consuming around 450 CPU hours per run, with the entire dataset generation requiring approximately 31,500 CPU hours. Conversely, employing the ROM enables prediction for a single sample in approximately 1 second on a local machine, resulting in a computational saving exceeding 99%.

Therefore, it is essential to consider the high computational cost associated with each high–fidelity simulation used for generating the dataset. Adopting a philosophy aimed at minimizing the amount of training data necessary for developing an accurate model is extremely important.

The training process was executed on an Intel XEON W-2255 CPU with a NVIDIA RTX A4000 GPU, ensuring efficient utilisation of computational resources.

Test case	**CFD (CPU hours)**		**GB-AE-GCN (GPU hours)**		
	Simulation		optimisation	Training	Prediction
	(70 runs)	(1 run)	(30 trials)	(1 model)	(1 sample)
BSCW	35,000	500	27	1.4	0.0003 ($\sim 1s$)
NASA CRM	28,000	400	28	1.5	0.0003 ($\sim 1s$)

TABLE 7.4: Computing cost comparison between GB-AE-GCN model and CFD for the two test cases.

7.5 Conclusions

This chapter highlighted the effectiveness and robustness of the implemented GB-AE-GCN model in delivering precise predictions within a parameter space for various test scenarios, featuring complex geometries and diverse physical phenomena. Through convolutional and pooling operations, the model efficiently influences predictions at individual nodes based on their neighbors, while also enabling information propagation to distant nodes during spatial reduction. Additionally, the model can directly process input grids without requiring preprocessing, greatly simplifying the modelling process.

In terms of performance, the GB-AE-GCN model consistently outperformed both the FCNN approach and literature models [233] across various test cases, particularly in capturing complex nonlinear flow phenomena such as shock waves and boundary layer separations. The GB-AE-GCN superior ability to handle sharp flow gradients and complex geometries, especially in high-gradient regions, is reflected in its lower MAPE and higher $R2$ scores compared to the other models. The pressure-gradient-based coarsening, fast connectivity reconstruction, physics-informed loss function, and network optimisation played key roles in enabling our model to better capture nonlinearities and critical flow features that the other methods struggled with.

In addition to flexibility, the model is designed to ensure scalability, enabling it to efficiently operate on larger grids due to the localized nature of GCN operations. As the grid size increases, certain adjustments become necessary, particularly regarding the number of layers in the network. Larger grids may demand deeper networks to capture the complex interactions across the expanded spatial domain. Additionally, increasing the number of pooling operations is important to ensure information propagation across more distant nodes, enabling them to communicate more effectively. This increased pooling facilitates the model ability to capture long-range dependencies, but it also requires careful tuning to balance computational efficiency with model accuracy.

From a broader thesis perspective, these results contribute in several ways:

1. **Capturing Nonlinear Transonic Phenomena**
   Leveraging graph-based operations, the model accurately represents complex shock interactions and boundary layer behaviour over surface vector fields, aligning with the thesis' emphasis on reducing computational effort while preserving the essential physics of steady nonlinear flows (Research Objective O3).

2. **Scalability for Industrial and Research Applications**
   The localized neighbourhood aggregation of graph convolutional layers allows the model to scale efficiently to larger grids, requiring only moderate adjustments (e.g., deeper networks, more pooling layers). This aligns with the thesis aim of

developing practical, high-impact reduced-order models suitable for large-scale 3D configurations.

3. **Flexibility Beyond Aerospace**
   By representing complex aerodynamic meshes as graphs, this framework can be adapted to other scientific and engineering domains that rely on unstructured, non-homogeneous data. The node-centric approach, which aggregates information based on local connectivity, provides robust and consistent predictions for different topologies.

These findings highlight the model ability to capture complex transonic flow physics and provide a pathway for further adaptations - in particular, extensions to transient problems and variations in geometry. Building on the results of the steady-flow formulation, Chapter 8 explores the incorporation of temporal dynamics in a similar GCN framework, thereby broadening the scope of the method for applications involving time-dependent phenomena. In parallel, Chapter 9 demonstrates how the same underlying technique can be adapted to different wing shapes, thus extending the scope of the model to a wide range of aerodynamic configurations.

Future research should also focus on improving the generalisation capabilities of GCN-based autoencoders. For example, adopting a transfer learning approach could improve model performance over a wider range of geometries and flow conditions. In particular, integrating information from coarser meshes could increase the volume of training data without significantly increasing offline computational cost, and subsequently refine predictions using finer meshes where higher accuracy is needed. In addition, uncertainty quantification methods could be implemented to identify regions within the design space with the highest uncertainty. This adaptive sampling strategy would inform the selection of additional high-fidelity simulations, thereby improving overall prediction reliability. In addition, it is recommended that a learnable pooling layer be incorporated into the network architecture, delegating feature selection directly to the model. This approach would ensure that the network autonomously identifies and retains the most important flow features in the reduced latent space.

Chapter 8

Spatio-temporal Graph Convolutional Autoencoder for Unsteady Transonic Flowfields

This chapter presents a framework for predicting unsteady transonic wing pressure distributions resulting from pitch and plunge motion. The approach integrates graph convolutional networks with autoencoder architectures for dimensionality reduction, and graph-based temporal layers to model time dependencies. High-dimensional pressure data are compressed into a latent space via the autoencoder, preserving key aerodynamic features while enabling efficient representation. Within this space, temporal graph layers predict future pressure fields based on past data, capturing dynamics and improving accuracy. Addressing Research Objective O4, this chapter extends the spatial surrogate modelling of Chapter 7 to unsteady regimes. By incorporating temporal recurrence, the framework enables prediction of transient aerodynamic responses on unstructured meshes. Four temporal schemes are evaluated, with the spatio-temporal graph convolutional network showing the best performance due to combined time-space convolution. The framework is benchmarked against Dynamic Mode Decomposition with control and validated using the BSCW test case at Mach 0.74.

8.1 Introduction

Accurately predicting unsteady aerodynamic forces in transonic flight conditions remains a fundamental challenge in aerospace engineering. Traditional CFD simulations, while precise, are computationally expensive, making real-time or iterative design processes impractical [37]. This study proposes a ML-driven framework that significantly reduces computational costs while maintaining predictive accuracy.

Recent advancements in ML have demonstrated significant potential in addressing this challenge by enabling rapid and accurate predictions of complex flow dynamics. Early applications of ML in fluid mechanics relied primarily on FCNNs and CNNs, both of which showed strong predictive capabilities for structured grid environments [55, 297, 160]. However, many aerodynamic applications rely on unstructured meshes, which require more advanced learning architectures capable of handling complex data structures such as GNNs [182, 383]. In particular, GCNs have been effectively used in transonic aerodynamics due to their ability to process irregular spatial structures [118, 169, 255, 264, 294]. By applying localized convolution operations directly on graphs, GCNs avoid the need to re-sample unstructured meshes onto regular grids [383], thus preserving important geometric details in the data. However, capturing temporal dependencies in unsteady flows remains an open challenge.

Projection-based ROMs, such as POD-Galerkin, have long provided efficient surrogates by projecting the governing equations onto a low-dimensional subspace spanned by energy-optimal modes [194, 415, 358]. Despite their interpretability and negligible online cost, POD-Galerkin formulations are typically intrusive: they require access to the discretised operators (residuals, fluxes, Jacobians) for projection and, in transonic regimes, often depend on empirical closures or stabilisation to cope with shocks, separation and strong convection-reaction imbalance. These requirements limit portability to proprietary or black-box solvers and complicate reuse across distinct discretisations and grids. Moreover, when accurate operators are unavailable (e.g., experimental data only, restricted industrial codes), the projection step itself may be infeasible, motivating alternatives that operate directly on snapshot data.

Sparse Identification of Nonlinear Dynamics (SINDy) involves regressing state time-derivatives onto a sparse library of candidate nonlinearities to seek parsimonious governing equations [48, 59, 295]. In fluids, SINDy has been combined with low-dimensional coordinates (e.g., POD coefficients) or with control inputs (SINDYc) to obtain compact, interpretable models. While attractive for interpretability, SINDy presupposes: high-quality state trajectories (and, typically, reliable time-derivatives), a sufficiently expressive but well-designed function library, and dynamics that are well-represented by smooth bases, assumptions strained by transonic phenomena with shock-induced discontinuities, regime switches and hysteresis. Derivative estimation on noisy data and library misspecification can degrade robustness, and extending discovered ODEs uniformly across large parametric domains (e.g., Mach-AoA) is non-trivial. These considerations, together with limited access to solver internals, further motivate non-intrusive, equation-free surrogates that learn spatio-temporal mappings directly from data.

Another major obstacle is the high dimensionality of the input data. As with CNN-based methods for image recognition, aerodynamic networks must process large amounts of data describing flow variables at numerous mesh points. Such high dimensionality increases both model complexity and computational cost. Previous studies have shown that dimensionality reduction can be effectively achieved by AE architectures that significantly compress the input space while preserving essential flow features [137, 232, 294, 298]. Through encoding and decoding, AEs learn an efficient representation of complex data, reducing computational complexity without losing important information [150, 359].

Capturing temporal dependencies is critical for accurate modelling of unsteady aerodynamics. RNNs are a natural fit for such tasks because they track evolving patterns through hidden states [67, 152]. However, simple RNNs often struggle with long-term dependencies due to vanishing or exploding gradients [299]. LSTM networks and GRUs mitigate these problems by using gating mechanisms that intelligently preserve relevant information [67, 152]. Both have been successfully applied to time series prediction in aerospace domains, such as turbulence modelling [47, 205] and structural response prediction [225, 231, 366]. More recently, attention mechanisms have gained traction, transforming time-series prediction by allowing models to selectively focus on critical regions within the input sequence, dynamically weigh the importance of different time steps, and improve the capture of high-frequency or transient phenomena [65, 66, 357]. Applied to fluid dynamics, attention layers improve the accuracy and robustness of long-term predictions [66, 98, 138]. Similarly, STGCNs extend GCNs by applying temporal convolutions over graph-structured data [18]. This approach processes entire sequences in parallel and introduces specialized filters in the time dimension, capturing multi-scale temporal patterns more effectively than purely recurrent architectures.

Guided by these considerations, we deliberately target purely data-driven, non-intrusive surrogates that: (i) do not require access to the discretised governing equations, (ii) operate natively on unstructured meshes, and (iii) remain portable across solvers and datasets. This work builds on steady-state transonic flow prediction (Chapter 8) by integrating an autoencoder architecture with graph-based temporal learning methods, systematically evaluating four temporal models, LSTM networks, GRUs, attention-based architectures, and Spatio-Temporal Graph Convolutional Networks (STGCNs), to identify the most effective approach for time-dependent aerodynamic forecasting. By leveraging the strengths of GCNs for handling unstructured grids, autoencoders (AEs) for efficient dimensionality reduction, and temporal layers for capturing dynamic dependencies, the proposed framework enhances the prediction of unsteady surface pressure distributions over a transonic wing. Applied to the BSCW test case at Mach 0.74, where complex aerodynamic interactions such as shock waves and boundary layer separation play a significant role, this approach demonstrates that STGCNs outperform other temporal models, achieving accuracy comparable to high-fidelity CFD while significantly reducing prediction time. To provide a clear reference for performance gains, we benchmarked the proposed framework against a DMDc model due to its relevance and maturity in this field.

8.2 Methodology

This section outlines the proposed implemented approach. It begins with an overview of the spatio-temporal graph convolutional autoencoder framework, denoted as AeroNet, then provides an in-depth description of each component of the model.

8.2.1 AeroNet

The proposed AeroNet framework integrates a GCN-based AE architecture with a temporal prediction layer to model and forecast wing pressure distributions for subsequent timesteps. The encoding and decoding modules operate with graph nodes based on the pressure-gradient distribution values across the wing surface, performing pooling and unpooling operations on the input, respectively. Initially, a pre-trained AE is used to compress the pressure distribution data into a lower-dimensional representation, preserving fundamental features while reducing computational complexity. This pre-training step reduces the full model training time and computational costs, enhancing the overall efficiency of the prediction process. After the pooling operation, the reduced-space representation is passed through a temporal prediction layer. This layer is designed to capture the temporal dependencies in the data and forecast wing pressure from a series of previous timesteps to a future one. To account for the complexities caused by shock waves and boundary layer separation, which affect force and moment calculations, a penalty term for the pitching moment coefficient C_{M_y} is added to the `MAE` loss function. This addition is represented as $Loss = \texttt{MAE} + \lambda \cdot C_{M_y}$, with $\lambda = 0.01$ for dimensional consistency. The combination of AE, GCN layers, and temporal modelling enables the framework to provide precise and reliable pressure predictions, which are crucial for analysing aerodynamic performance.

A visual overview of the model architecture is presented in Figure 8.1. The model input features include data from the n previous timesteps ($t-1, ..., t-n$): spatial coordinates (x, y, z); pitch $(\theta_{t-n}, \dot{\theta}_{t-n}, \ddot{\theta}_{t-n})$; plunge $(\dot{\xi}_{t-n}, \ddot{\xi}_{t-n})$; and pressure coefficient $(C_{P_{t-n}})$, with $n = 3$. Finally, the output of the model is represented by the pressure coefficient (C_{P_t}) at the current timestep t. The choice of this sequence length ensures that the model has access to sufficient temporal context to capture the evolution of unsteady aerodynamic features, such as flow separation and shock dynamics, while avoiding the inclusion of redundant or excessive data, which would increase computational complexity without significantly improving accuracy.

Building on this, we developed two different types of architecture: a feedforward model and an autoregressive-moving-average (ARMA) model. In the case of the feedforward model, the inputs consist solely of the coordinates of the wing surface and the prescribed motion at n previous timesteps casted on each graph node (using only

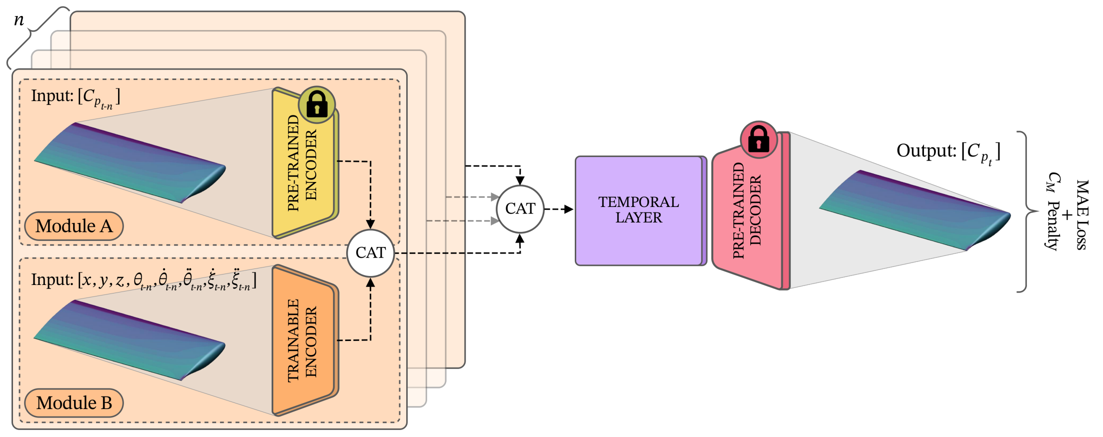


FIGURE 8.1: Overview of the GST GraphNet architecture for predicting wing pressure distributions. Module A represents the autoregressive component, incorporating previously predicted C_P values, while Module B processes spatial coordinates and motion data from previous timesteps.

Module B in Figure 8.1). This model does not incorporate any past predicted pressures into its input, relying purely on the historical spatial and motion data of the wing to make its predictions. Conversely, the ARMA model includes additional information in its input by integrating the pressures predicted at prior timesteps (utilizing both Module A and Module B in Figure 8.1). This autoregressive component allows the ARMA model to potentially capture more complex temporal dependencies by considering the history of its own predictions, aiming to enhance the accuracy of the pressure forecasts. Implementing both models serves to evaluate the trade-offs between simplicity and predictive depth: while the feedforward model offers a simpler, stable approach less prone to error accumulation, the ARMA model is designed to capture complex temporal dependencies and unsteady behaviours, potentially enhancing accuracy under dynamic conditions.

To limit error accumulation in the time-marching scheme, we employed a Back-Propagation Through Time (BPTT) algorithm [299] for the total loss calculation, dividing the dataset into mini-sequences. The model processes each sequence consecutively, accumulating error over time. After processing each sequence, the loss function is applied to update the model parameters through backpropagation. A sequence length of three was chosen based on its performance, yielding the best results.

The architecture and training process for both the ARMA and feedforward models used in this study are described in detail in the Appendix A.6.

8.2.2 Graph Representation

The aerodynamic domain is represented as a cyclic graph, where each grid point serves as a node. Each node carries features such as spatial coordinates (x, y, z), pitch $(\theta, \dot{\theta}, \ddot{\theta})$, plunge $(\dot{\xi}, \ddot{\xi})$, and the pressure coefficient C_P from the previous n timesteps.

Graph connectivity is represented by the adjacency matrix $\mathbf{A}$, where each edge weight e_{ij} is the Euclidean distance between grid points i and j: $e_{ij} = \|\mathbf{x}_i - \mathbf{x}_j\|_2$. To normalize the weights to $(0, 1]$ and include self-loops ($e_{ii} = 1$), the adjacency matrix is augmented: $\tilde{\mathbf{A}} = \mathbf{A} + \mathbf{I}$. Since edges are bidirectional ($e_{ij} = e_{ji}$), $\tilde{\mathbf{A}}$ is symmetric.

Due to the graph sparsity, the adjacency matrix is stored in a memory-efficient Coordinate List (COO) format, where the edge-index matrix contains pairs of node indices, and the edge-weight matrix stores the corresponding weights, with n_e being the number of edges.

8.2.3 Graph Convolutional Networks

GCNs are employed to extract spatial dependencies from the aerodynamic field by propagating information across neighbouring nodes. The spectral graph convolution operator, based on the graph Laplacian, enables efficient feature aggregation and transformation. The specific formulation of GCNs used in this study, including the layer-wise propagation rule and spectral filtering, is outlined in Section 2.5.3 of the Foundations chapter.

8.2.4 Graph-temporal Layers

In our framework, we explored various layers for temporal modelling: GRUs, LSTMs, attention mechanisms, and STGCN layers. Each method offers a distinct way to capture temporal dependencies, with varying level of complexity and performance suited to different contexts. The detailed formulation of eaech graph-temporal layer is reported in Section 2.5.5 of the Foundations chapter.

GRUs are efficient recurrent architectures that utilize update and reset gates to selectively retain past information while reducing computational complexity. LSTMs extend this capability by incorporating memory cells that preserve information over long sequences, making them particularly suitable for capturing extended temporal dependencies in aerodynamic data. Attention mechanisms dynamically assign importance to different time steps, allowing the model to focus on the most relevant historical data for improved forecasting accuracy. STGCN layers, in contrast, integrate graph convolution with temporal convolutions, enabling simultaneous learning of spatial and temporal correlations within aerodynamic flowfields.

8.2.5 Dimensionality Reduction/Expansion Module

The dimensionality reduction and expansion strategy follows the approach introduced in Chapter 7, where redundant regions of the computational domain are discarded while preserving important flow features. Briefly, gradient-based pooling first selects nodes with high pressure gradients, ensuring that shock waves and other highly non-linear phenomena remain well resolved. To reconnect the reduced mesh, Mahalanobis distance is used to maintain proper node adjacency, taking into account the covariance of the distribution and mitigating false connections caused by proximity errors under Euclidean distance.

After constructing this reduced graph, node values are interpolated using MWLS method. A weight function based on spatial proximity drives the interpolation matrix $I_{S_s \to S_d}$, mapping source-grid values S_s to the destination grid S_d. For unpooling (i.e., decoder stage), a separate inverse interpolation matrix $I_{S_d \to S_s}$ is computed. The reader is referred to Chapter 7 for the full derivation and detailed equations underlying these steps.

8.2.6 Pre-trained Autoencoder

Our proposed framework leverages an AE architecture, pre-trained for subsequent integration into the complete model. The pre-training phase involved using C_P data as both input and output to the AE, ensuring the model accurately captures the essential features of the pressure distribution over the wing surface. The training dataset comprised the four signals detailed in Table 8.1.

To enhance the robustness of the AE, a data augmentation technique was employed. Specifically, the dataset was augmented by 30% through the addition of Gaussian noise with 10% standard deviation of the input data. This augmentation strategy was designed to improve the model ability to generalize and handle variability in the pressure distribution data. Skip connections were integrated before each encoding module to facilitate the direct flow of information across the network. These connections allow the model to bypass certain layers, enabling the retention of critical features and mitigating the risk of information loss during the encoding and decoding processes. The network architecture has been optimised using a Bayesian optimisation strategy. A schematic of the pre-trained AE architecture is shown in Figure 8.2.

8.2.7 Dynamic Mode Decomposition with Control

DMDc provides a systematic approach for deriving ROMs from high-dimensional data, explicitly incorporating external inputs. Since this is a well-known methodology in

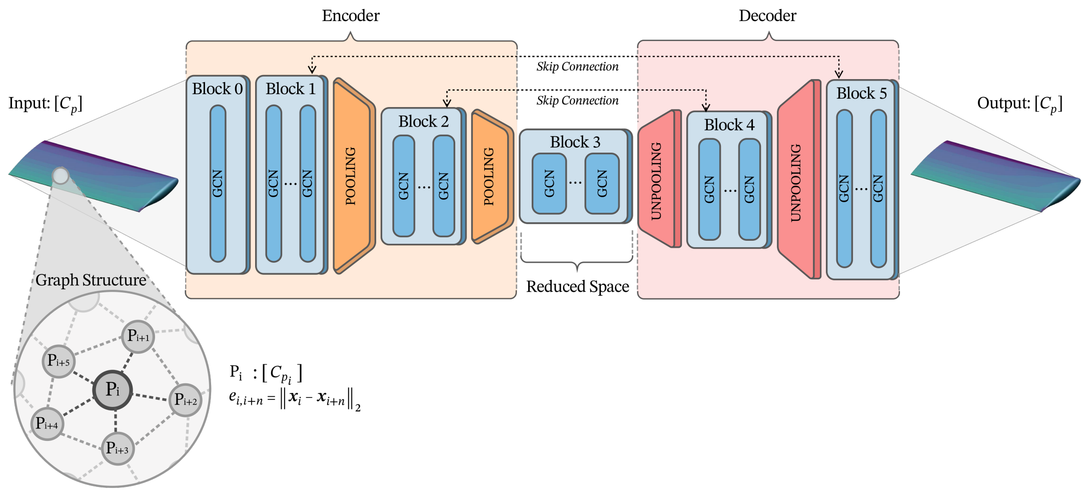


FIGURE 8.2: Schematic of the pre-trained AE architecture for compressing and reconstructing the C_P data within the GST GraphNet framework.

this field, we developed a baseline model to serve as a reference for our GCN-based framework. The full mathematical derivation is given in Chapter 2.1.

The state vector $\mathbf{x}_t$ collects the instantaneous surface pressure coefficients, whereas the control vector $\mathbf{u}_t$ contains the rigid–body motion of the wing, pitch $(\theta_t, \dot{\theta}_t, \ddot{\theta}_t)$ and plunge $(\dot{\xi}_t, \ddot{\xi}_t)$:

$$\mathbf{x}_t = \begin{bmatrix} C_{p,1} & C_{p,2} & \dots & C_{p,m} \end{bmatrix}^T, \qquad \mathbf{u}_t = \begin{bmatrix} \theta_t & \dot{\theta}_t & \ddot{\theta}_t & \dot{\xi}_t & \ddot{\xi}_t \end{bmatrix}^T.$$

The discrete dynamics are approximated by the linear map:

$$\mathbf{x}_{t+1} = \mathbf{A}\mathbf{x}_t + \mathbf{B}\mathbf{u}_t,$$

where $\mathbf{A} \in \mathbb{R}^{m\times m}$ and $\mathbf{B} \in \mathbb{R}^{m\times p}$ are identified from the CFD snapshots following the procedure of Fonzi et al. [112]. The resulting ROM predicts the transonic wing pressure field over time.

8.3 Test Case

Benchmark Super Critical Wing

The selected case study for evaluating our framework is the BSCW, described in Section 4.5. The freestream conditions for this case are defined by a Mach number of 0.74, a Reynolds number of $4.49 \cdot 10^6$, and an initial angle of attack of 0 degree. The wing features a reference chord length of 0.4064 [m] and a surface area of 0.3303 [m^2], with pitching motion occurring around 30% of the chord. It is mounted on a flexible support system that allows two degrees of freedom: pitch θ and plunge ξ. As emphasized in earlier chapters, the transient motion of the shock, the onset of shock-induced boundary layer separation, and the subsequent interaction of these unsteady flow features with the detached boundary layer introduce significant complexity. Such nonlinear effects require advanced modelling strategies to ensure that the framework can reliably capture the resulting aerodynamic behaviour. The computational mesh used corresponds to that presented in Chapter 7, with an unstructured layout of 86,840 surface elements.

The dataset used to train the model was generated with CFD unsteady responses using the URANS formulation with SU2 v7.5.1 [104]. The simulations employed the one-equation Spalart-Allmaras turbulence model for RANS closure. To accelerate convergence, a $1v$ multigrid scheme was used. The JST central scheme with artificial dissipation handled convective flow discretisation, and the gradients of flow variables were calculated using the Green-Gauss method. The biconjugate gradient stabilisation linear solver, along with the ILU preconditioner, was utilized. All URANS simulations started from a steady-state solution, with a timestep of $2 \cdot 10^{-4}$ seconds and a total simulation time of 2 seconds. These values were chosen to ensure a high temporal resolution, capturing rapid aerodynamic variations while keeping the computational cost manageable over the full simulation period. However, due to computational constraints, the timestep was reduced to $2 \cdot 10^{-3}$ seconds through downsampling for model training, resulting in 660 timesteps per signal. This ensures a balance between computational feasibility and quality of aerodynamic data.

Excitation Signal

The Schroeder-phased harmonic signal is used in this study to improve the robustness and generalizability of the model by covering a wide frequency spectrum. These signals are constructed by summing sinusoidal components, with the phases optimised to minimize the overall peak amplitude. This results in an evenly distributed energy spectrum, which is beneficial for training the model to handle different frequency interactions and reduces the risk of overfitting. To cover this wide frequency range with minimal peak amplitude, a total of 9 harmonics are selected, with $M = 9$.

The undamped (US) and damped (DS) Schroeder-phased signals were applied to both pitch $\theta(t)$ and plunge $\xi(t)$. The undamped signal distributes energy uniformly across the spectrum:

$$\theta_{US}(t) = \sum_{m=1}^{M} a_m \sin\left((m+1)\omega_m t + \phi_m\right) \tag{8.1}$$

where a_m is the amplitude of the m-th component, ω_m is the angular frequency, and ϕ_m is the phase. The phase terms are optimised to minimize constructive interference:

$$\phi_m = -\frac{m(m+1)\pi}{M} \tag{8.2}$$

For transient response modelling, the DS signal incorporates amplitude decay:

$$\theta_{DS}(t) = \sum_{m=1}^{M} \left(\left(\frac{a_{\text{end}} - a_0}{t_{\text{end}} - t_0}(t - t_0) + a_0 \right) \sin\left((m+1)\omega_m t + \phi_m\right) \right) \tag{8.3}$$

where a_0 and a_{end} define the amplitude range over the interval $[t_0, t_{\text{end}}]$, ensuring that by the final timestep, the amplitude reduces to $0.1a_0$.

Dataset Structure

A total of 12 time-varying simulations were run, divided into training, test, and validation sets as described in Table 8.1. The training set consists of damped Schroeder-phased harmonic (DS) simulations with varied combinations of the parameters: reduced frequency for pitch (κ_θ), maximum pitch amplitude (a_θ), reduced frequency for plunge (κ_ξ), and maximum plunge amplitude (a_ξ), ensuring that the model learns from diverse conditions where these variables exhibit both positive and negative values (Figure 8.3). This variation helps the model understand the complex interactions between the parameters. The test set extends this diversity by including additional combinations and signal types, specifically undamped Schroeder-phased harmonic (US) and cases focused solely on θ or ξ variations. The signs associated with pitch and plunge indicate the initial directions of oscillation: a positive pitch corresponds to a nose-up rotation, while a positive plunge indicates downward motion. This approach allows us to accurately assess the model accuracy and sensitivity in predicting the effects of individual parameters, ensuring that it performs well across different conditions. The use of Schroeder signals is motivated by their ability to effectively cover a broad frequency spectrum while minimizing peak amplitudes, which enhances the model robustness and ability to generalize. The validation set is designed to evaluate the model generalizability to new and unseen data. It includes a scenario similar to those in the training set but with distinct parameter values, as well as a unique single harmonic (SH) signal type, which the model did not encounter during training. This ensures that the model can handle both familiar and novel situations effectively.

Table 8.2 summarizes the number of samples in the training, test, and validation datasets, providing a clear overview of the dataset structure used in this study. Each sample consists of three consecutive timesteps as input and one timestep as output.

Signals	κ_θ [-]	a_θ [deg]	κ_ξ [-]	a_ξ [m]	Type
Training 1	0.114	0.80	0.152	-0.098	DS, θ>0, ξ<0
Training 2	0.114	-0.80	0.152	0.098	DS, θ<0, ξ>0
Training 3	0.148	1.00	0.181	-0.123	DS, θ>0, ξ<0
Training 4	0.148	-1.00	0.181	0.123	DS, θ<0, ξ>0
Test 1	0.091	0.70	0.123	0.074	DS, θ>0, ξ>0
Test 2	0.104	0.90	0.089	0.061	DS, θ>0, ξ<0
Test 3	0.104	-0.90	0.089	-0.061	DS, θ<0, ξ<0
Test 4	0.092	0.75	0.081	-0.059	US, θ>0, ξ<0
Test 5	0.147	-1.00	0.000	0.000	DS, θ<0
Test 6	0.000	0.00	0.072	0.049	DS, ξ>0
Validation 1	0.147	-1.00	0.072	0.049	DS, θ<0, ξ>0
Validation 2	0.106	3.00	0.089	-0.246	SH, θ>0, ξ<0

TABLE 8.1: Parameters and types of training, test and validation signals. DS: damped Schroeder-phased harmonic, US: undamped Schroeder-phased harmonic, SH: single harmonic.

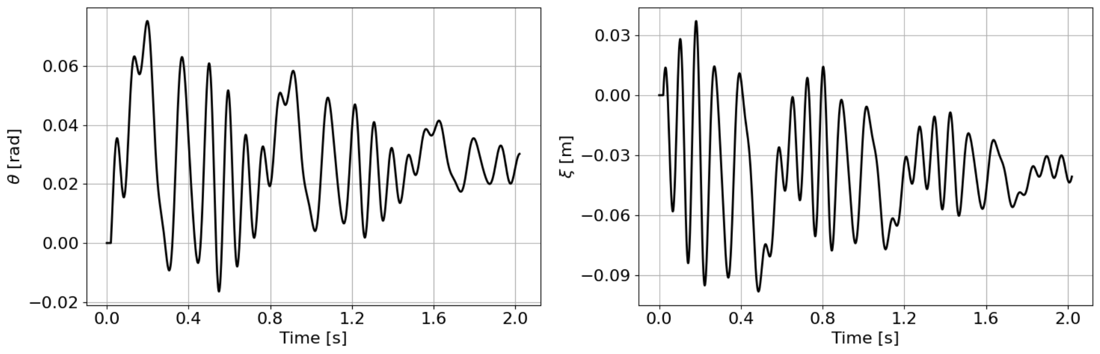


FIGURE 8.3: Training signal 1: DS with $\kappa_\theta = 0.114$, $a_\theta = 0.80$ [deg], $\kappa_\xi = 0.152$, and $a_\xi = -0.098$ [m].

Dataset	Number of Samples
Training	2,640 (4 × 660)
Test	3,960 (6 × 660)
Validation	1,320 (2 × 660)

TABLE 8.2: Number of samples for the training, test and validation datasets.

8.4 Results

In this section, we present the results of the reconstructed validation signals for two different types of architectures: feedforward model and ARMA model. By comparing the performance of these two architectures, we aim to illustrate the influence of incorporating predicted C_P into the model input on the overall prediction accuracy and robustness. The impact of the temporal layer on the accuracy of each architecture is also studied.

8.4.1 Feedforward Model

The feedforward model utilizes spatial and temporal data from previous timesteps without relying on its own past predictions, thus preventing the accumulation of errors over time. The model performance across different temporal layers is evaluated using both DS and SH validation signals.

For the DS signal (Figures 8.4 and 8.5), the predictions of the aerodynamic coefficients C_L and C_M align well with the CFD reference data in all temporal layers, as shown in Figure 8.4. The STGCN layer demonstrates the lowest average error for C_L, while LSTM performs slightly better for C_M. A region of maximum error, indicated by the red circle, is consistent across all models and represents the point where the predictions for C_L and C_M show the greatest deviation from the reference data. This error appears to be related to the models difficulty in capturing sharp transitions or nonlinearities in the aerodynamic flow, particularly near shock waves. Figure 8.5 shows the C_P distribution at this maximum error point and highlights the models overall ability to predict the pressure distribution across the wing. While most temporal layers perform reasonably well, some discrepancies near the leading edge and areas affected by shock waves and flow separation are noticeable. These areas, characterized by strong flow gradients and nonlinear aerodynamic behaviour, are challenging for all models, although STGCN and LSTM exhibit slightly better accuracy compared to GRU and Attention mechanisms. This behaviour reflects the inherent sensitivity of shock-boundary layer interaction to rapid variations in incidence and flow conditions, where even small discrepancies in phase or amplitude propagate into noticeable load prediction errors.

For the SH signal (Figures 8.6 and 8.7), which involves higher-frequency oscillations, the models encounter increased errors, particularly in predicting peak values for C_L and C_M. LSTM and STGCN continue to provide the most accurate results, although both exhibit some phase and amplitude errors due to the rapid oscillations. The red-circled point, indicating the region of maximum error for C_L and C_M, again highlights the challenges of accurately capturing high-frequency aerodynamic loads. Figure 8.7 provides a detailed view of the C_P distribution at this maximum error point, where the models struggle more to match the rapid fluctuations in C_P. Again, regions near the shock wave, which undergo significant temporal variation, show greater discrepancies. While the general C_P distribution is captured, the accuracy decreases in areas where shock-induced flow separation occurs, with LSTM and STGCN again showing marginally better performance in these challenging regions. These results underline the increased difficulty of resolving rapid unsteady shock motion and associated separation dynamics, which demand models with both longer temporal memory and finer spatial resolution to capture the underlying physics.

To quantitatively evaluate the models, we calculated three error metrics, Mean Absolute Percentage Error (MAPE), coefficient of determination (R2), and Root Mean Square

Error (RMSE), for C_P predictions across both validation signals, as shown in Table 8.3. The MAPE was derived by averaging the absolute errors across each timestep, weighted by the corresponding cell area, and normalized accordingly. It reflects the average percentage error across predictions, showing that the LSTM model consistently achieves the lowest values, while the STGCN model follows closely. GRU and Attention mechanisms, on the other hand, display significantly higher MAPE, particularly for the high-frequency oscillations of the SH signal, indicating their difficulty in capturing rapid temporal variations and nonlinearities. In terms of R2, which measures how well the model explains variance in the data, LSTM again performs the best, capturing the highest degree of variability, followed closely by STGCN, which also demonstrates strong performance in the DS signal but slightly lower accuracy for the SH signal. GRU and Attention layers show lower R2, further confirming their difficulty to fully capture the complexities in regions involving shock waves and high dynamic variability. Regarding RMSE, which emphasizes larger errors due to its quadratic nature, the STGCN model performs better with the lowest values, indicating a good robustness against significant deviations. LSTM shows similar performance but struggles slightly more with large deviations in dynamic conditions like the SH signal. Again, GRU and Attention mechanisms exhibit the highest RMSE. These results suggest that all models are well-suited for capturing both spatial and temporal dependencies with good overall performance, but GRU and Attention layers struggle more to capture the rapid temporal variations

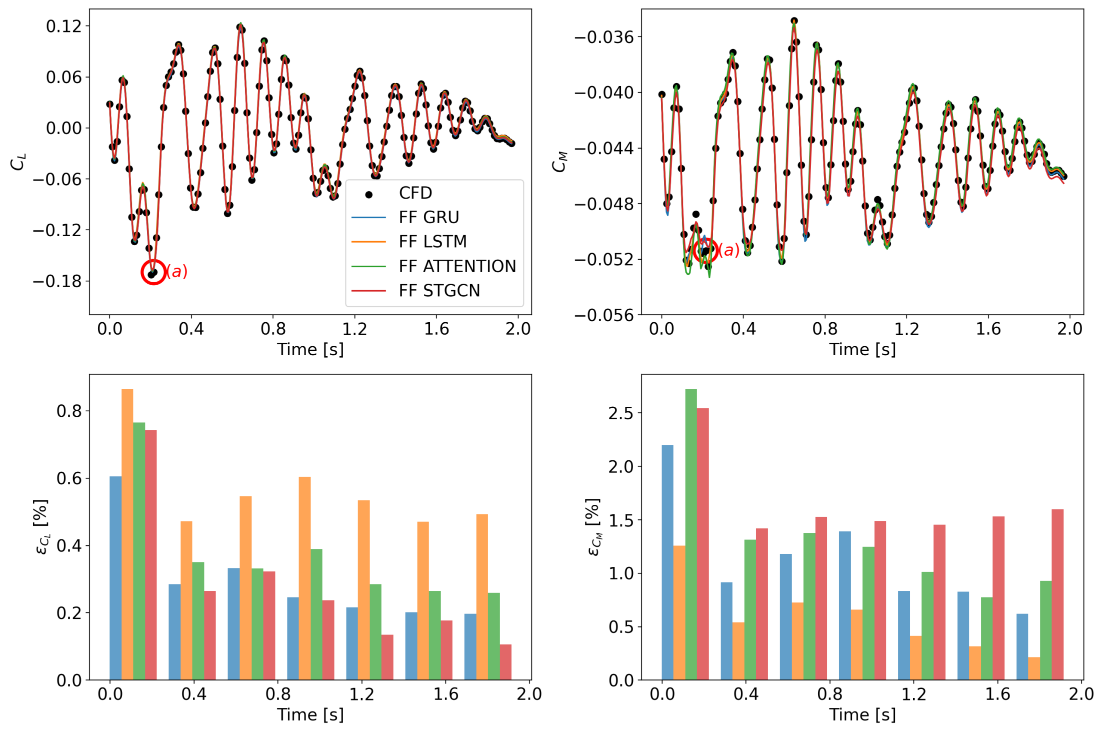


FIGURE 8.4: Validation signal 1 - DS type: Effect of temporal layer selection on C_L and C_M predictions in the feedforward model. The red circle marks the point of maximum error, used for plotting the C_P distribution. FF: FeedForward.

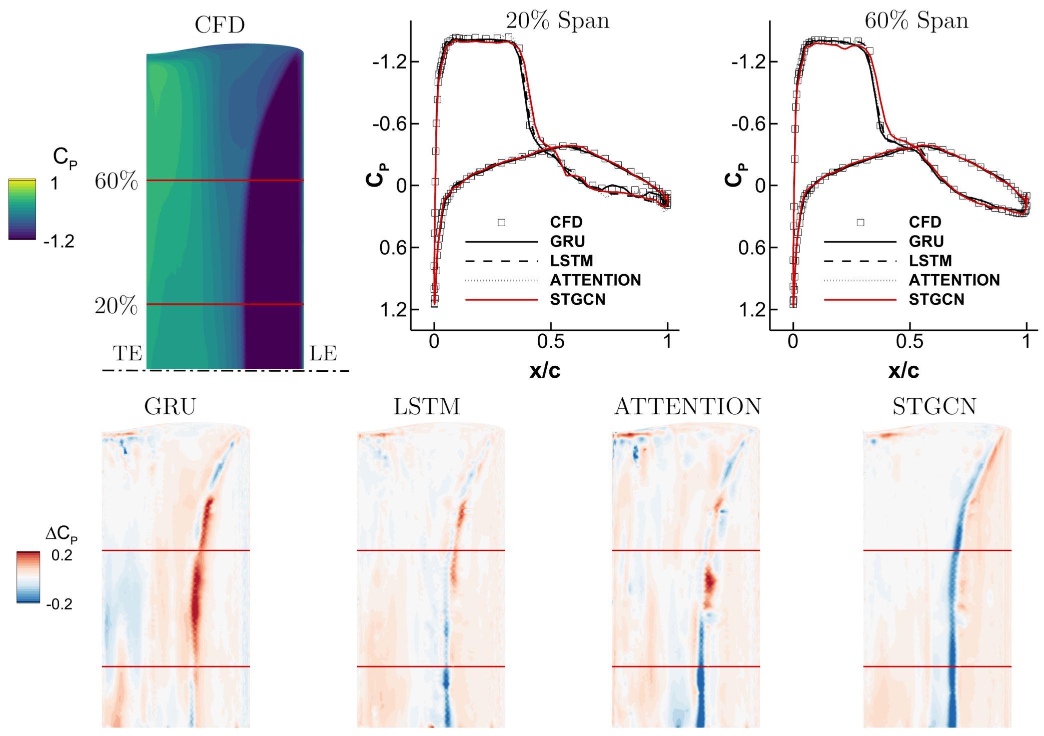


FIGURE 8.5: Validation signal 1 - DS type: Effect of temporal layer selection on C_P prediction at the maximum error point in the feedforward model. The lower surface of the wing is shown. LE: Leading Edge. TE: Trailing Edge. Dash-dot line indicates the symmetry plane.

Temporal Layer	**MAPE**		**R2**		**RMSE**	
	DS	SH	DS	SH	DS	SH
GRU	1.2382	1.7851	0.9851	0.9830	0.0217	0.0229
LSTM	**0.7471**	**0.9695**	**0.9937**	**0.9909**	0.0194	0.0215
Attention	1.0470	1.5452	0.9887	0.9815	0.0238	0.0291
STGCN	0.8524	0.9975	0.9918	0.9897	**0.0163**	**0.0181**

TABLE 8.3: Comparison of MAPE, R2, and RMSE for C_P predictions in the feedforward model with different temporal layers for DS and SH validation signals.

and nonlinearities in the C_P distribution.

In conclusion, while all temporal layers provide reasonable predictions for unsteady aerodynamic phenomena, the LSTM model delivers the most robust performance across both validation signals. STGCN also performs well, particularly for C_L predictions, while GRU and Attention mechanisms exhibit larger errors, especially in more complex dynamics. The evaluation of C_P distribution highlights the importance of accurately capturing nonlinear flow features, with regions near shock waves and flow separations proving to be the most challenging for all models.

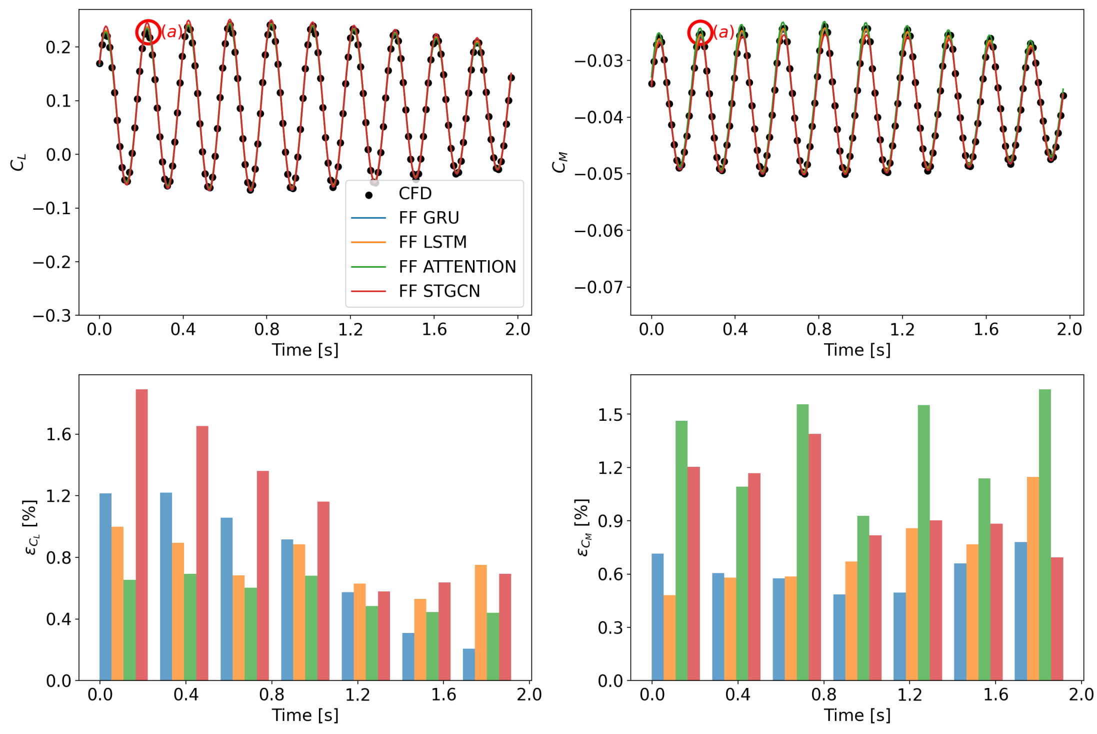


FIGURE 8.6: Validation signal 2 - SH type: Effect of temporal layer selection on C_L and C_M predictions in the feedforward model. The red circle marks the point of maximum error, used for plotting the C_P distribution. FF: FeedForward

8.4.2 ARMA Model

The ARMA model, which integrates past predictions into its input, was evaluated using various temporal layers to assess its performance in predicting C_L, C_M, and C_P across DS and SH validation signals. As shown in Figure 8.8, one of the biggest limitations of the ARMA model consists of accumulation of errors over time, particularly when using the GRU and Attention mechanisms. Again, LSTM and STGCN demonstrate better predictive performance, although both exhibit some deviations in complex flow regions. The red-circled points were selected to visualize the evolution of the error over time. These points were spaced at regular intervals along the signal to provide insight into how prediction accuracy changes throughout the sequence. This allows us to observe how the model handles different stages of prediction and highlights its difficulty in accurately capturing sharp transitions and nonlinearities in the aerodynamic flow, particularly around shocks. This issue is further emphasized in Figure 8.9, where the C_P at these selected points shows that the ARMA model has more difficulty maintaining accuracy near the leading edge and in shock-affected regions. Despite these challenges, LSTM and STGCN continue to provide the most reliable predictions despite the inherent error accumulation. This behaviour reflects a fundamental limitation of autoregressive predictors: when the underlying flowfield exhibits rapid nonlinear transitions, such as shock motion or local boundary layer separation, even minor phase errors are recursively propagated, amplifying discrepancies over time.

For the SH signal (Figures 8.10 and 8.11), which involves higher frequency oscillations, the ARMA model exhibits increased errors, particularly for the GRU and Attention layers. The rapid oscillations introduce phase and amplitude discrepancies, with LSTM and STGCN handling these variations better, though both models still show increased errors compared to the DS signal. The C_P distribution in Figure 8.11 reveals that the ARMA model is more susceptible to error propagation in highly dynamic regions near shock waves, where rapid changes in flow introduce greater inaccuracies. Although LSTM and STGCN mitigate these issues to some extent, they still experience some degradation in predictive accuracy due to the model autoregressive nature. The higher-frequency SH excitation exacerbates this limitation, since the autoregressive formulation cannot easily recover from small timing mismatches in the shock oscillation cycle, leading to compounding phase errors in C_L, C_M, and local C_P distributions.

The superior performance of LSTM and STGCN can be attributed to their inherent design, which allows them to handle temporal dependencies more effectively. The LSTM architecture, with its complex gating mechanisms, enables the model to retain and manage both long- and short-term dependencies, preventing information loss over multiple timesteps and helping to mitigate the error accumulation issue inherent in the ARMA model. The STGCN layer combines both spatial and temporal convolutions, making it

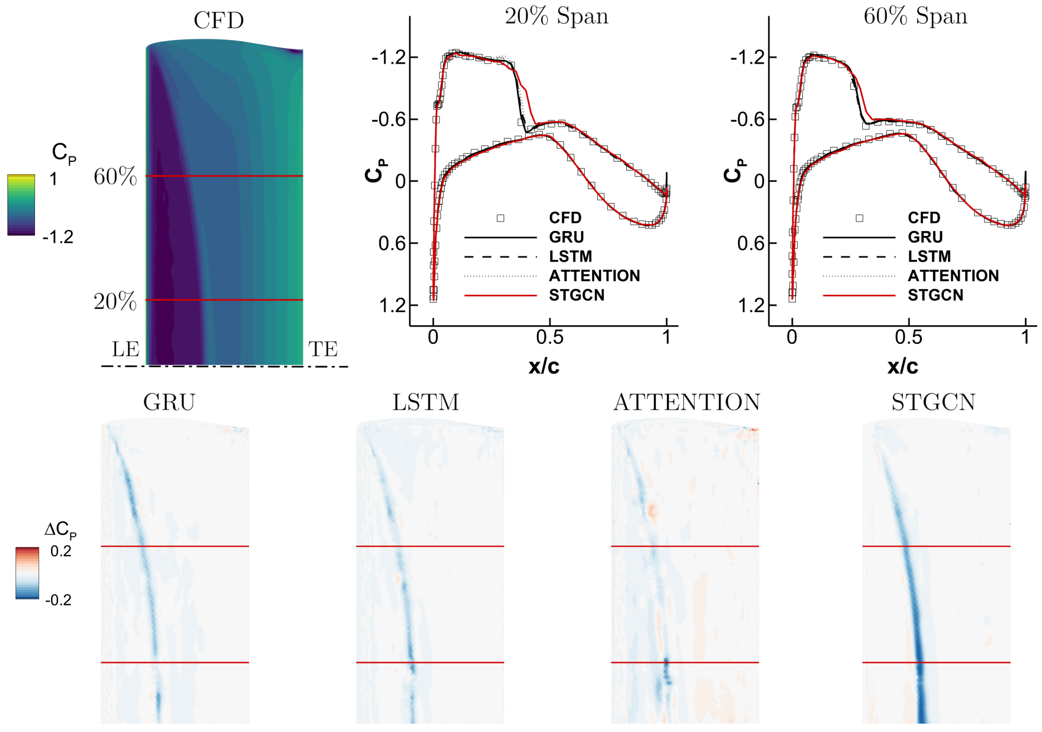


FIGURE 8.7: Validation signal 2 - SH type: Effect of temporal layer selection on C_P prediction at the maximum error point in the feedforward model. The upper surface of the wing is shown. LE: Leading Edge. TE: Trailing Edge. Dash-dot line indicates the symmetry plane.

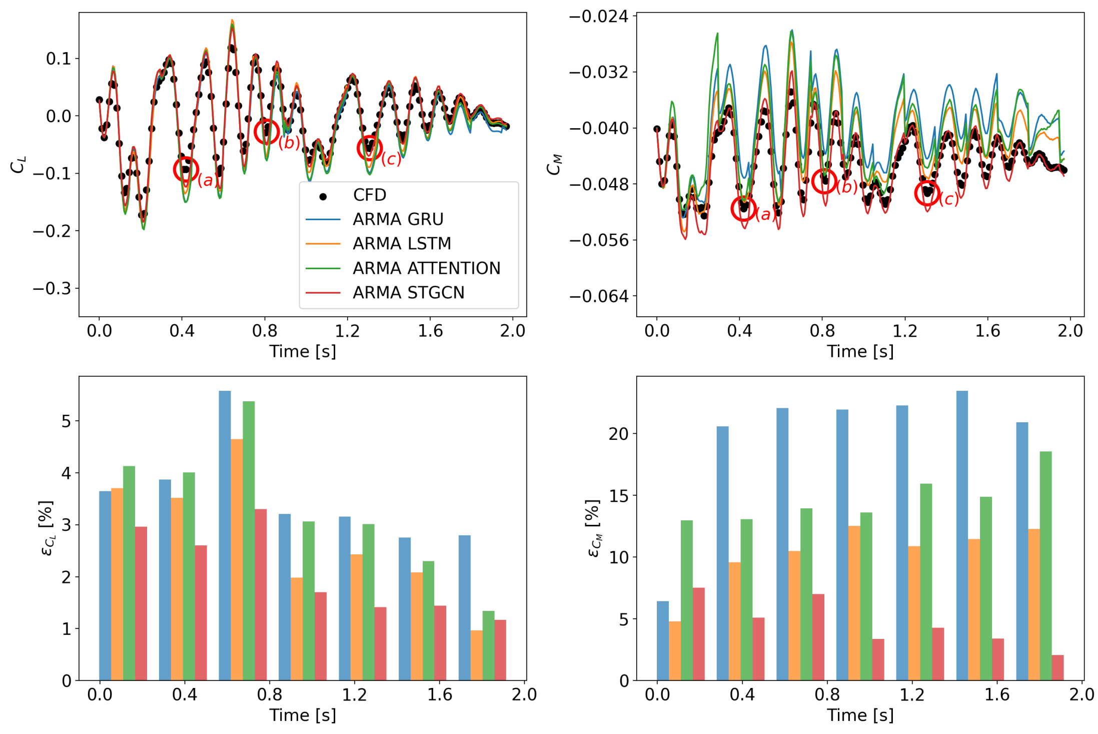
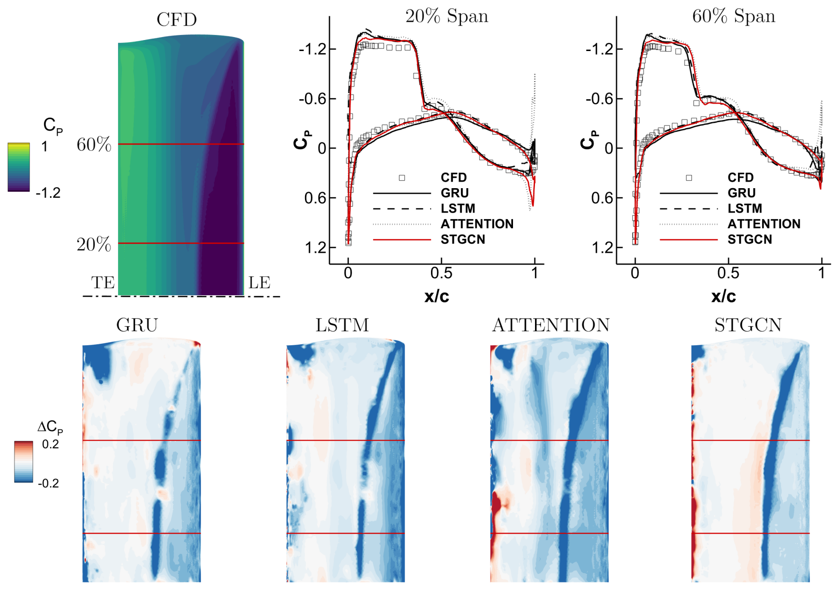


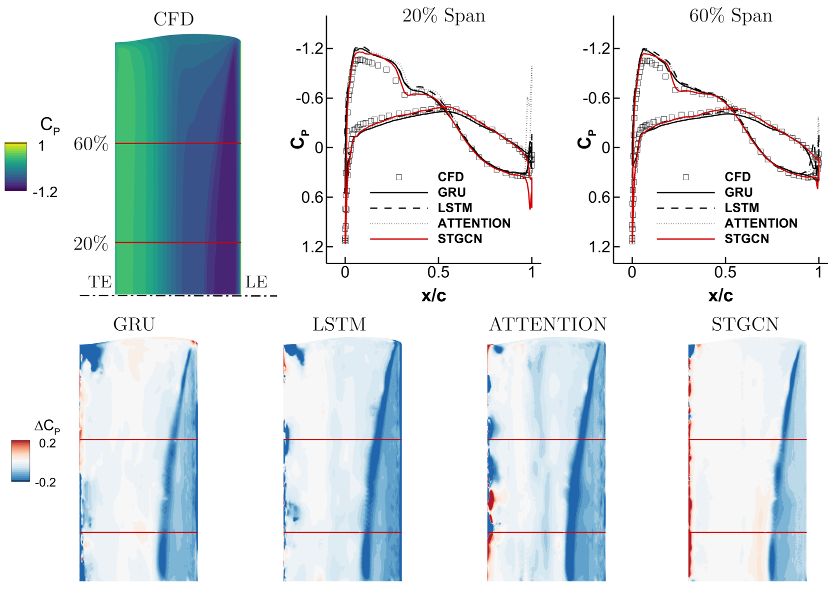
FIGURE 8.8: Validation signal 1 - DS type: Impact of temporal layer selection on C_L and C_M predictions in the ARMA model. Red circles denote the points used for plotting the C_P distribution.

well-suited to handle non-uniform grid structures and effectively capture spatial correlations (such as pressure gradients across the wing) as well as temporal dependencies during rapid changes in aerodynamic conditions, as seen in the SH signal.

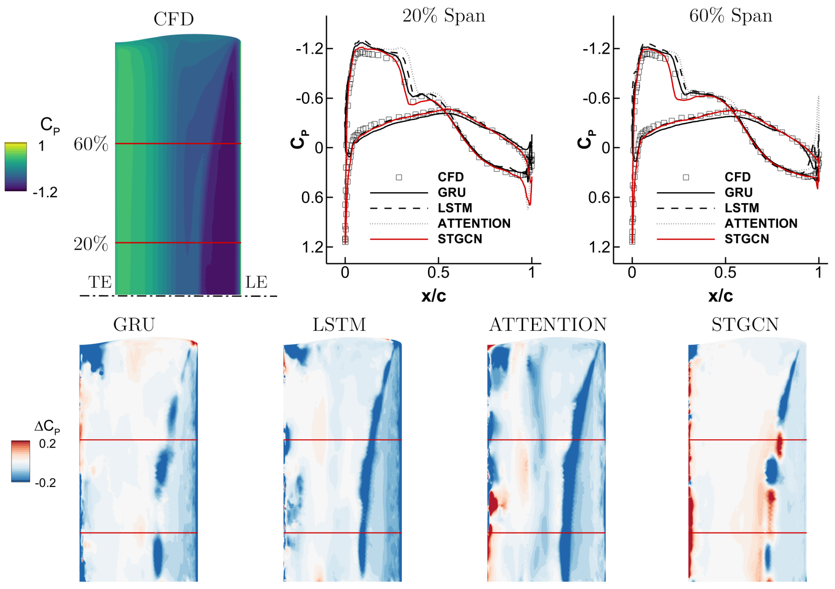


Time Instance (a)

In contrast, the GRU simpler structure limits its capacity to capture long-term dependencies effectively. The Attention mechanism, while effective for capturing important temporal relationships in longer sequences, is not as well suited to the short input sequences used in this model, where the benefits of dynamically weighting timesteps are reduced, limiting the Attention layer effectiveness.



Time Instance (b)



Time Instance (c)

FIGURE 8.9: Validation signal 1 - DS type: Impact of temporal layer selection on C_P prediction at the maximum error point in the ARMA model. The lower surface of the wing is shown. LE: Leading Edge. TE: Trailing Edge. Dash-dot line indicates the symmetry plane.

Overall, these results emphasise that accurate treatment of memory effects is essential when modelling transonic responses, where the loads depend not only on the instantaneous state but also on the short-term history of shock and separation dynamics.

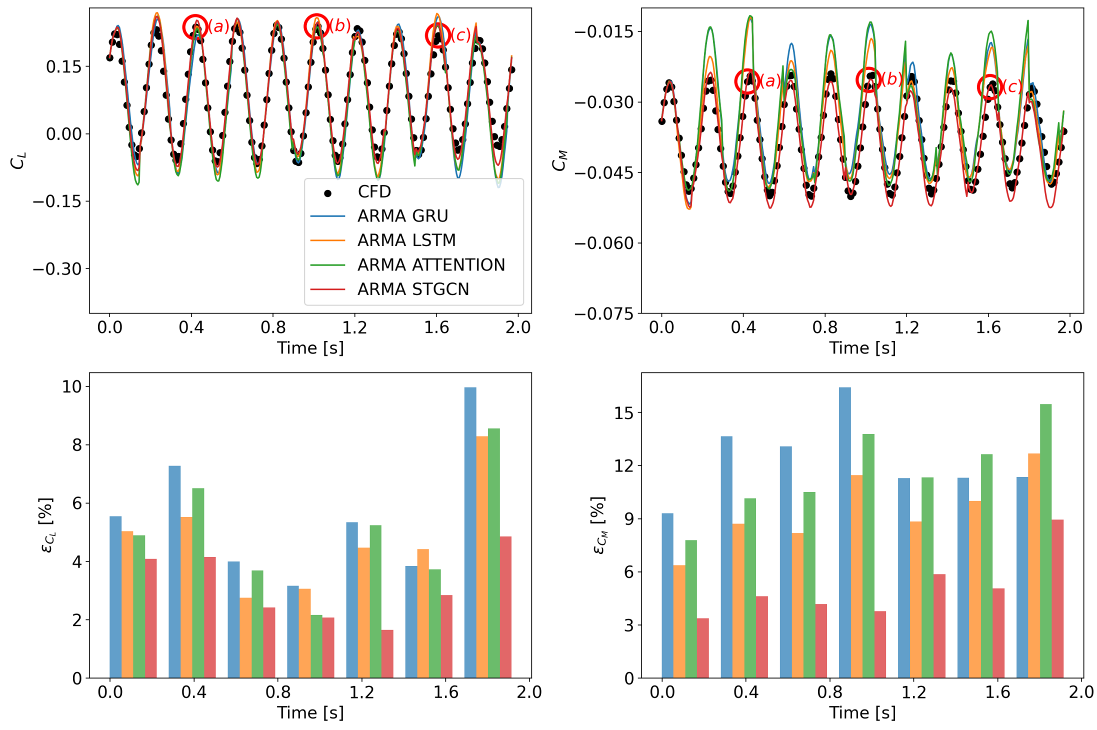
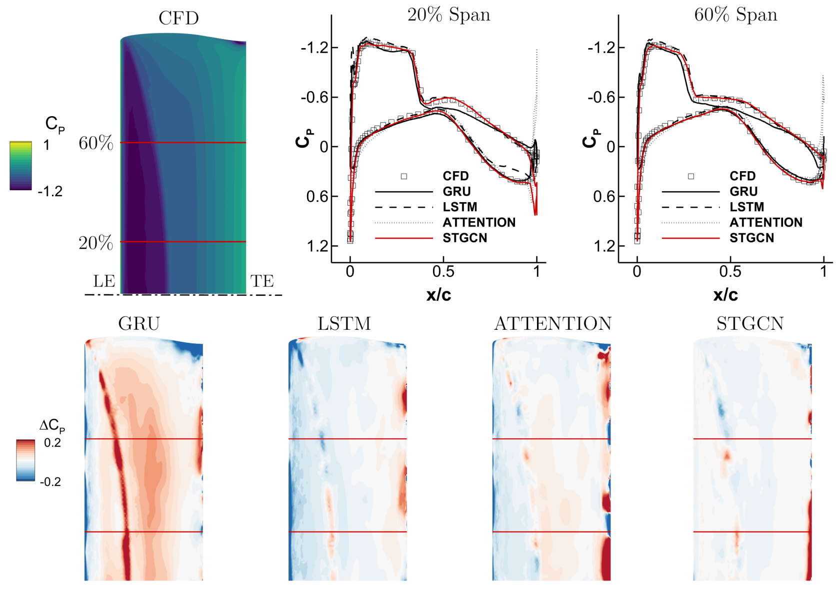
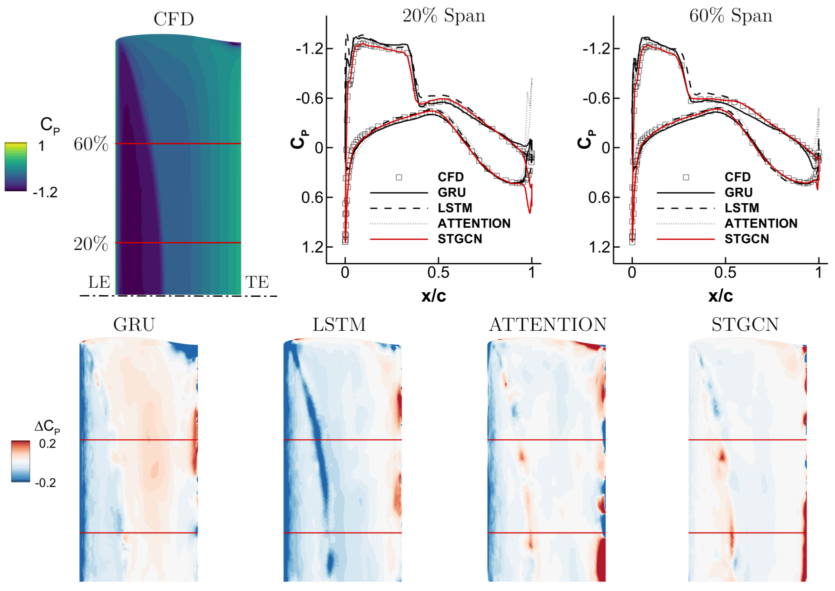


FIGURE 8.10: Validation signal 2 - SH type: Impact of temporal layer selection on C_L and C_M predictions in the ARMA model. Red circles denote the points used for plotting the C_P distribution.

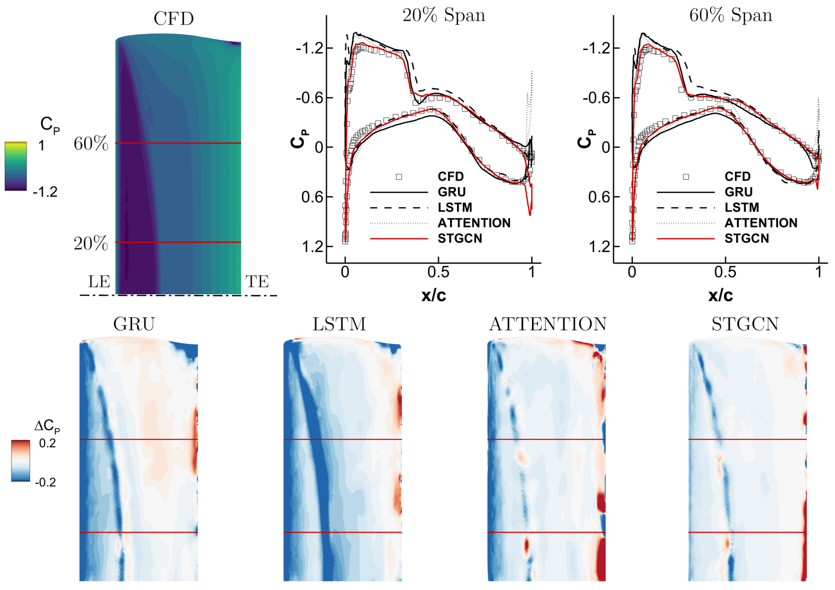


Time Instance (a)

Table 8.4 quantifies the performance of the different temporal layers in terms of MAPE, R2, and RMSE for C_P predictions in the ARMA model. The STGCN model consistently



Time Instance (b)



Time Instance (c)

FIGURE 8.11: Validation signal 2 - SH type: Impact of temporal layer selection on C_P prediction at the maximum error point in the ARMA model. The upper surface of the wing is shown. LE: Leading Edge. TE: Trailing Edge. Dash-dot line indicates the symmetry plane.

achieves the lowest `MAPE` and `RMSE` values across both the DS and SH signals, with LSTM performing nearly as well. This suggests that both models are adept at handling temporal dependencies even in autoregressive contexts, although the STGCN slightly outperforms LSTM, particularly in the dynamic SH signal. GRU and Attention layers, on the other hand, show higher `MAPE` and `RMSE`, along with lower `R2`, indicating that they struggle to mitigate the error accumulation inherent in the ARMA model, particularly in high-frequency oscillations. The lower R2 values for GRU and Attention mechanisms highlight their reduced ability to capture the full variability in the data with rapid aerodynamic changes. Overall, the results suggest that while the ARMA model can capture general trends, its susceptibility to error propagation makes it essential to use robust temporal layers, such as LSTM and STGCN, to minimize predictive inaccuracies in highly dynamic and nonlinear aerodynamic conditions.

Temporal Layer	**MAPE**		**R2**		**RMSE**	
	DS	SH	DS	SH	DS	SH
GRU	8.5577	6.7837	0.7560	0.7993	0.1389	0.1439
LSTM	7.7511	6.4299	0.8426	0.8506	0.1002	0.0864
Attention	7.7985	5.8077	0.8167	0.7796	0.1233	0.1048
STGCN	**6.9381**	**5.7971**	**0.8571**	**0.8648**	**0.0938**	**0.0844**

TABLE 8.4: Comparison of MAPE, R2, and RMSE for C_P predictions in the ARMA model with different temporal layers for DS and SH validation signals.

8.4.3 Comparison between Feedforward and ARMA

The comparison between ARMA and feedforward models, both using the STGCN temporal layer, reveals notable differences in performance, especially regarding error accumulation and prediction stability. The STGCN temporal layer was selected for this comparison because it consistently yielded the most accurate results across both validation signals, as demonstrated in previous sections. In this case, ARMA model uses ground-truth C_P values for the first half of the signal, after which it switches to using its own predictions for subsequent timesteps.

As shown in Figure 8.12, the feedforward model provides more accurate predictions for C_L and C_M on the DS signal, avoiding the accumulation of errors observed in the ARMA model. The red circled points were chosen to be in the middle of the first and second halves of the signal, providing information on the behaviour of the model during the transition from using ground truth to self-predicted values. This choice allows for a clearer comparison of model performance during both phases, highlighting the ARMA model difficulty in limiting error accumulation, while the feedforward model maintains closer alignment with the reference data. This trend is further confirmed in Figure 8.13, where the evolution of the MAPE in C_P reveals that the feedforward model consistently maintains lower error levels over time compared to the ARMA model, which, as expected, shows a clear pattern of accumulation of errors.

Interestingly, the ARMA model performs relatively well during the first half of the signals, when ground-truth C_P values are used as input. In this phase, the ARMA model can even outperform the feedforward model, as seen in the initial part of Figures 8.12 and 8.13. However, once the model begins using its own predicted C_P values for subsequent timesteps, error accumulation begins, resulting in larger deviations from the reference data. This behaviour is clearly seen in Figure 8.14, where the ARMA model shows growing discrepancies in C_P predictions as the autoregressive process progresses. In contrast, the feedforward model avoids this problem by not relying on its past predictions, allowing it to maintain better accuracy in regions near the leading and trailing edges, where flow complexity is higher.

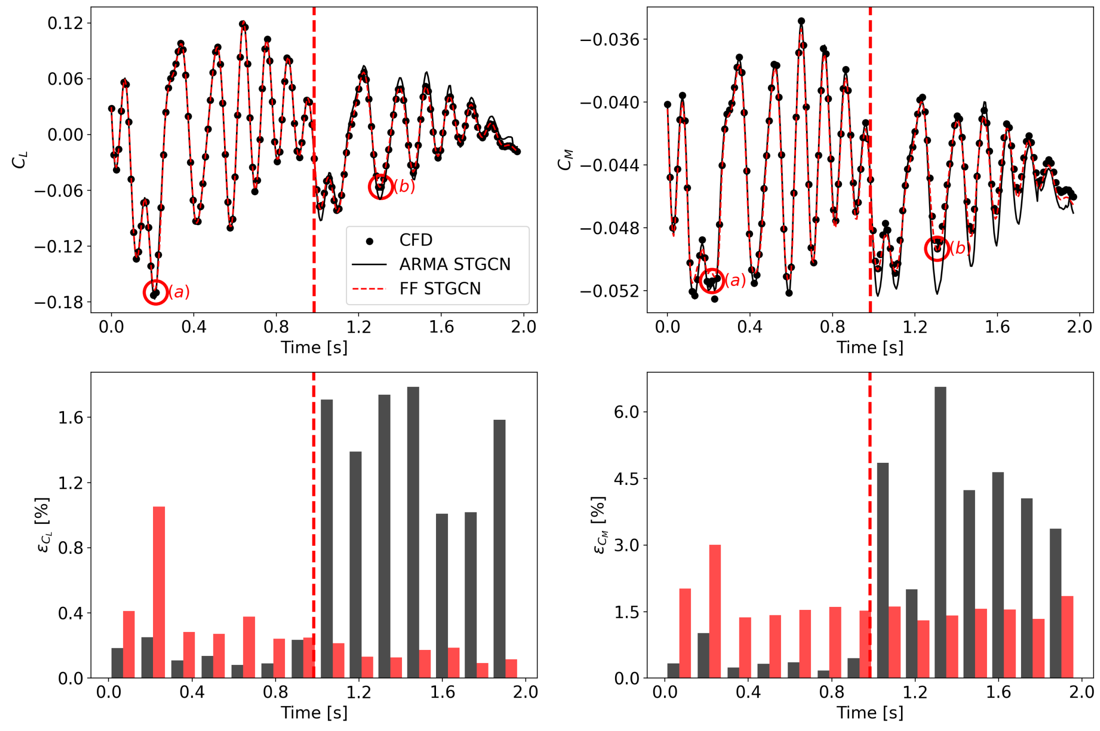


FIGURE 8.12: Validation signal 1 - DS type: ARMA vs. Feedforward model using STGCN temporal layer on C_L and C_M predictions. Red circles denote the points used for plotting the C_P distribution. Vertical dashed line marks the time instance where the ARMA model transitions to autoregressive C_P predictions.

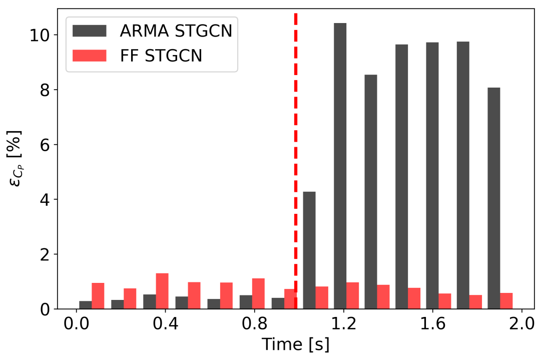
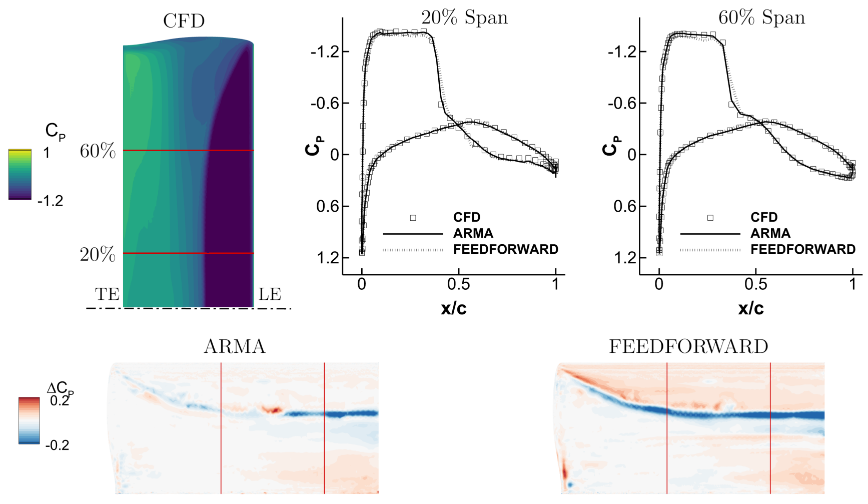
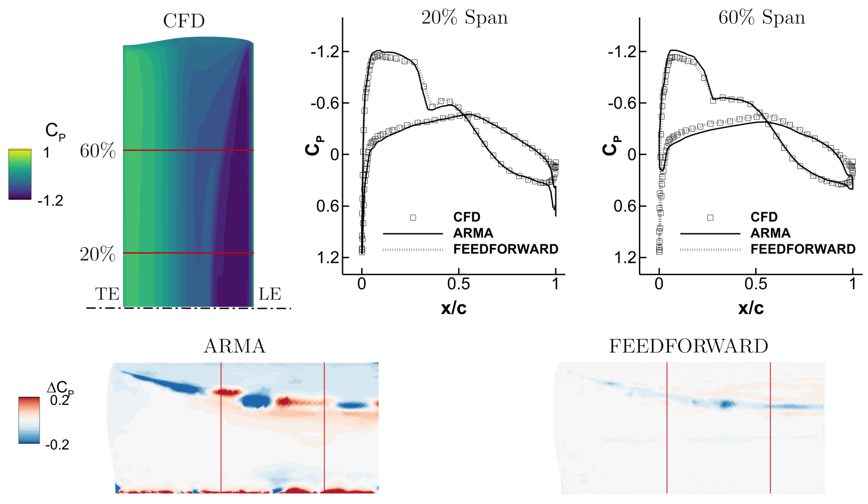


FIGURE 8.13: Validation signal 1 - DS type: Evolution of MAPE of C_P with ARMA and Feedforward model. Vertical dashed line marks the time instance where the ARMA model transitions to autoregressive C_P predictions.

These trends are even more evident with the SH signal, which features rapid oscillations. Figure 8.15 shows that the feedforward model handles these high-frequency



Time Instance (a)



Time Instance (b)

FIGURE 8.14: Validation signal 1 - DS type: ARMA vs. Feedforward model using STGCN temporal layer on C_P prediction. The lower surface of the wing is shown. LE: Leading Edge. TE: Trailing Edge. Dash-dot line indicates the symmetry plane.

aerodynamic variations more effectively, with significantly fewer phase and amplitude errors than the ARMA model. The ARMA model, due to its time-marching scheme, exhibits a greater sensitivity to reduced frequency. At higher frequencies, errors accumulate faster as small inaccuracies from previous steps compound. Once the ARMA model switches from using ground-truth inputs to its own predictions, it fails to keep pace with the rapid changes in aerodynamic forces, resulting in larger phase lags and more significant deviations from the reference data. In contrast, as the feedforward model relies only on the wing spatial coordinates and prescribed motions at previous timesteps (without feeding back its own predictions), it results in a more stable error profile and in a more reliable and accurate predictions of unsteady phenomena.

As shown in Figure 8.16, the feedforward model consistently outperforms the ARMA model in terms of `MAPE` on C_P for the SH signal. The ARMA model error accumulation is particularly pronounced in more dynamic conditions, such as rapid oscillations, which results in significantly higher `MAPE`. Figure 8.17 further supports this observation, showing that the feedforward model is better at predicting the C_P distribution, where the ARMA model struggles due to the compounding of prediction errors. This behaviour can be traced back to the recursive nature of the ARMA formulation: although it employs a larger input set, including past and possibly self-predicted values, it operates in a multi-step free-run simulation where small one-step errors are repeatedly fed back, leading to distribution shift and error amplification. Such limitations are well documented in the multi-step forecasting and system identification literature, where recursive strategies typically suffer from error accumulation over the prediction horizon, while direct or multi-output strategies alleviate this effect [27, 284, 334, 335]. Nevertheless, when the ARMA model is provided with ground-truth C_P values, it can achieve very accurate predictions, in some cases outperforming the feedforward model. This highlights its potential in scenarios where error accumulation is less critical, such as systems with lower reduced frequency. Moreover, insufficient convergence of the CFD solution may exacerbate error growth in regions with complex flow patterns, suggesting that enhancing solution stability could help mitigate these issues.

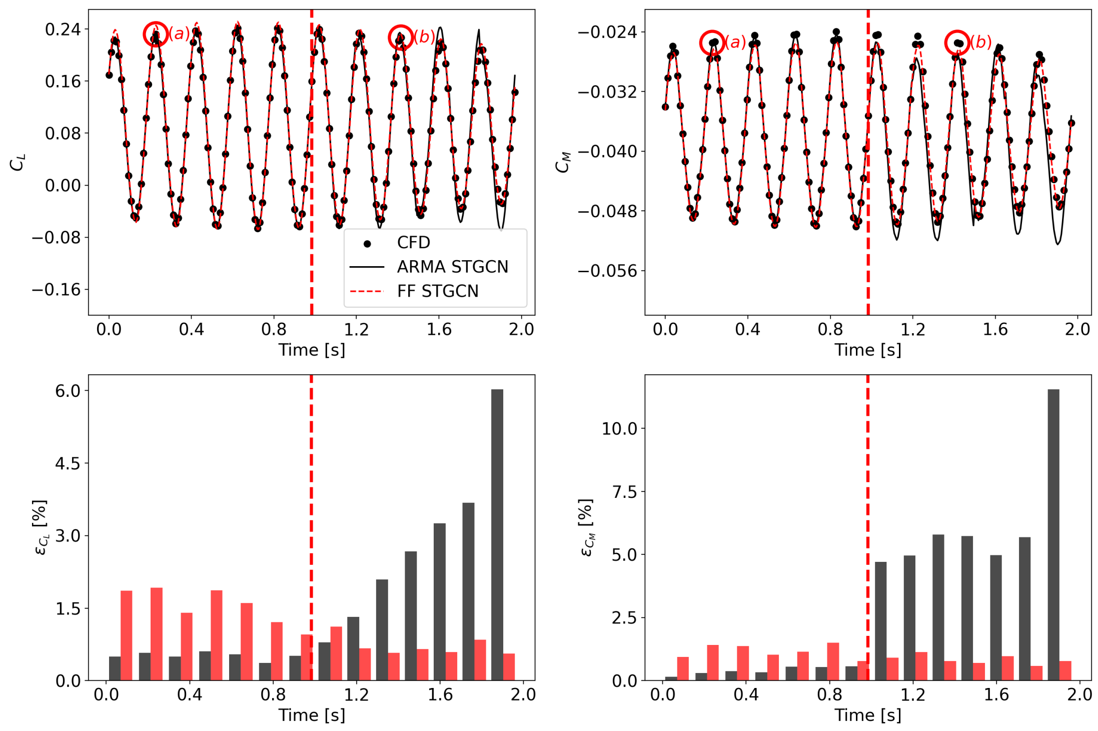


FIGURE 8.15: Validation signal 2 - SH type: ARMA vs. Feedforward model using STGCN temporal layer on C_L and C_M predictions. Red circles denote the points used for plotting the C_P distribution. Vertical dashed line marks the time instance where the ARMA model transitions to autoregressive C_P predictions.

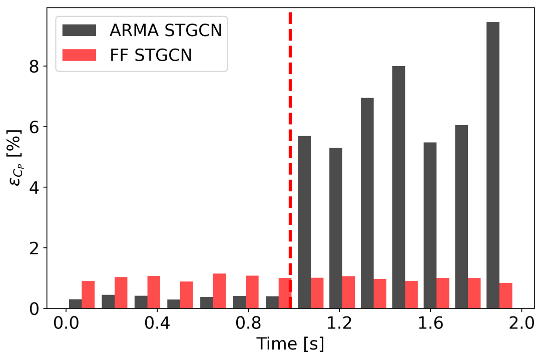
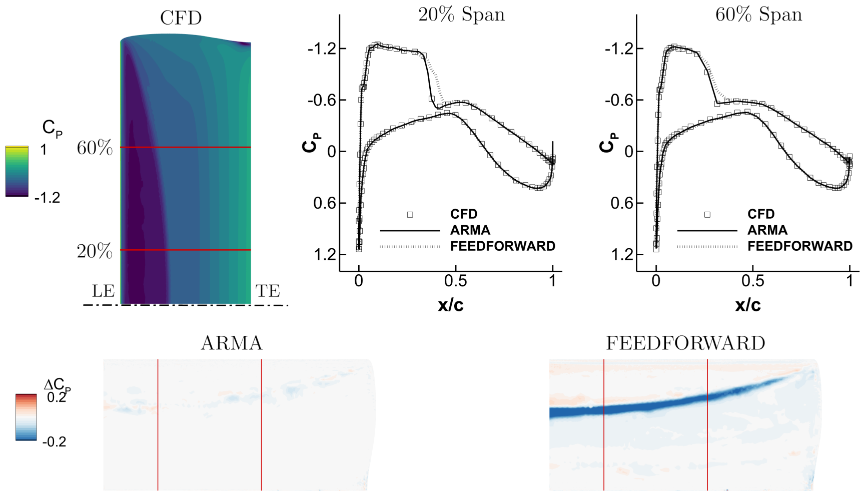
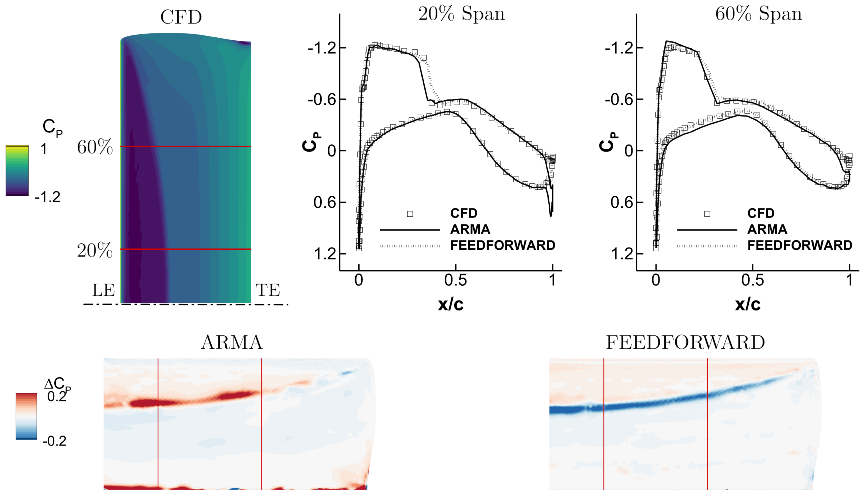


FIGURE 8.16: Validation signal 2 - SH type: Evolution of MAPE of C_P with ARMA and Feedforward model. Vertical dashed line marks the time instance where the ARMA model transitions to autoregressive C_P predictions.



Time Instance (a)



Time Instance (b)

FIGURE 8.17: Validation signal 2 - SH type: ARMA vs. Feedforward model using STGCN temporal layer on C_P prediction. The upper surface of the wing is shown. LE: Leading Edge. TE: Trailing Edge. Dash-dot line indicates the symmetry plane.

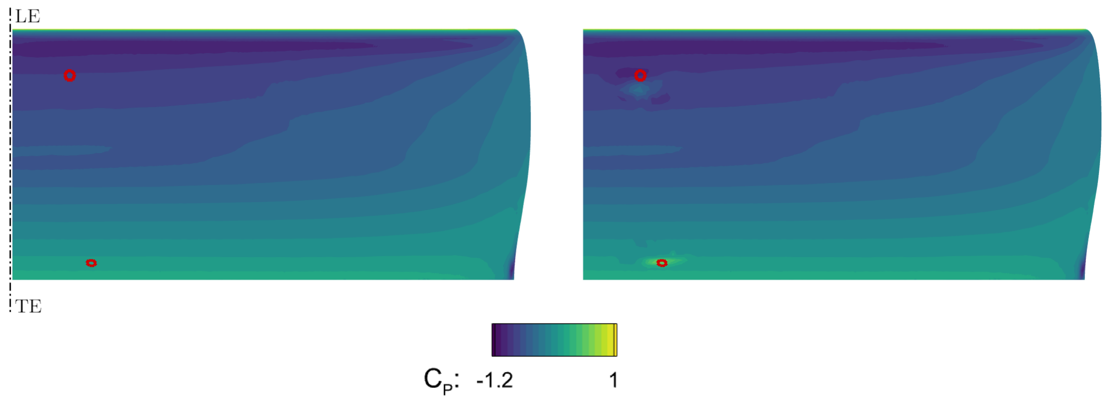


FIGURE 8.18: Instantaneous C_P perturbation study demonstrating model sensitivity to elliptical and hyperbolic flow behaviour.

8.4.4 Sensitivity Analysis to Elliptical and Hyperbolic Regions

The framework has the ability to independently detect the elliptic or hyperbolic nature of the flow, as demonstrated by a sensitivity analysis in which localised perturbations were applied separately in subsonic and supersonic regions (see Figure 8.18). Specifically, we introduced a 100 % instantaneous increase in C_P at a single surface node. Two cases are examined: (a) the perturbed node lies inside a subsonic area downstream of the shock, and (b) the node is inside the supersonic region immediately upstream of the shock. The perturbation is injected into the ARMA variant, and its spatial propagation is tracked for one time-step through the decoder. Our analysis showed distinct differences in the response to these perturbations: In the subsonic (elliptical) region, the perturbation affected the pressure distribution in all directions around the perturbed node, reflecting the expected elliptical behaviour characterized by long-range spatial correlation. In the supersonic (hyperbolic) region, the perturbation only affected nodes downstream of the perturbed node, consistent with hyperbolic behaviour where information propagates primarily in the downstream direction. Closer inspection reveals that the affected region is not only biased in the downstream direction, but also laterally constrained by the local Mach cone. In weakly supersonic pockets where the local Mach number lies between $M_{\text{loc}} = 1.0$ and 1.2, the corresponding Mach angle spans approximately 60 [deg] to 90 [deg]. As illustrated in Figure 8.18, the high-ΔC_P response falls almost entirely inside this 2μ-wide cone-shaped sector. These results confirm that the convolution kernels in our graph-based architecture adaptively encode flow-specific properties without explicit a priori definitions, successfully capturing the essential physical behaviour of transonic flows.

8.4.5 Extrapolation to Off-Design Flow Conditions

The Feedforward AeroNet was trained exclusively on URANS data obtained at the reference condition $M = 0.74$, but in practical applications the wing will often operate at significantly different freestream parameters. To assess how far the network can be pushed outside its training envelope, we confronted it with wind tunnel measurements of pitch-only motion [269]. The experimental database provides mean pressure coefficient $(C_P)_{Mean}$ distributions extracted at 60 % span for two off-design Mach numbers, $M = 0.70$ and $M = 0.85$, respectively, tested at mean incidences (AoA_∞) of 0 [deg] and 5 [deg], while keeping the pitch amplitude ($a_\theta = 1.03$ [deg]) and reduced frequency ($k_\theta = 0.439$) identical for the two maneuvers. Note that the current implementation has no explicit input of flow parameters-the current network receives only the instantaneous wing kinematics $\{\theta, \dot{\theta}, \ddot{\theta}\}$ as input. Variations in M_∞ or Re therefore affect the model only indirectly through the rescaled timestep, which enters through the first and second order derivatives of the motion. In practice, this means that any prediction at an off-design Mach number is, strictly speaking, an extrapolation task. As with all data-driven models, robust performance can be expected only within the input space spanned by the training data; good generalization requires that the training set include sufficiently rich and diverse examples covering the desired range of operating conditions.

Figure 8.19 contrasts the time-averaged AeroNet predictions (continuous curves) with the corresponding experimental data (symbols). For the subcritical case at $M = 0.70$, the network shows good agreement with the measurements, especially at $AoA_\infty = 0$ deg where the model follows the nearly flat pressure plateau. At higher angles of incidence, the distribution on the lower surface is correct, but the shock, which is relatively weak at this Mach number, appears at about $x/c \approx .0.35$ instead of 0.18, and the associated compression is slightly underpredicted. The trailing edge pressure levels are correctly matched, suggesting that this modest deviation from the design Mach number is well accommodated by the network's learned manifold. At $M = 0.85$, however, the discrepancies become more pronounced. The model tends to shift the shock aft compared to the experimental observations. For $AoA_\infty = 5$ [deg] the discrepancy becomes more significant: the network overestimates the suction on the lower surface downstream of the leading edge, probably due to an incomplete representation of the shock-induced separation, which becomes more relevant at this higher Mach number; a result consistent with the stronger and more upstream shock motion that the network never encountered during training.

The encouraging performance at $M = 0.70$ suggests that moderate deviations from the design point remain within the latent manifold learned by the network, while the larger deviations at $M = 0.85$ expose its limited awareness of transonic shock dynamics when freestream parameters are kept invisible. Making M and Re explicit conditioning variables is the next logical step for improving generalizability.

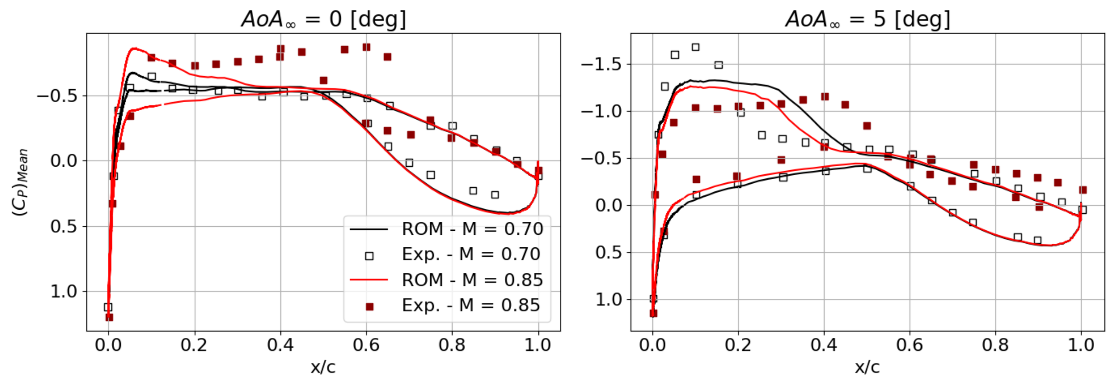


FIGURE 8.19: Time-averaged surface pressure coefficient at 60 % span for pitch-only oscillations with $a_\theta = 1.03$ [deg] and $k_\theta = 0.439$.

8.4.6 Comparison between AeroNet and DMDc

To contextualize the performance of the proposed AeroNet models, we benchmark them against a well-established reduced-order technique, the DMDc, given its similarity to the autoregressive mechanism inherent to our AeroNet ARMA variant. The DMDc implementation follows the procedure detailed by Fonzi et al. [112], which was previously employed successfully on the same benchmark wing under similar flow conditions. For optimal dimensionality reduction and computational efficiency, a total of 25 modes were retained based on a Frobenius norm threshold of 0.1, ensuring a low reconstruction error on the training dataset.

Figures 8.20 and 8.21 compare the predicted C_L and C_M histories from the DMDc and AeroNet models against the CFD reference data for the two validation signals detailed in Table 8.1. For the DS validation signal (Figure 8.20), all models exhibit good agreement with the CFD reference, with the AeroNet FF variant demonstrating notably superior accuracy, indicating its ability to track the broadband energy content of the Schroeder excitation without excessive phase drift. This observation is quantitatively supported by the metrics provided in Table 8.5. For the SH signal, characterized by higher-frequency oscillations (Figure 8.21), both AeroNet variants outperform the DMDc method, reflecting the difficulty of linear ROMs to capture nonlinear shock oscillations and phase lags at higher reduced frequencies. In particular, AeroNet effectively captures dynamic peaks and mitigates the error propagation that noticeably affects the pitching moment predictions, a consequence of its capability to incorporate short-term memory of shock and separation dynamics into the prediction. These qualitative insights are further confirmed by the error metric summarized in Table 8.5.

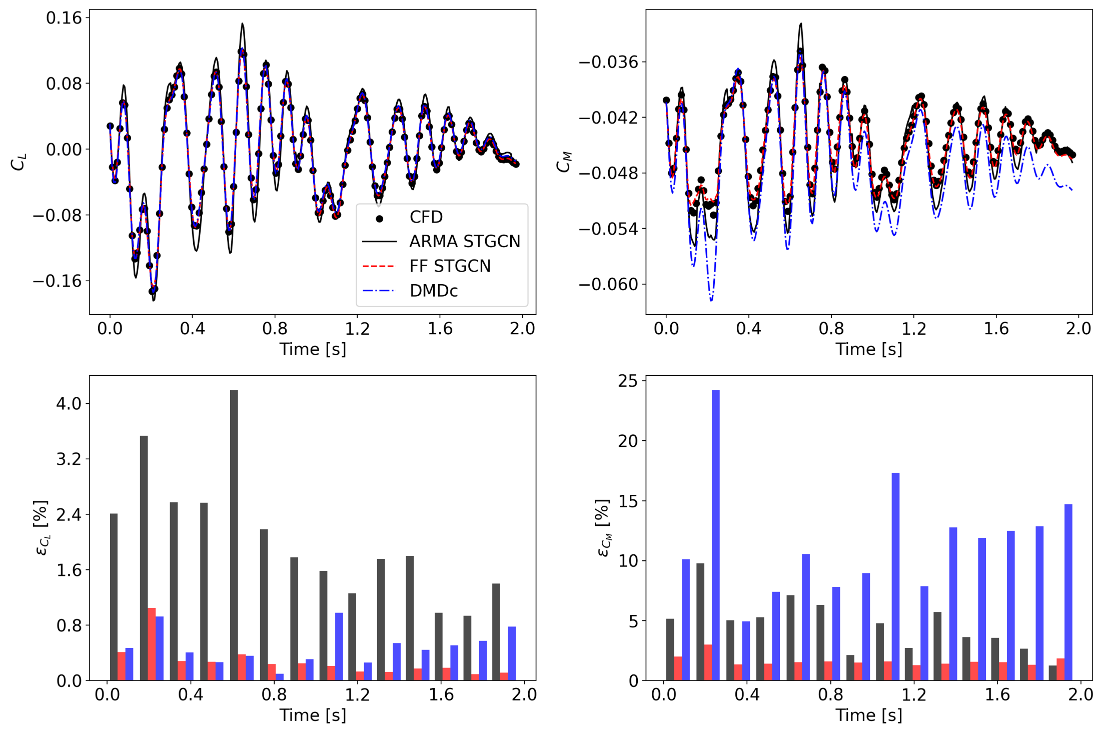


FIGURE 8.20: Validation signal 1 - DS type: DMDc vs. AeroNet models using STGCN temporal layer on C_L and C_M predictions.

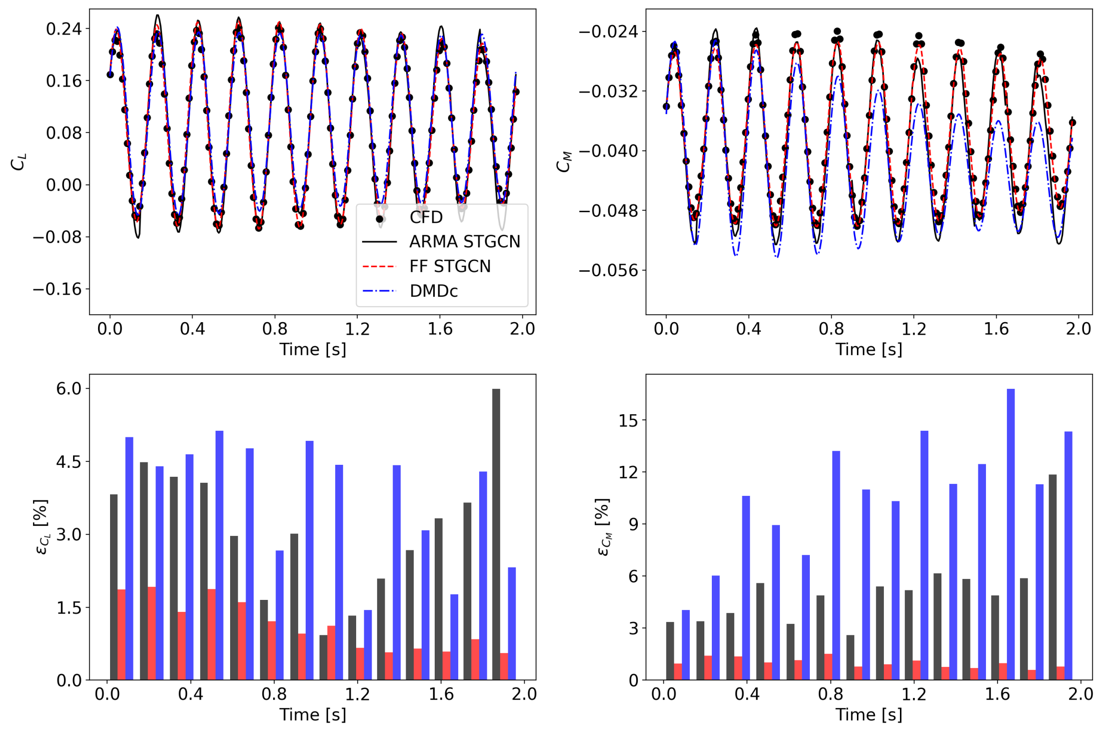


FIGURE 8.21: Validation signal 2 - SH type: DMDc vs. AeroNet models using STGCN temporal layer on C_L and C_M predictions.

To complement the error-propagation analysis, we quantified the stability of the learned latent-space dynamics by estimating the largest Lyapunov exponent, λ_1, for each model

Model	**MAPE**		**R2**		**RMSE**	
	DS	SH	DS	SH	DS	SH
DMDc	4.7961	6.4841	0.9137	0.8574	0.0608	0.1362
Aeronet ARMA	6.9381	5.7971	0.8571	0.8648	0.0938	0.0844
Aeronet FF	0.8524	0.9975	0.9918	0.9897	0.0163	0.0181

TABLE 8.5: Comparison of `MAPE`, `R2`, and `RMSE` for C_P predictions in the DMDc and AeroNet ARMA - FF models with STGCN temporal layer for DS and SH validation signals.

and excitation signal on the predicted C_P. The error is scaled with respect to the cell area. Table 8.6 lists the resulting values together with a qualitative stability assessment: positive λ_1 indicates exponential divergence (chaotic/unstable behaviour), whereas a negative value reflects asymptotic convergence to an attractor.

Signal Type	**Model**	λ_1 [s^{-1}]	**Stability**
DS	DMDc	+0.581	Unstable
DS	AeroNet ARMA	+0.513	Unstable
DS	AeroNet FF	0	Neutrally Stable
SH	DMDc	+0.844	Unstable
SH	AeroNet ARMA	+0.795	Unstable
SH	AeroNet FF	0	Neutrally Stable

TABLE 8.6: Estimated largest Lyapunov exponents λ_1 for AeroNet and DMDc models under DS and SH excitations.

For both signals, the feedforward model has a Lyapunov exponent of zero by definition, as the network never feeds back its own outputs into the next step, local perturbations neither grow nor decay and no systematic error build-up occurs, so the system remains neutrally stable. In contrast, for the DS signal, both ARMA and DMDc variants yield a positive λ_1 , so that perturbations double every 1.3 [s] and 1.1 [s] respectively. This confirms that the autoregressive feedback loop amplifies numerical noise and modelling inaccuracies, a mechanism responsible for the error increase in Figure 8.13. The SH case imposes a higher reduced frequency and a stronger shock oscillation. Here the ARMA model becomes even more unstable, with $\lambda_1 \approx 0.80$ $[s]^{-1}$, implying that trajectory separation doubles roughly every 0.9 [s]. Similar considerations can be drawn for DMDc model, with perturbations double every 0.8 [s]. The feedforward formulation maintains forecast fidelity over extended roll-outs, whereas the autoregressive ARMA is prone to error accumulation unless external correction (e.g., periodic sensor feedback) is available. The DMDc model exhibits slightly faster prediction degradation. Unlike neural networks, which can learn to adaptively dampen or stabilize unsteady behaviour, DMDc relies on a purely linear, open-loop operator. This makes it inherently more susceptible to noise amplification and model drift, especially when extrapolating beyond the training region or under higher frequency oscillations, as seen in the SH scenario.

The differences between AeroNet and DMDc models can be attributed to their underlying methodologies. DMDc inherently carries specific constraints, such as the requirement for explicit stabilization procedures and calibration within a relatively narrow operating envelope. Consequently, DMDc is highly sensitive to deviations from the training conditions, limiting its general robustness. In contrast, AeroNet models attain comparable or better accuracy without relying on such stringent stabilization and tuning steps. This allows them to perform robustly across a wider range of dynamic conditions.

The trade-off of employing AeroNet over traditional methods such as DMDc lies in interpretability. The neural network acts as a black box, optimizing weights solely to minimize predictive errors without explicitly reflecting physical flow mechanisms. Thus, despite AeroNet superior adaptability and performance in dynamically challenging scenarios, its physical interpretability remains limited.

8.5 Computational Cost Analysis

A comprehensive analysis of computational costs was performed to compare the efficiency of the proposed AeroNet model with that of a higher-order approach and DMDc tecnique, as shown in Table 8.7. A single CFD run on a high-performance computing system, using an Intel Skylake-based architecture with 3 nodes and 40 CPU cores per node, typically requires around 6,000 CPU hours. Generating the entire dataset demands approximately 75,000 CPU hours. In contrast, the proposed framework predicts a full signal in under two minutes on a local machine with a NVIDIA RTX A4000 GPU, with a per-timestep inference time of 0.15 seconds (6.67 frames-per-second), yielding over 99% computational savings with respect to CFD. The DMDc model, once trained, is still faster in wall-clock terms (60 frames-per-second), but (as shown in Section 8.4.6) at the cost of lower prediction accuracy. Finally, we note that the large up-front expense of producing high-fidelity training data motivates future work on transfer-learning or active-learning strategies.

CFD (CPU hours)		**AeroNet (GPU hours)**				
Simulation		Pre-Trained AE	Training		Prediction	
(12 runs)	(1 run)	(Optimisation + Training)	(ARMA)	(FF)	(ARMA)	(FF)
75,000	6,000	42.6	35.2	33.1	0.03	0.03

DMDc (CPU hours)	
Training	Prediction
0.15	0.003

TABLE 8.7: Computing cost comparison between AeroNet models, DMDc and CFD. FF: FeedForward.

8.6 Conclusions

This chapter presented a framework for predicting unsteady transonic wing pressure distributions, integrating an AE architecture with GCN and graph-based temporal layers to capture time dependencies. Tested on a grid consisting of 86,840 surface points, the proposed model effectively compresses high-dimensional C_P distribution data into a lower-dimensional latent space using the AE, preserving essential features for accurate representation. The GCN layers are well-suited for handling the unstructured grids characteristic of aerodynamic data, while the temporal layers capture and leverage temporal dependencies for robust forecasting of wing C_P distributions.

Our results demonstrated that this integrated approach can achieve an accuracy comparable to traditional CFD methods, while significantly reducing computational costs, requiring initially about 76 GPU hours for training and then less than 2 minutes to predict an entire signal. We evaluated two architectures, the feedforward model and the ARMA model, using four different temporal layers (GRU, LSTM, STGCN, ATTENTION), with the STGCN layer consistently delivering the most accurate results across the validation signals. The feedforward model demonstrated clear advantages in terms of predictive accuracy and stability, particularly in avoiding the error propagation inherent in the ARMA model. By not relying on its own predictions for subsequent inputs, the feedforward model is better suited for dynamic simulations ranging from steady-state to high reduced frequency, showing greater accuracy in predicting complex flow features, such as shock waves and flow separation. The ARMA model, while capable of capturing general trends, is more prone to error accumulation, particularly in scenarios with high-frequency oscillations or rapid changes in aerodynamic forces. Nonetheless, when using ground-truth inputs, the ARMA model can yield highly accurate results, underscoring its potential in scenarios with reliable data inputs. Both AeroNet variants outperformed the established DMDc approach, particularly in high-frequency, dynamically complex situations. Notably, AeroNet achieves superior accuracy without requiring explicit stabilization or calibration, indicating greater robustness and adaptability.

Sensitivity analyses demonstrated that the proposed framework inherently distinguishes between elliptic and hyperbolic flow characteristics typical of transonic flows. Localized perturbations in subsonic (elliptical) regions affect global pressure distributions, while perturbations in supersonic (hyperbolic) regions propagate downstream within localized Mach cones. These findings confirm AeroNet capability to accurately capture the distinct physical behaviours associated with different aerodynamic flow regimes.

From a broader thesis perspective, these outcomes underscore:

1. **Scalable Spatio-Temporal Modelling**
   By integrating graph convolutions with temporal layers, the chapter demonstrates

a powerful approach to unsteady flow prediction that preserves essential nonlinearities while maintaining computational efficiency. This directly addresses Research Objective O4, which focuses on developing surrogate models capable of accurately forecasting time-resolved surface vector fields in transonic conditions.

2. **Comparative Performance with Projection-Based ROMs**
   The ML-based ROMs developed in this chapter demonstrate improved predictive capabilities over DMDc in modelling high-dimensional and nonlinear unsteady flows. Whereas projection-based approaches remain attractive for their interpretability and negligible online cost, they are constrained by mode truncation and the need for explicit stabilisation. In contrast, the proposed ML approaches offer greater flexibility and generalisation across varying flow conditions, despite being less interpretable and requiring more computing resources during training.

3. **Sensitivity to Local Flow Regime**
   The ability of the model to capture and differentiate between elliptical and hyperbolic flow regions confirms the importance of integrating spatio-temporal context. This suggests that the GCN architecture implicitly encodes spatial anisotropies in transonic flows, enabling the network to adjust its predictive attention based on localised flow features.

While initially validated on a specific wing geometry and Mach number, the graph-based framework is inherently capable of accepting diverse flow conditions (e.g., different M and Reynolds numbers) and geometric perturbations (e.g., structural modes) as input features. This allows the surrogate to generalise across a broader parametric space, supporting future applications involving shape optimisation and structural deformation. Approaches such as subgraph splitting and padding may be introduced to handle new spatial structures or significantly larger graphs, while maintaining an important balance between capturing long-range dependencies and preserving computational feasibility.

Chapter 9

Graph-Convolutional Autoencoder for Modal Space Analysis

This study introduces a geometric deep learning framework for rapid aerodynamic predictions on deformed wings, focusing on the Agard 446.5 test case. Building upon the spatial modelling approach of Chapter 7, this work extends the graph-based autoencoder framework to a modal space analysis. Using graph convolutional layers within an autoencoder architecture, the method efficiently handles unstructured surface meshes and significantly lowers computational cost compared to traditional CFD. Two predictive approaches are developed, each using modal coordinates to represent structural deformations and capture a wide range of geometry variations. Addressing Research Objective O5, the chapter targets fast and accurate prediction across modal deformation spaces, enabling efficient analysis of structurally perturbed configurations. By combining graph-based encoding with physically meaningful modal inputs, the framework maintains geometric consistency even for large deflections. Both approaches show good accuracy and reliability in predicting aerodynamic forces and flowfields across multiple deformations. Their rapid inference also supports integration into optimisation workflows for accelerated modal space exploration.

9.1 Introduction

The design of an aerospace system often involves developing and refining the geometry of critical components to meet stringent performance requirements. In aircraft design, for instance, this development follows an iterative process in which aerodynamic performance must be evaluated for representative loading conditions at each new shape configuration. Multiple flight conditions need to be assessed for each design iteration to provide aerodynamic engineers with sufficient information to propose subsequent shape modifications.

An increasing industrial demand for rapid design cycles underscores the importance of modelling platforms that can deliver both high accuracy and faster simulation. Long CFD runs slow design progress, potentially leading to financial losses, while inaccurate predictions risk leading the process toward suboptimal geometries. As a result, modern research is increasingly focused on models that can efficiently balance speed and fidelity.

ROMs based on deep learning are attractive because they can offer gains on both fronts. However, published works that use these methods for aerodynamic shape modelling are narrow, in particular for 3D configurations. Most contributions target 2D cases [177, 331, 343, 380]. For example, Chen et al. [63] applied generative adversarial networks (GANs) to aerofoil optimisation and required more than 1,600 samples to train a 2D model. 3D studies are comparatively rare and often address problems that are not directly comparable to surface flowfield prediction on wings. Larrañaga et al. [193] optimised thermal fields on a finned surface with more than 15,000 CFD samples. Zan et al. [399] predicted force coefficients from geometric parameters, which does not directly address distributed flowfields that are critical for design decisions. In addition, CNNs typically assume a Cartesian layout, which limits their direct applicability to unstructured CFD meshes.

For irregularly discretized domains, GNNs are generally more suitable [44, 46]. In this context, Baqué et al.[23] employed geodesic convolution for aerodynamic optimisation on simplified car shapes, later extending the method to the shaping of a fixed-wing drone[22]. However, geodesic convolutions often introduce significant computational overhead due to complex kernel operations [228, 244]. By contrast, GCN layers [183] offer a simpler, more scalable formulation, making them appealing for optimisation tasks. Recent work has also explored message-passing (MP) GNNs [126], wherein edge embeddings capture local interactions at potentially higher computational expense. For instance, Duthé et al.[102] applied a MP-GNN to predict aerodynamic flow around a 2D aerofoil given sparse chordwise pressure measurements, and Jin et al.[171] employed a similar approach for 3D stress optimisation on a wheel. Meanwhile, state-of-the-art GNNs for physics simulation (e.g., mesh-graph-net [266, 303, 304]) also rely on edge embeddings and thus carry notable computational costs.

Despite this progress, previous studies on 3D, parametric surface flowfield prediction over unstructured meshes remain limited. The combination of geometric variability, mesh dependence, and the need for intensive supervision to resolve complex transonic features makes the problem highly data-intensive and methodologically challenging.

Therefore, to advance the state of the art, we introduce a GDL framework for aerodynamic prediction on parametric 3D shapes, with particular focus on aircraft wings relevant to shape optimisation. The proposed approach is based on two parallel GCN-driven, multi-resolution autoencoders that map modal deformation parameters to distributed aerodynamic quantities on the wing surface. By embedding graph convolutions within hierarchical encoder-decoder structures, the models achieve dimensionality reduction while preserving local flow features. Specifically, we develop the graph-convolutional multi-mesh autoencoder (GCN-MM-AE) and an enhanced gradient-based autoencoder graph-convolutional network (GB-AE-GCN), which extends the steady-state formulation of Chapter 7 to deformed geometries. In contrast to more elaborate message-passing GNNs with edge embeddings, the present architectures prioritise computational efficiency by relying on lightweight GCN layers, making them more suitable for integration within iterative design and optimisation loops.

To demonstrate these methods for aerodynamic flow prediction on varying shapes, we use the Agard 445.6 wing [393], widely recognized as a canonical test model for aeroelastic studies. The dataset comprises CFD solutions for different wing configurations derived from combinations of modal deflections. We employ our two proposed GCN-based approaches to learn the surface flow distribution as a function of design parameters. The final objective is to support rapid design iterations of wing geometries through efficient flowfield predictions.

9.2 Methodology

We propose a graph-based AE framework for predicting aerodynamic fields over varying three-dimensional wing shapes. By integrating GCNs within a hierarchical approach, the network learns compact latent representations of high-dimensional, unstructured surface data while preserving important flow features at different resolutions, discretizing in a cycle similar to classical multi-grid schemes for CFD solutions [235]. In the encoding phase, GCN layers at each grid level aggregate features from neighboring nodes, capturing local topology and aerodynamic gradients. Pooling operations then coarsen the mesh in a hierarchical fashion, reducing the number of nodes and compressing the spatial information into a latent state. During decoding, unpooling operations revert each coarsened graph to its previous resolution, assisted by additional GCN layers that refine and reconstruct the aerodynamic fields to match the original mesh.

In addition to general shape parameters, the framework accommodates problem-specific inputs tailored to the deformation space under consideration. In the Agard test case discussed in the next section, the input variables consists of the amplitudes of the wing deflection modes, $\boldsymbol{m} = [m_1, m_2, \ldots, m_m]$, each describing a particular structural deformation, where m is the number of modes. These modal amplitudes span the subspace of static aeroelastic deformations that the structure can exhibit under aerodynamic loading. Together with the mesh coordinates $\boldsymbol{xi}$ of each surface node, they define the node features provided to the model. Embedding both modal parameters and geometric coordinates is important for learning a robust spatial mapping of aerodynamic phenomena to any design configuration. The output fields at each node include a set of flow-related quantities, pressure and shear stress coefficients $\boldsymbol{Y} = \left[C_P, C_{fx}, C_{fy}, C_{fz}\right]$, which are decoded back to the original high resolution mesh.

Two variants of this approach have been developed to address the computational and accuracy trade-offs commonly encountered in aerodynamic modelling. The first model, GCN-MM-AE, employs a simple hierarchical scheme with a grid-density based coarsening method and a standard loss function that focuses on pointwise errors in predicted variables. Despite its simplicity, this model effectively exploits unstructured surface data, leading to promising initial results in shape-to-aerodynamics mapping. However, it has a relatively large number of trainable parameters and relies on manual hyperparameter tuning.

Building upon these insights, the second model, GB-AE-GCN, incorporates gradient-based pooling strategies, a physics-informed loss function, and a systematic hyperparameter optimisation procedure. The gradient-based pooling retains nodes with strong flow gradients longer in the coarsening hierarchy, thus preserving critical aerodynamic details. The physics-informed loss augments standard pointwise errors with penalties for discrepancies in integrated quantities (pitching moment), guiding the model to maintain consistency between global loads and local flow distributions. Finally, Bayesian optimisation of hyperparameters allows for smarter selection of network configurations, improving prediction accuracy while reducing the total number of parameters.

Both approaches accept as node features the shape-specific parameters - often derived from mode amplitudes or geometric coordinates - along with the spatial positions of each mesh node. Inference then proceeds by encoding these node features into a latent space through successive graph convolutions and pooling steps, followed by reconstruction through unpooling and GCN layers. By performing multi-scale compression of the unstructured grid data, the framework preserves important aerodynamic details while reducing computational overhead. Operating directly on unstructured grids makes it highly adaptable to diverse geometries and flow regimes, allowing rapid evaluations of distributed aerodynamic fields in newly deformed configurations. This capability substantially accelerates modal space exploration and shape optimisation.

Approach 1: GCN-MM-AE

As shown in Figure 9.1, this first approach applies a grid density-based pooling to coarsen the unstructured grids, facilitating an efficient dimensionality reduction stage for aerodynamic data. Inspired by multi-grid PDE methods [236], the mesh undergoes progressive resolution changes between blocks of graph-convolutional layers, capturing essential features at multiple spatial scales. Nodes are stochastically selected for removal based on local face areas, so that finely resolved regions with complex flow structures retain a higher mesh density, whereas more uniform areas can be aggressively coarsened. The probabilistic selection function and related details follow the procedures outlined in Chapter 7. Once the coarse mesh has been generated, node-by-node interpolation and transfer are performed via the MWLS technique (see Chapter 7 for specifics). MWLS leverages a second-order polynomial basis, weighting node contributions according to their spatial proximity, thus striking a balance between interpolation accuracy and computational efficiency. During encoding, the source mesh data are pooled down to coarser grids, while unpooling reintroduces fine-scale structure in the decoder by mapping latent states back to the original resolution.

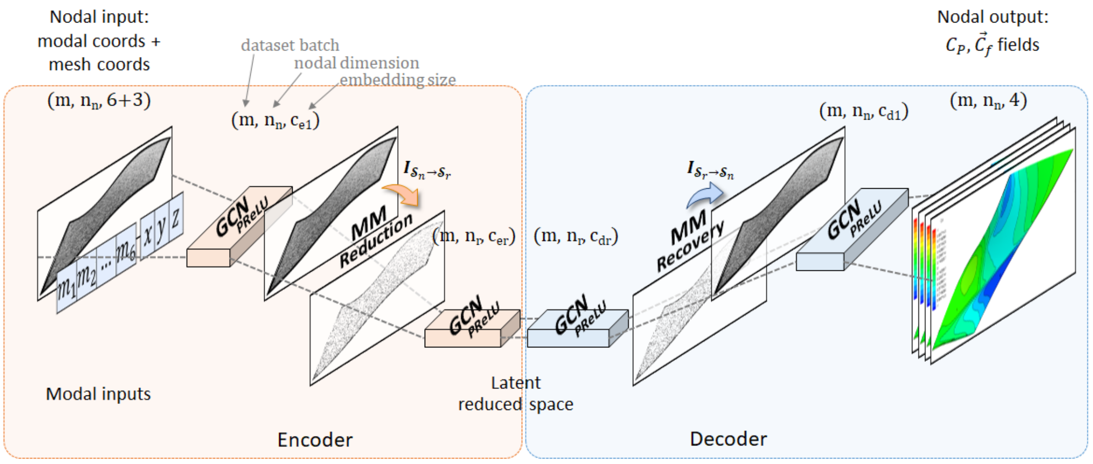


FIGURE 9.1: Approach 1: GCN-MM-AE model architecture using one-step grid density-based pooling with standard loss [230].

Approach 2: GB-AE-GCN

Figure 9.2 illustrates the second approach, which extends the GCN-MM-AE by incorporating multiple gradient-based space reductions, Mahalanobis distance-based connectivity reconstruction, and a physics-based loss. Specifically, node selection here focuses on retaining higher-gradient regions—rather than purely grid-density based considerations—so that areas featuring shock waves or flow separations are preserved with finer detail. The gradient-based selection criteria (including pressure gradient evaluation, reference node definitions, and point-based probability functions) are described in Chapter 7. Once nodes are selected, the connectivity between them is reconstructed using Mahalanobis distance, which significantly reduces false connections compared to simple Euclidean distance method. MWLS interpolation remains the core mechanism for transferring flow values to and from the coarse grids.

In addition to a standard `MAE` term, the loss function explicitly penalizes inaccuracies in the pitching moment coefficient, a key global aerodynamic quantity sensitive to shock placement. Mathematically:

$$Loss = \texttt{MAE} + \lambda \cdot C_M$$

with $\lambda = 0.01$ for dimensional consistency. This physics-based penalty consistently refines flowfield predictions in critical regions [285].

Lastly, a Bayesian optimisation strategy explores the hyperparameter design space, including the number of layers per block, units per layer, and compression/expansion ratios, to balance model complexity and accuracy. The number of layers per block defines groups before or after spatial reductions, mirrored in encoding and decoding to save resources. Units per layer represent neuron counts in each GCN module for dimensional compatibility. Additionally, the compression/expansion value controls the ratio between coarser and finer meshes during encoding and decoding. Table 9.1 outlines the hyperparameter ranges. A total of 30 trials, each limited to 500 epochs, were run to identify the optimal configurations. The best performing architecture was further trained for 2000 epochs, resulting in an efficient, high-fidelity surrogate for unstructured aerodynamic data.

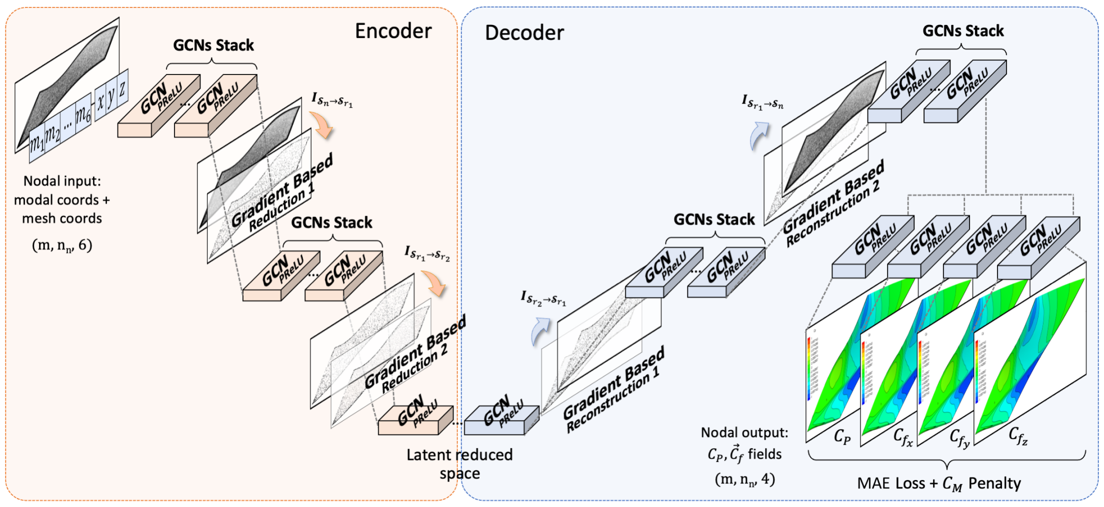

Encoder
Decoder
GCNs Stack
GCN PReLU
$m_1\ m_2\ \dots\ m_6\ x\ y\ z$
Nodal input: modal coords + mesh coords
(m, n_n, 6)
$I_{s_n \to s_{r_1}}$
Gradient Based Reduction 1
$I_{s_{r_1} \to s_{r_2}}$
Gradient Based Reduction 2
Latent reduced space
$I_{s_{r_2} \to s_{r_1}}$
Gradient Based Reconstruction 1
$I_{s_{r_1} \to s_n}$
Gradient Based Reconstruction 2
Nodal output: $C_P, \vec{C}_f$ fields
(m, n_n, 4)
C_P
C_{f_x}
C_{f_y}
C_{f_z}
MAE Loss + C_M Penalty

FIGURE 9.2: Approach 2: GB-AE-GCN model architecture using two-step gradient-based pooling with physics based custom loss [163].

By combining gradient-based mesh coarsening, robust connectivity techniques, and a custom loss function, the GB-AE-GCN model enhances predictive accuracy across high-gradient transonic regions while keeping model size manageable, making it a strong candidate for rapid aerodynamic design iterations.

A comprehensive overview of the architecture of the two proposed approaches is detailed in Appendix A.7.

9.3 Test Case

The Agard 445.6 wing, originally developed by NASA for aeroelastic research, is frequently chosen as a benchmark model due to its swept planform and use of the NACA 65A004 aerofoil [393]. As illustrated in Figure 9.3, the wing has a mean aerodynamic chord of $c = 0.464\,[\text{m}]$, a semispan of $b = 0.762\,[\text{m}]$, and a reference surface area $S = 0.353\,[\text{m}]^2$. The moment reference point is taken at $x_{ea} = c/2$.

In the transonic conditions considered ($M = 0.96$), a primary shock forms on the suction side of the wing, yet the boundary layer remains attached and no shock–induced separation is observed. Viscous effects are therefore secondary for the present setup, so the dominant nonlinearity arises from compressibility (shock). As incidence increases, the local normal Mach number rises and a near-normal shock develops over the mid- to outboard span. In the absence of separation, the shock remains comparatively thin and its streamwise excursion is primarily governed by the instantaneous pressure recovery required to balance the inviscid outer flow. This yields a sharp but reversible change in chordwise pressure gradient across a narrow region, producing strong, spatially localised sensitivity to small variations in effective incidence and sweep.

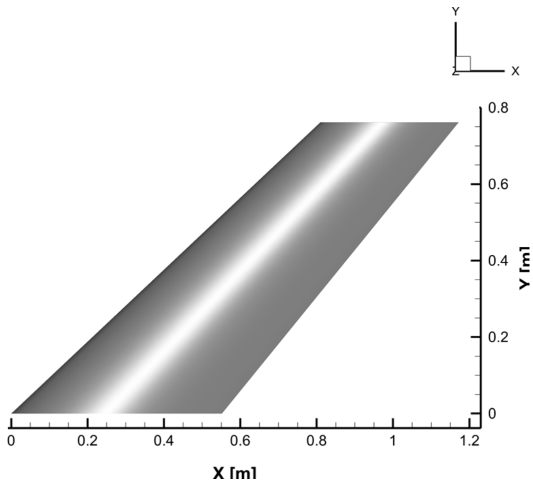
On the other hand, the aeroelastic deformations (bending and torsion) modify the local incidence, camber, and effective sweep, thereby altering the normal Mach component and displacing the shock. The resulting load redistribution is amplitude dependent: increasing torsional twist advances and strengthens the shock, while outboard bending modifies the spanwise pressure gradient and the shock curvature. Even with attached flow, these geometry-induced changes lead to a nonlinear relationship between generalised coordinates and aerodynamic coefficients.

For the CFD analysis, a grid comprising 12,334,685 volume cells was generated, with 45,943 nodes on the wing surface, as shown in the lower panels of Figure 9.3. All simulations were carried out using SU2 v7.5.1 solver at a freestream Mach number of 0.96

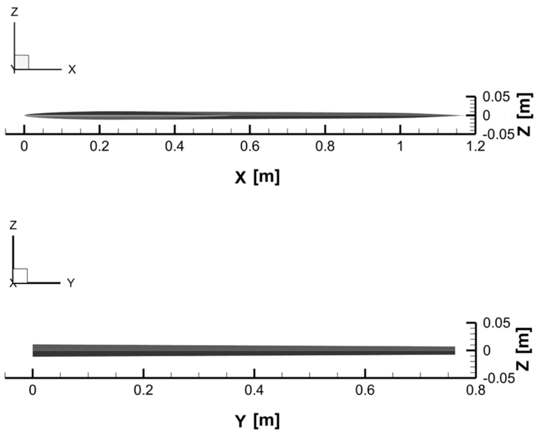
Hyperparameter	**Value**	**Step size**
Compression ratio	1/4 - 1/3 - 1/2	–
Number of Hidden Layers per Block	1 to 3	1
Number of Neurons per Hidden Layer	32 to 512	16

TABLE 9.1: Hyperparameters design space.

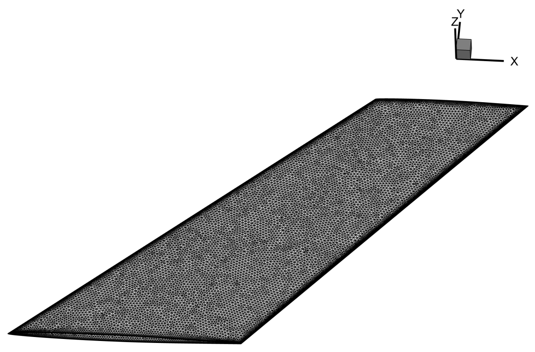
and a Reynolds number of 4.51×10^5, corresponding to a velocity of $318.07\,\mathrm{m\,s^{-1}}$. The one-equation SA turbulence model, discretized with a scalar upwind scheme, was employed to capture viscous effects. The JST method with viscous damping provided convective-flux discretisation, while the biconjugate gradient stabilizer (with ILU preconditioning) and multigrid V-cycle were used to solve the linear system efficiently. Gradient reconstruction was handled via a Green-Gauss approach, and a CFL number of 20.



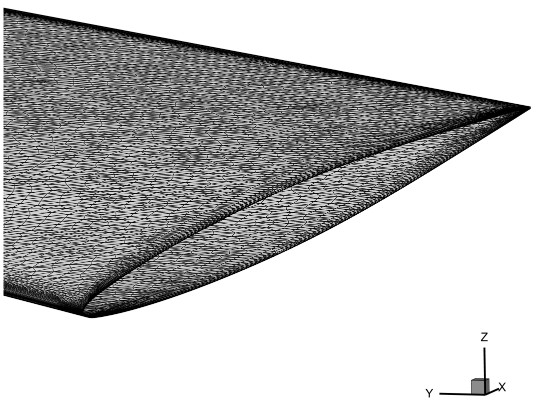
(A) Top view.



(B) Front (bottom) and side (top) views.



(C) Isometric view.

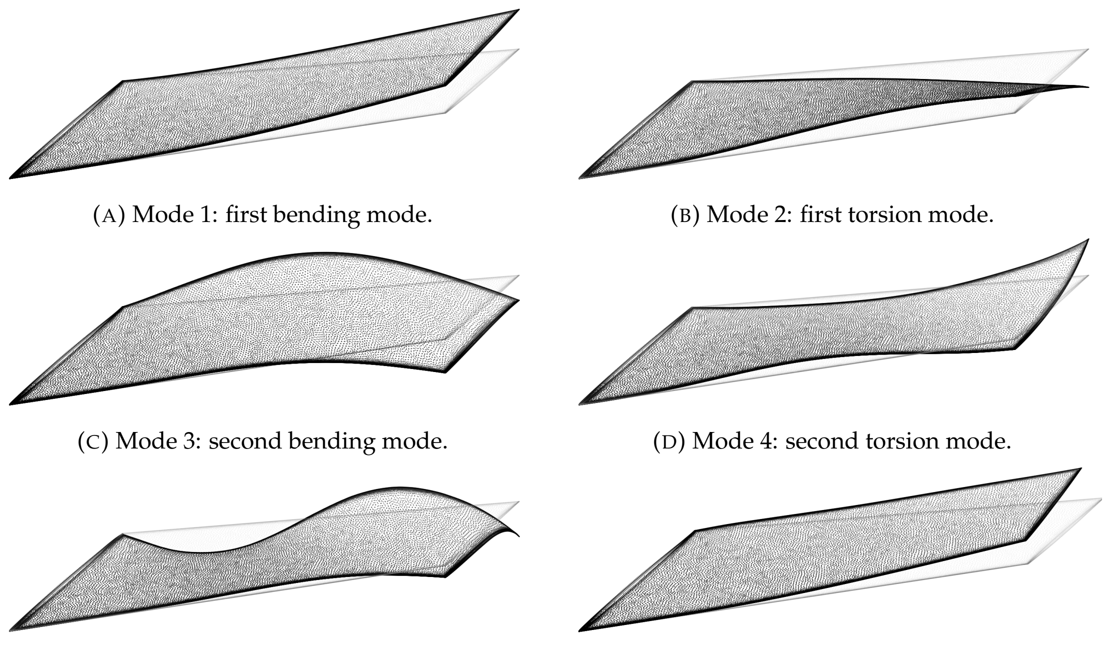


(D) Close view of the wing chord mesh.

FIGURE 9.3: Details of the Agard 445.6 wing and CFD surface mesh adopted.

9.3.1 Structural Deflection for Shape Variation

The Agard wing is structurally flexible, and its finite element (FEM) analyses [393] provide six major deflection modes, shown in Figure 9.4 (adapted from Li et al. [202]). These include the first and second bending modes, torsional modes, and an in-plane bending mode. By superimposing these modes with different amplitudes, a wide range of deformed wing shapes can be generated for parametric studies.



(A) Mode 1: first bending mode.

(B) Mode 2: first torsion mode.

(C) Mode 3: second bending mode.

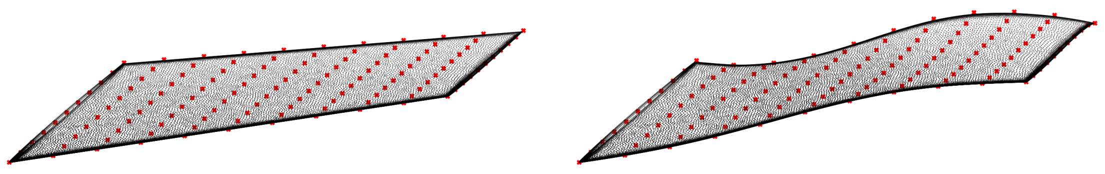
(D) Mode 4: second torsion mode.

(E) Mode 5: third torsion mode.

(F) Mode 6: in-plane bending.

FIGURE 9.4: Shape of the six deflection modes from the Agard 445.6 FEM model.

To map the FEM-based displacements to the CFD surface mesh, the displacements at the FEM nodes are interpolated to the corresponding mesh points, and then the volume mesh is morphed using the SU2_DEF utility. Figure 9.5 illustrates a representative deformation: the left panel shows the undeformed mesh, while the right panel shows an example deformed shape. In both views, FEM nodes (red) are superimposed on CFD mesh points (black) to demonstrate the consistency of the displacement interpolation and subsequent mesh morphing process.



(A) Undeflected condition.

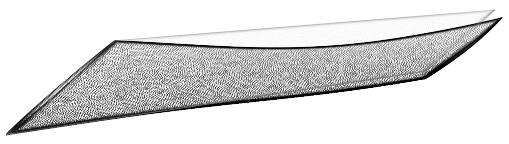
(B) Deflected condition.

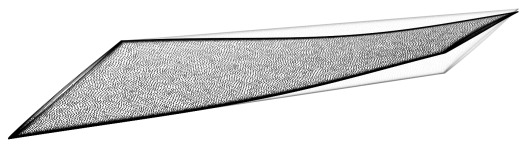
FIGURE 9.5: Comparison between undeflected and deflected meshes of the Agard 445.6 wing, for the FEM (red) and CFD (black) surface mesh.

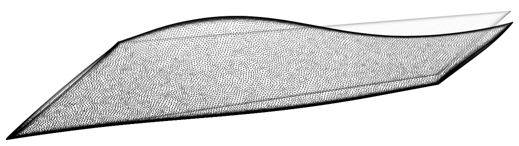
9.3.2 Wing Shape Dataset

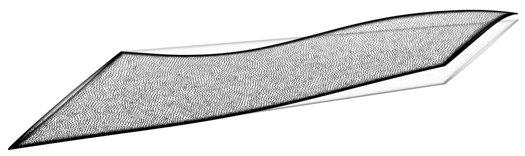
To explore the modal deformation space, 250 deflected wing geometries were generated by scaling each of the six deflection modes to a maximum displacement of 10.0 [mm]. LHS scheme was then used to construct the Design of Experiments (DOE) for these mode amplitudes, $\boldsymbol{m} = [m_1, m_2, ..., m_6]$. For larger deformations, the SU2 solver was prone to divergence due to degraded mesh quality; therefore, each mode amplitude was restricted to the range $\in [-75, 75]$ [mm], corresponding to about 10% of the wing span at the most extreme deformations. A total of 251 samples were assembled, including the undeflected reference shape.

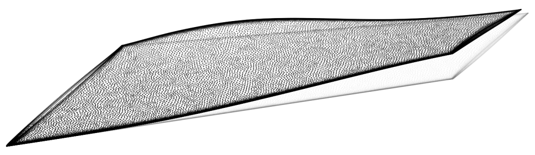
Figure 9.6 shows examples of the resulting geometries, highlighting the wide variety of shape changes achievable through different combinations of deflection modes. The sampled amplitude values from the LHS are also noted, illustrating how mixing the different modes produces both subtle and more pronounced wing deformations.


(A) $\boldsymbol{m}$ = -65.7, -0.3, -65.7, 34.5, -2.7, -70.5.


(B) $\boldsymbol{m}$ = 26.1, 60.9, -68.7, -50.7, -19.5, 2.7.


(C) $\boldsymbol{m}$ = -49.5, -18.9, 24.9, -53.7, -71.1, -13.5.


(D) $\boldsymbol{m}$ = 62.7, -9.3, -35.7, -38.1, 45.3, 5.7.


(E) $\boldsymbol{m}$ = 59.1, 65.7, 26.7, -34.5, -35.7, 47.7.

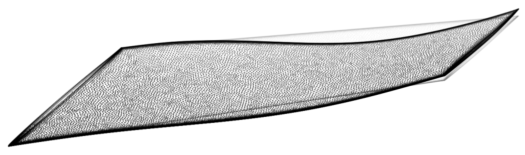

(F) $\boldsymbol{m}$ = -29.7, -33.9, -42.9, -33.3, -32.1, -2.1.

FIGURE 9.6: Snapshots of morphed configurations based on the Agard wing, as part of the dataset of wing geometries for the generation of the GNN based aerodynamics model. The mode amplitude variables are also reported.

9.3.3 Background Aerodynamics

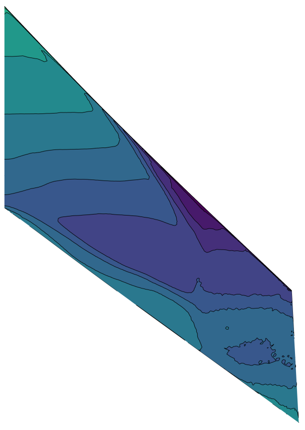
Figure 9.7 shows the C_P distribution for six examples obtained from the `SU2` solutions, corresponding to the deformations reported in Figure 9.6. The pressure distributions exhibit significant variation across the selected configurations, demonstrating how changes in the wing shape, driven by different combinations of deflection modes, lead to pronounced alterations in the surface flow characteristics. Notably, regions of high and low pressure shift depending on the deflection mode, with certain configurations showing concentrated areas of high pressure near the leading edge, while others display more distributed pressure fields. These differences highlight the sensitivity of the aerodynamic performance to shape deformations, particularly in terms of how pressure distributions affect the overall C_L, C_D, and C_M. This variability emphasizes the importance of developing a GNN-based framework capable of capturing the different geometric features as well as predicting the different fluid phenomena observed on the wing surfaces.

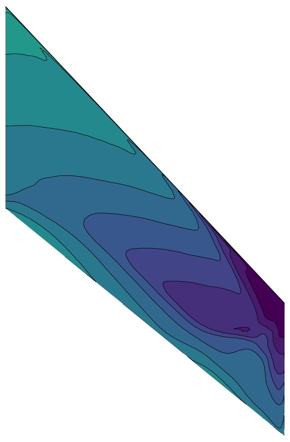
9.4 Results

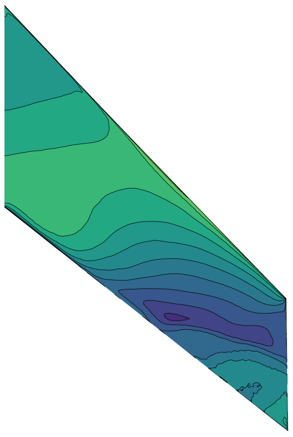
The results for the different modelling approaches are presented to evaluate their performance. We demonstrate the effectiveness of the mode-based models described in Section 9.2 in simulating the morphing Agard wing.

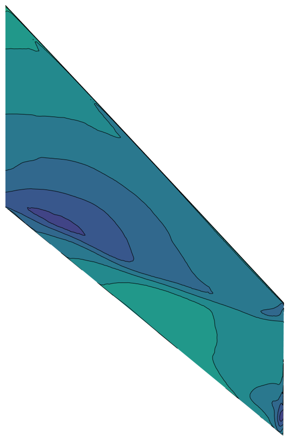
From the DOE containing 251 samples, 153 (60%) were selected for training. An additional 57 samples (25%) were set aside for validation to fine-tune the hyperparameters, while the remaining 41 samples (15%) were reserved for testing, following standard practices. To split the dataset into these three sets, we used a stratified sampling method based on cluster analysis [346]. The stratification process started by applying K-means clustering to classify the six deflection modes, using different values of k to group the samples into different clusters for stratification. Each cluster at this stage contains a group of samples with similar deflection characteristics. We then used the StratifiedK-Fold [260] to apply stratification based on the clusters to maintain similar proportions of clusters across the training, validation, and test subsets. This ensured that each split contained representative data points for the different deflection modes in the dataset. To determine the optimal value of k, we calculated a correlation index between the six features within clusters, as a sum of all elements in the lower triangular portion of the correlation matrix. The final choice of $k = 20$ was made to minimize the feature correlation and thereby maximize the diversity of deflection modes across clusters.

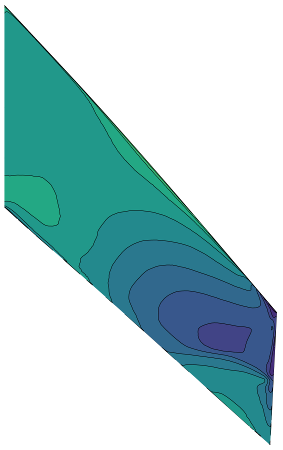
Figure 9.8 illustrates the effectiveness of the stratified sampling strategy, showing a well-distributed representation of the six deflection modes across the datasets. The main diagonal of the pairplot matrix shows the final distributions of samples, which for all six modes mostly result in a good non-skewed and symmetric set of Gaussians. In addition, the cluster distribution highlights the diversity within the data: this ensures that each mode is balanced across the training, validation, and test sets, while guaranteeing that the dominant and more representative deflections are included in the same proportion along the datasets. This approach also allows the validation and test sets to better assess the generalisation capabilities of the models, while providing representative samples in the training set to help the model to capture complex physical phenomena.

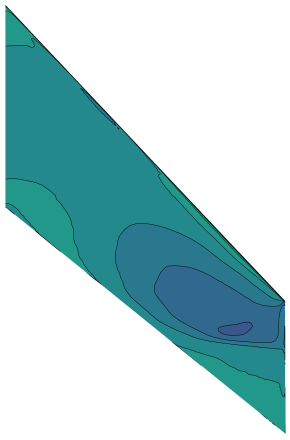

(A) $\boldsymbol{m}$ = -65.7, -0.3, -65.7, 34.5, -2.7, -70.5.

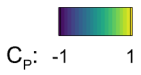

(B) $\boldsymbol{m}$ = 26.1, 60.9, -68.7, -50.7, -19.5, 2.7.


(C) $\boldsymbol{m}$ = -49.5, -18.9, 24.9, -53.7, -71.1, -13.5.


(D) $\boldsymbol{m}$ = 62.7, -9.3, -35.7, -38.1, 45.3, 5.7.


(E) $\boldsymbol{m}$ = 59.1, 65.7, 26.7, -34.5, -35.7, 47.7.


(F) $\boldsymbol{m}$ = -29.7, -33.9, -42.9, -33.3, -32.1, -2.1.



(G)

FIGURE 9.7: Pressure distribution on the upper surface from CFD for the deflected Agard configurations of Figure 9.6.

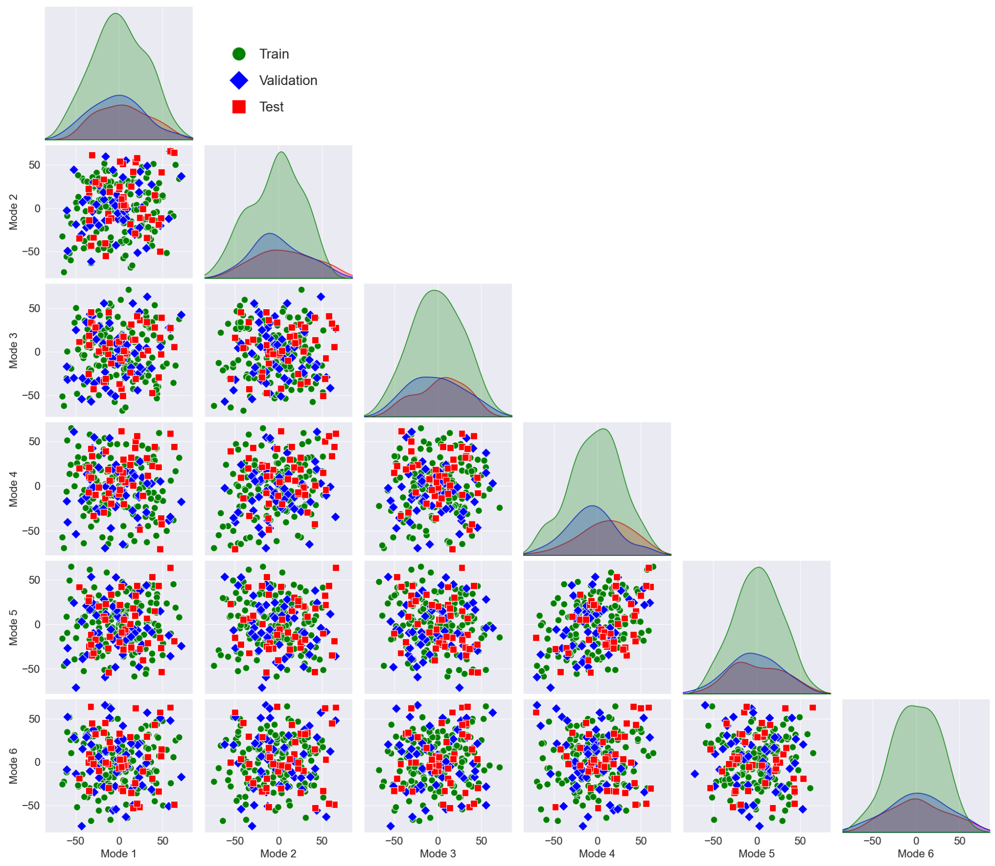


FIGURE 9.8: Pairplot of stratified sampling strategy for AGARD 445.6 deflection modes.

9.4.1 Aerodynamic Force Predictions

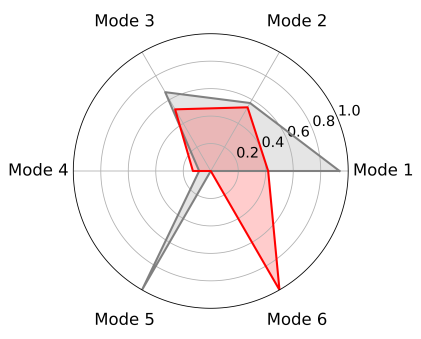
This section evaluates the prediction accuracy of the models for the integrated aerodynamic coefficients. It is important to note that these coefficients (C_L, C_D, and C_M) are not directly predicted by the models but are instead obtained by integrating the pressure and shear stresses across the wing surface. Table 9.2 presents the error metrics for the two modelling approaches. Three metrics are computed for comparison: `MAPE`, `RMSE`, and `R2`. Compared to the GCN-MM-AE model, the GB-AE-GCN one shows a 17% reduction in `MAPE` for C_L, a 41% reduction for C_D, and a 27% reduction for C_M. In terms of `RMSE`, GB-AE-GCN achieves a 28% reduction for C_L, 38% for C_D, and 31% for C_M, showing marked improvement in overall prediction accuracy. The `R2` further support these improvements, with GB-AE-GCN achieving higher predictive accuracy scores across all metrics: C_L at 0.9951, C_D at 0.9937, and C_M at 0.9866.

Figure 9.9 highlights the relative correlations between the three aerodynamic coefficients errors and the six modes for both the GCN-MM-AE and GB-AE-GCN models, using a radar chart. This illustrates how architectural complexity shifts the way each model interacts with structural modes. The correlation values for each mode were first calculated using the Pearson correlation coefficient between the mode values and the

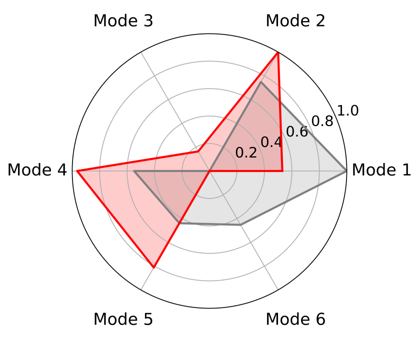
Model	**MAPE**			**RMSE**			**R2**		
	C_L	C_D	C_M	C_L	C_D	C_M	C_L	C_D	C_M
GCN-MM-AE	1.1282	1.5782	1.8838	0.0229	0.0326	0.0383	0.9914	0.9841	0.9736
GB-AE-GCN	0.9361	0.9284	1.3774	0.0165	0.0203	0.0264	0.9951	0.9937	0.9866

TABLE 9.2: Comparison of C_L, C_D, and C_M prediction accuracy metrics on the test set for two modelling approaches.

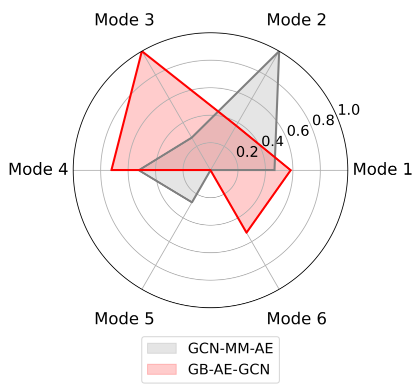
respective error quantities. The absolute values of the correlations were then normalized between 0 and 1 to facilitate comparison across modes. In the radar chart, each axis represents one of the six modes, while the distance from the centre of the chart indicates the strength of the correlation. A value closer to 1 represents a stronger relationship between the mode and the corresponding error, while values near 0 suggest minimal correlation. The GB-AE-GCN model tends to be more influenced by bending modes for C_L and C_D, while torsional modes primarily impact its C_M predictions. Conversely, the GCN-MM-AE model is more sensitive to torsional modes for C_L and C_D, and to bending modes for C_M.



(A) Lift coefficient.



(B) Drag coefficient.



(C) Pitching moment coefficient.

FIGURE 9.9: Radar chart showing the normalized correlation between the six modes and aerodynamic coefficients errors for the two modelling approaches.

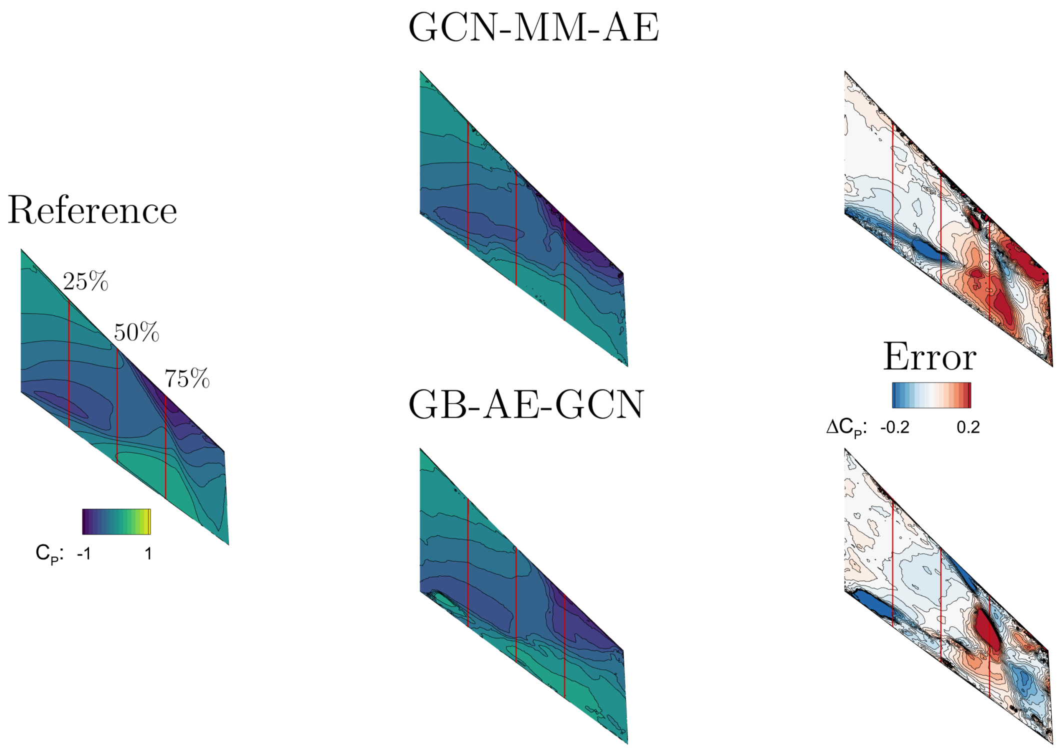
9.4.2 Vector Field Predictions

Beyond the aerodynamic loads, the focus lies on evaluating the framework ability to accurately capture surface flow features. In this section, we analyse the distributed quantities predicted by the models to assess their performance in replicating complex flow fields. We begin by presenting the results for the worst-predicted test sample in terms of the three integral aerodynamic loads for both approaches. This analysis highlights the models limitations and identifies areas where improvements in predicting flow features are necessary. However, it is important to emphasize that, despite the displayed samples, both models demonstrate robust accuracy across the broader modal space, consistently capturing key aerodynamic features and maintaining reliable performance.

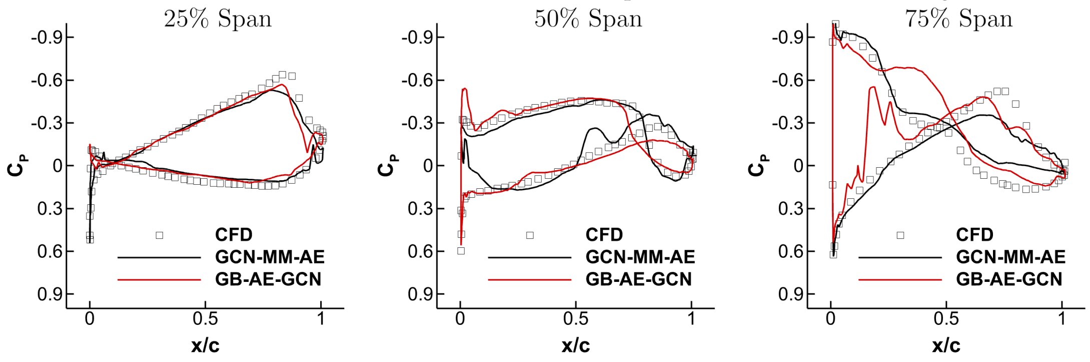
Figures 9.10 and 9.11 illustrate the prediction comparison of C_P and C_F fields for the two modelling approaches, GCN-MM-AE and GB-AE-GCN, on the test sample with the highest error from the GCN-MM-AE model. In the upper panels, the contour plots depict the CFD reference solution (left), the models prediction (middle), and the resulting error distributions (right), highlighting the regions where each model diverges from the reference values. The discrepancies align with regions where transonic flow features are most sensitive to geometric perturbations. Depending on the deflection mode, this shock can shift in position and intensity, modifying the local pressure gradients and, consequently, the surface shear stress distribution. These shock–induced variations are particularly evident near mid-span, where abrupt pressure jumps translate into steep C_P and C_F gradients that challenge both models. While the GB-AE-GCN is more effective at mitigating oscillatory artefacts, neither approach fully resolves the abrupt pressure changes. This limitation reflects the underlying physical difficulty of approximating discontinuous fields within a smooth latent representation, especially in regions where structural deflections amplify aerodynamic nonlinearities. The lower panels illustrate the C_P and C_F distributions along cross-sections at 25%, 50%, and 75% of the span. The GB-AE-GCN model does align more closely with CFD, yet subtle and oscillatory deviations in high-gradient zones persist.

Table 9.3 quantifies these observations, presenting prediction errors for C_L, C_D, and C_M between the two modelling approaches. The GCN-MM-AE model consistently exhibits higher errors across these metrics, with the most significant difference observed in C_M. The increased complexity of the GB-AE-GCN model improves its ability to capture C_P and C_F fields, essential for accurate force and moment predictions.

Figures 9.12 and 9.13, along with Table 9.4, examine the performance of the GCN-MM-AE and GB-AE-GCN models on the test sample with the highest prediction error from the GB-AE-GCN model. This sample offers insights into areas where complex aerodynamic features pose challenges even for the enhanced model.



(A) C_P contours for CFD reference (left), models prediction (mid), and error (right).



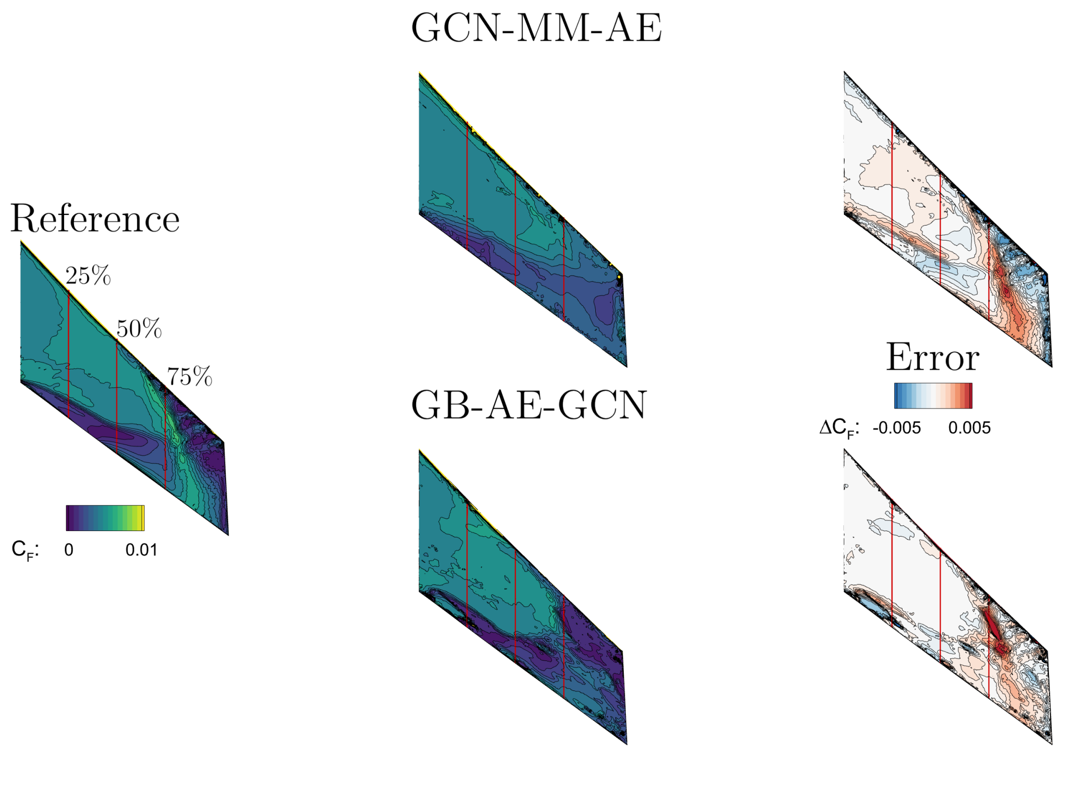
(B) C_P distributions at the prescribed cross sections. CFD reference (black square), GCN-MM-AE model prediction (black solid), GB-AE-GCN model prediction (red solid).

FIGURE 9.10: Comparison of prediction errors for C_P between the two modelling approaches on the test sample with the highest prediction error from the GCN-MM-AE model, $\boldsymbol{m} = [20.7, 57.9, 41.1, 56.1, 43.5, -47.7]$.

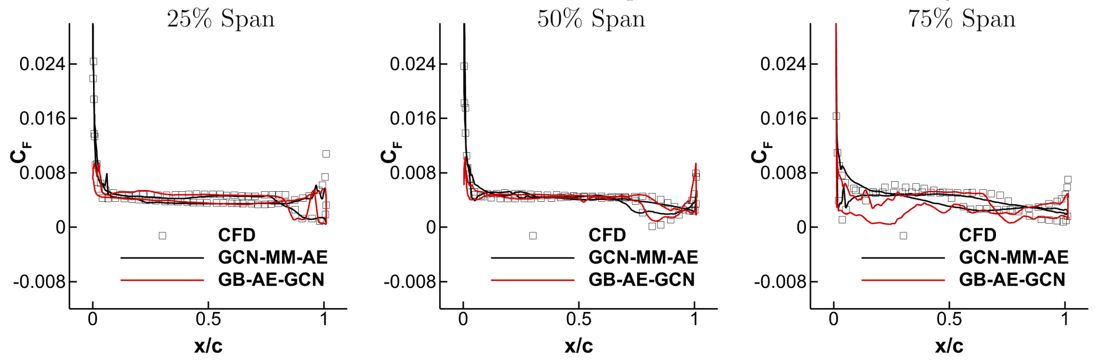
Model	ε_{C_L} [%]	ε_{C_D} [%]	ε_{C_M} [%]
GCN-MM-AE	-1.3219	0.3502	2.3498
GB-AE-GCN	0.4221	0.1171	-0.9414

TABLE 9.3: Comparison of prediction errors for C_L, C_D, and C_M between the two modelling approaches on the test sample with the highest prediction error from the GCN-MM-AE model, $\boldsymbol{m} = [20.7, 57.9, 41.1, 56.1, 43.5, -47.7]$.

In Figure 9.12, C_P fields for CFD reference, model predictions, and error distributions are displayed. The GB-AE-GCN model aligns more closely with the CFD solution than GCN-MM-AE, especially in lower-pressure zones, though both models exhibit minor deviations in high-gradient regions near the leading edge. The contour plots highlight



(A) C_F contours for CFD reference (left), models prediction (mid), and error (right).

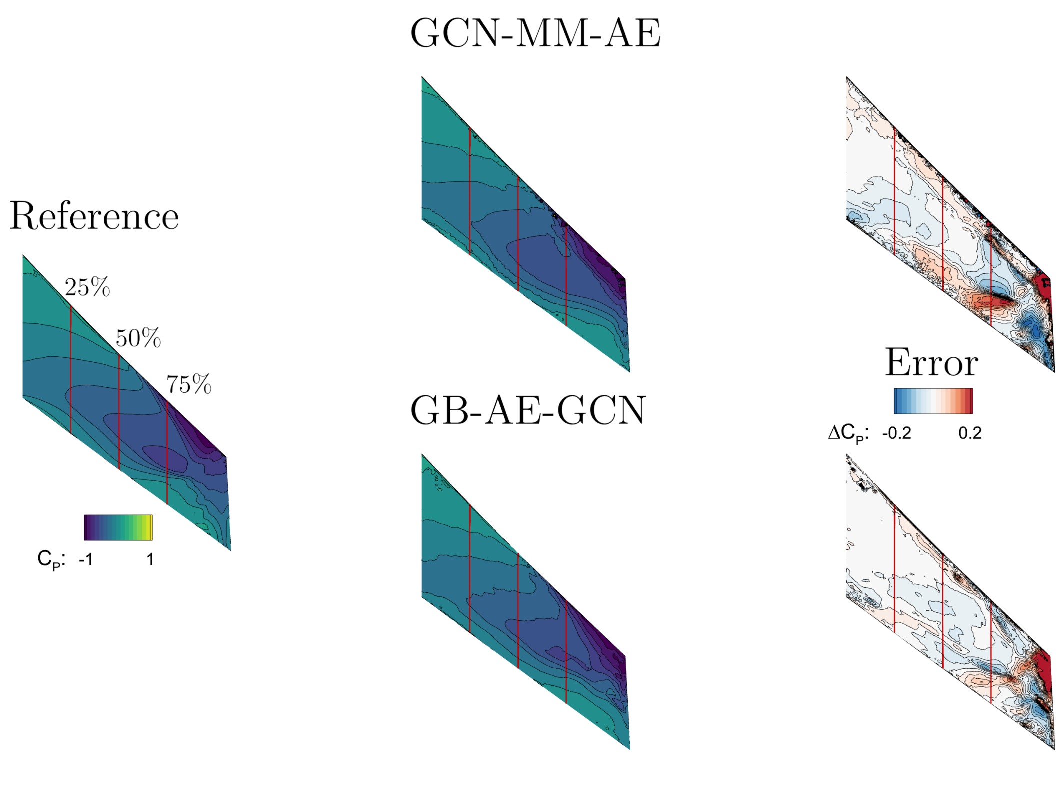


(B) C_F distributions at the prescribed cross sections. CFD reference (black square), GCN-MM-AE model prediction (black solid), GB-AE-GCN model prediction (red solid).

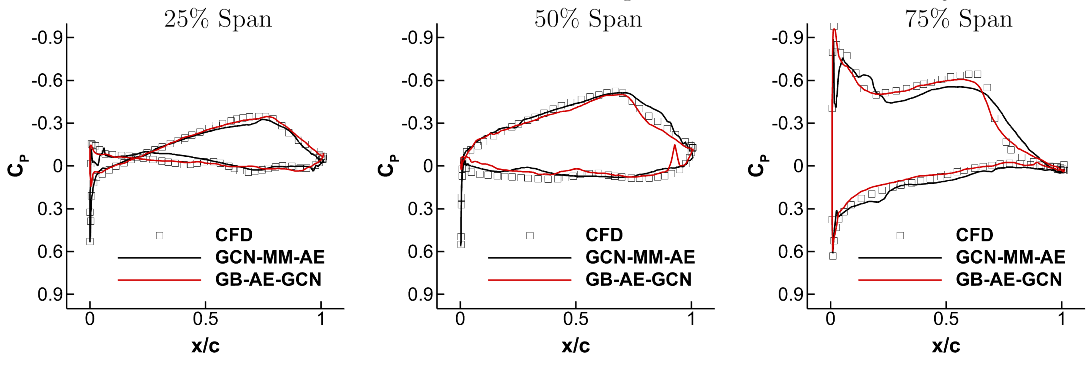
FIGURE 9.11: Comparison of prediction errors for C_F between the two modelling approaches on the test sample with the highest prediction error from the GCN-MM-AE model, $\boldsymbol{m} = [20.7, 57.9, 41.1, 56.1, 43.5, -47.7]$.

these areas, with the GB-AE-GCN model showing smaller errors overall. The cross-sectional C_P profiles confirm that GB-AE-GCN generally follows the CFD trend well, with minor variations in matching precise gradient changes. These discrepancies are physically consistent with the difficulty of reproducing shock-induced discontinuities in pressure distributions, where the flow undergoes rapid compression and local non-linearities dominate.

Figure 9.13 provides a similar comparison for C_F, where both models experience some difficulties capturing the exact distribution of surface shear stresses. The GB-AE-GCN model demonstrates improved accuracy over GCN-MM-AE, particularly in high-shear



(A) C_P contours for CFD reference (left), models prediction (mid), and error (right).

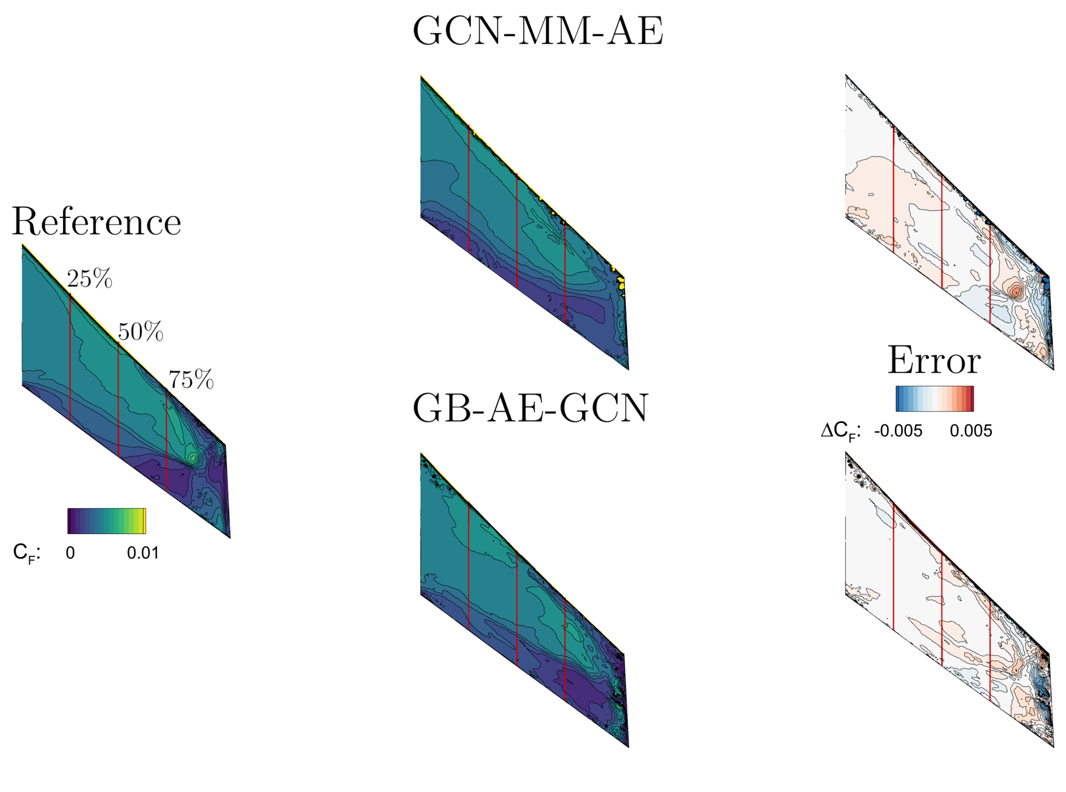


(B) C_P distributions at the prescribed cross sections. CFD reference (black square), GCN-MM-AE model prediction (black solid), GB-AE-GCN model prediction (red solid).

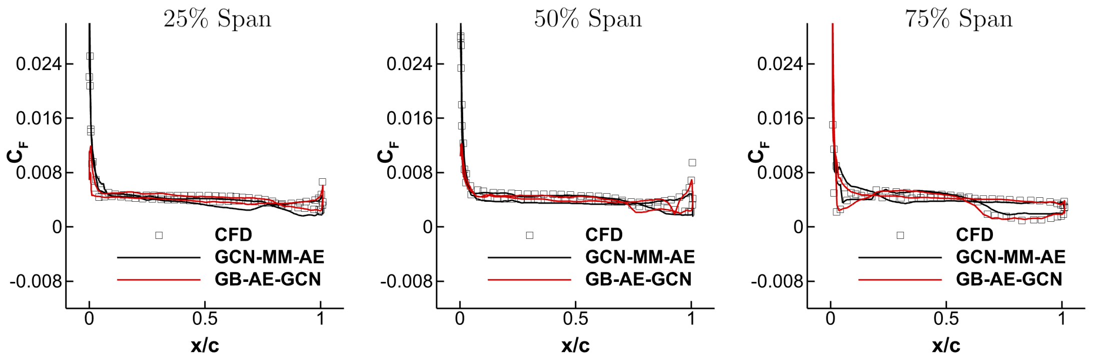
FIGURE 9.12: Comparison of prediction errors for C_P between the two modelling approaches on the test sample with the highest prediction error from the GB-AE-GCN model, $\boldsymbol{m} = [63.2, 64.3, 5.8, 43.9, -18.5, -48.8]$.

regions, though certain fine-scale details seen in the CFD reference remain difficult to capture. These regions correspond to boundary layer distortions induced by shock interactions, where shear layers are locally intensified. The inability to resolve such sharp gradients within a reduced latent space highlights the challenge of modelling the coupled shock-boundary layer system, in which small geometric or flow perturbations can significantly alter the local shear distribution. These observations indicate that while gradient-based space reduction enhances the GB-AE-GCN model predictive capability, capturing fine-scale variations in friction remains a challenge.

Table 9.4 summarizes prediction errors for C_L, C_D, and C_M for this challenging sample.



(A) C_F contours for CFD reference (left), models prediction (mid), and error (right).



(B) C_F distributions at the prescribed cross sections. CFD reference (black square), GCN-MM-AE model prediction (black solid), GB-AE-GCN model prediction (red solid).

FIGURE 9.13: Comparison of prediction errors for C_F between the two modelling approaches on the test sample with the highest prediction error from the GB-AE-GCN model, $\boldsymbol{m} = [63.2, 64.3, 5.8, 43.9, -18.5, -48.8]$.

The GCN-MM-AE model shows slightly higher accuracy in C_L and C_D, while the GB-AE-GCN model exhibits a notably lower error in C_M.

Model	ε_{C_L} [%]	ε_{C_D} [%]	ε_{C_M} [%]
GCN-MM-AE	-0.6172	-0.1049	1.265
GB-AE-GCN	0.6398	0.1723	-0.7695

TABLE 9.4: Comparison of prediction errors for C_L, C_D, and C_M between the two modelling approaches on the test sample with the highest prediction error from the GB-AE-GCN model, $\boldsymbol{m} = [63.2, 64.3, 5.8, 43.9, -18.5, -48.8]$.

To further analyse the performance of both approaches when predicting C_P and C_F values, three different error metrics are presented: `MAPE`, `RMSE` and `R2`. The GB-AE-GCN model achieves a 36% reduction in `MAPE` for C_P and a 31% reduction for C_F compared to the GCN-MM-AE model. Examining `RMSE` values, GB-AE-GCN achieves a 24% reduction for C_P and a 20% reduction for C_F. In terms of `R2`, the GB-AE-GCN model attains higher values, achieving 0.9734 for C_P and 0.9627 for C_F. These improvements indicate that GB-AE-GCN captures surface pressure and skin friction variations more accurately, demonstrating a stronger capability to model physical phenomena.

Model	**MAPE**		**RMSE**		**R2**	
	C_P	C_F	C_P	C_F	C_P	C_F
GCN-MM-AE	3.6681	9.6997	0.0831	0.2070	0.9680	0.9576
GB-AE-GCN	2.3479	6.6808	0.0631	0.1654	0.9734	0.9627

TABLE 9.5: Comparison of C_P and C_F prediction accuracy metrics on the test set for the two modelling approaches.

9.5 Computational Cost Analysis

The computing cost analysis reported in Table 4.6 compares the computational efficiency of the two proposed GDL frameworks against traditional CFD simulations. The analysis shows a significant reduction in computational resources when using the GCN-MM-AE and GB-AE-GCN frameworks. While CFD simulations require substantial CPU hours for generating one sample, the ROMs leverage GPU-based processing, drastically reducing the time required for model training and prediction.

The GCN-MM-AE model, despite being simpler in terms of architecture, requires a longer training time due to its larger number of weights and less efficient structure. In contrast, the GB-AE-GCN model benefits from architectural refinements, which significantly reduces the number of trainable parameters and computational time. As a result, the GB-AE-GCN approach achieves faster training and prediction times, with a reduction of nearly 50% in training hours and 78% in per-sample prediction time compared to the GCN-MM-AE framework.

These results make the ROM approaches particularly attractive for real-time aerodynamic shape optimisation, enabling faster exploration of design spaces without significantly compromising accuracy.

It is also important to note that the computational cost for generating the dataset represents a significant portion of the overall time investment. This step remains a critical bottleneck in developing ROMs and highlights the need for efficient dataset generation strategies when considering the scalability of such frameworks for large-scale design optimisation tasks.

Approach	CFD (CPU hours)		ROM (GPU hours)		
	Simulation (251 runs)	(1 run)	optimisation (30 trials)	Training (1 model)	Prediction (1 sample)
1. GCN-MM-AE	39,000	155	–	5.0	10.0×10^{-5} ($\sim 0.4s$)
2. GB-AE-GCN	=	=	45	2.5	2.2×10^{-5} ($\sim 0.1s$)

TABLE 9.6: Computing cost comparison between CFD and the two proposed ROM approaches.

9.6 Conclusions

This chapter presented and compared two geometric deep learning architectures, GCN-MM-AE and GB-AE-GCN, for predicting aerodynamic surface fields under varying wing geometries. Both models were tested on the Agard 446.5 wing test case, highlighting their potential to drastically reduce computational costs compared to high-fidelity CFD while maintaining a robust level of prediction accuracy.

Initially, the GCN-MM-AE model validated the potential of GDL for aerodynamic prediction over different geometries, but it also revealed areas for improvement, particularly in streamlining parametrisations and optimizing computational overhead. Building on these insights, the GB-AE-GCN model introduced architectural refinements that reduced the number of trainable parameters, shortened training times, and improved prediction quality. This evolution highlights the trade-off between model complexity and efficiency, and underscores that the simpler yet more efficient GB-AE-GCN method achieves superior performance for predicting distributed aerodynamic fields.

In terms of quantitative accuracy, both models showed promising results for C_L and C_D, although discrepancies were noted for C_M. In addition, both frameworks successfully captured detailed surface flow patterns, demonstrating significant potential for aerodynamic shape optimisation where fast and accurate flow predictions can reduce design iteration costs.

From a broader perspective, these findings reinforce the goals of the thesis by:

1. **Rapid Aerodynamic Predictions for Complex Shapes**
   Both GDL approaches demonstrate the ability to handle nontrivial shape variations efficiently, enabling accurate aerodynamic predictions across a wide range of deformations. This supports Research Objective O5, which aims to develop surrogate models capable of generalising across shape variations for use in shape optimisation, parametric studies, or rapid aerodynamic evaluation of novel configurations.

2. **Balancing Complexity and Efficiency**
   The improvement from GCN-MM-AE to GB-AE-GCN showcases how iteratively

refining network architectures can achieve higher accuracy and lower computational costs, aligning with the thesis ambition for scalable, data-driven aerodynamic solutions.

3. **Integration with Multidisciplinary Tasks**
   Using these surrogates in shape optimisation loops provides a path to significantly faster design convergence. Instead of repeatedly running full-order CFD or adjoint solvers to compute both loads and gradient-based sensitivities, a graph-based surrogate could in principle predict the aerodynamic fields or act as a substitute for the adjoint-based partial derivatives. However, it must be noted that the prediction error exhibited by the model may still be too high for direct use in the final stages of gradient-based optimisation. These phases, which rely heavily on accurate sensitivities to converge towards a local optimum, require highly precise estimates of aerodynamic loads and their derivatives. Therefore, further refinement with hybrid strategies, such as data fusion that combines model predictions with targeted CFD runs, may be necessary to achieve the required level of accuracy.

Despite the encouraging results, the models developed in this chapter exhibit a loss of performance compared to the steady and unsteady fixed-geometry cases of Chapters 7 and 8. This degradation arises from the increased complexity of simultaneously learning both aerodynamic mappings and geometric variations. While in fixed-geometry problems the graph structure remains constant, here each structural deformation alters node coordinates and local topology, requiring the model to adapt to changing mesh representations. Furthermore, the hierarchical autoencoder compresses the mesh to achieve efficiency, inevitably discarding fine-scale features that become critical when geometry itself varies. These limitations are compounded by the strong sensitivity of global loads, particularly the pitching moment, to small discrepancies in pressure distributions around the leading edge. Finally, the dataset used in this study covers only a limited set of modal deformations, restricting the ability of the model to generalise to unseen or extreme shapes. These factors highlight the trade-off between accuracy and scalability in geometry-aware surrogates.

Further research will focus on refining the training datasets and testing different applications to improve the representativeness and coverage of the training data. Because the predictive accuracy of the framework is highly dependent on the quality and completeness of the datasets used, without careful dataset selection, including also hybrid strategies (e.g., data fusion with targeted CFD runs) to correct local discrepancies, the model performance could degrade when applied to configurations or flow conditions that differ significantly from the training set, limiting reliable extrapolation. In addition, the current framework lacks an integrated method for uncertainty quantification, making its predictions inadequate for rigorous and safety-critical aircraft design evaluations. Future efforts will aim to integrate probabilistic techniques, such as Bayesian graph neural networks or ensembles of graph-convolutional autoencoders, to provide inherent confidence intervals for predictions. Such advances will position these models as powerful surrogate tools, enabling more accurate and robust exploration of the design space and deeper integration into modern, computationally demanding workflows.

Statement of Contributions

David Massegur developed the presented Approach 1: GCN-MM-AE.

Chapter 10

Conclusions and Future Work

This concluding chapter summarises the main findings and contributions of this research, emphasising the advancements in efficient aerodynamic modelling and reduced-order methodologies. The discussion revisits key insights gained from the application of ML techniques to nonlinear aerodynamics, highlighting both theoretical and practical implications. Additionally, the challenges encountered throughout the study are critically assessed, providing a comprehensive perspective on the limitations and potential refinements of the proposed methodologies. Finally, the chapter explores promising avenues for further research. Together, these considerations offer a concise summary of the research achievements and a prospective outlook on the role of ML-based surrogates in shaping the future of aerodynamic studies and beyond.

10.1 Significance and Contributions of the Research

This thesis systematically explored ML-based ROM strategies for aerodynamic applications, addressing the computational challenges of high-fidelity CFD simulations in both steady and unsteady regimes. Each chapter incrementally addressed methodological limitations, gradually refining capabilities for reduced-order modelling, uncertainty quantification, and modal space analysis. The investigations demonstrate that ML can significantly reduce computational costs and, in many cases, improve prediction accuracy compared to traditional approaches, especially in nonlinear conditions. However, practical effectiveness depends on careful selection of architectures, smart choice of training data, and incorporation of robust uncertainty quantification.

A key contribution of this research lies in demonstrating that ML-based ROMs can rival, and in certain cases outperform, classical projection-based techniques such as POD and DMD. Unlike traditional ROMs that rely on linear modal decompositions and often require explicit stabilisation and interpolation schemes, the ML-based frameworks developed herein offer broader applicability and better generalisation capabilities. These advantages are particularly apparent in high-dimensional or unsteady problems, where conventional ROMs suffer from sensitivity to mode truncation and limited extrapolation performance. Moreover, ML architectures eliminate the need for modal calibration, which simplifies deployment across varying flow conditions. Nonetheless, this flexibility comes with trade-offs. ML models act as black boxes, optimising weights to minimise predictive error without directly encoding physical mechanisms, thereby reducing interpretability. Additionally, while POD and DMD can be deployed on CPU-only platforms and modest hardware, the training and inference of ML-ROMs typically require GPU acceleration. While this may initially limit their practical integration, hardware advancements are rapidly narrowing this gap.

A core finding is that the choice of ROM architecture should be tailored to the specific problem. Chapter 3 illustrates how simpler models can accurately handle shock-induced transonic flow separation and flutter onset in 2D settings when predicting global coefficients, while Chapter 4 shows that more advanced architectures are necessary when extending to 3D configurations, where the underlying flow features are inherently more complex. However, once the objective shifts from global coefficients to modelling vector fields over wing surfaces with thousands of degrees of freedom, autoencoder-based techniques become essential to reduce data complexity and address the curse of dimensionality. Chapters 7, 8, and 9 demonstrate the ability of autoencoders to compress data into latent representations, without compromising the essential flow physics, which is a key to the predictive performance of ROMs in steady and unsteady transonic conditions. The research also emphasised the value of systematic hyperparameter optimisation, not only to enhance prediction accuracy but also to reduce inference time.

In aeroelastic scenarios, where the interaction between the structural and aerodynamic domains can rapidly amplify small inaccuracies in aerodynamic predictions, model robustness and data quality become extremely important. Robustness was found to depend heavily on the convergence quality of the CFD snapshots and the distribution of the training data. Including non-converged CFD samples in the training set led to higher prediction errors and resulted in overly complex network architectures after hyperparameter optimisation. However, it is worth noting that in regimes where traditional CFD solvers fail to converge, ML models can still offer regularised approximations, providing early estimates of flow conditions and structural loads that support preliminary assessments. In addition, the spatial distribution of training samples across the design space strongly influences performance: well-placed samples allow for accurate modelling with fewer simulations. These findings highlight the importance of data quality and coverage, especially in industrial applications where generating new CFD data is time-consuming and costly. Future efforts should prioritise reducing the number of required CFD evaluations through adaptive sampling and active learning. In this context, approaches such as data fusion and incremental network calibration hold promise for enabling efficient generalisation to new design spaces without the need for full retraining.

One of the major contributions is the introduction of geometric deep-learning techniques for aerodynamic flow prediction on non-homogeneous unstructured meshes. In computer vision, CNNs have proven effective largely because they operate directly on raw image data with minimal pre-processing, thereby preserving essential information, and with the task of abstraction and prediction fully assigned to the network itself, they resulted in powerful feature extraction and generalisation capabilities. Motivated by this principle, the research extended the convolutional paradigm to unstructured aerodynamic datasets by replacing the regular grid architecture of CNNs with graph-based representations. This led to the adoption of GCNs, which exploited mesh connectivity and geometric relationships to learn directly from flowfields defined on arbitrary meshes. Chapter 7 presented a graph-convolutional autoencoder that captures spatial relationships in complex aerodynamic configurations, and Chapter 8 extended this methodology by incorporating temporal layers, enabling accurate forecasts of unsteady transonic loads on 3D geometries. Chapter 9 further generalised the approach to accommodate varying structural shapes, demonstrating that GCNs could maintain predictive accuracy even in the presence of geometric variation. Through these interconnected studies, the thesis demonstrated that GCN architectures can scale effectively to large aerodynamic datasets, providing a good balance between computational efficiency and predictive fidelity.

Although GCNs were selected for their conceptual similarity to CNNs and their natural compatibility with graph-structured data, several alternative approaches merit consideration. Implicit Neural Representations (INRs), for example, enable discretisation-invariant predictions by modelling physical quantities as continuous functions over spatial coordinates. As shown by Catalani et al. [56], INRs are capable of accurately capturing nonlinear flow features through zero-shot super-resolution. Nonetheless, they are typically associated with greater model complexity, longer training times, and increased sensitivity to the choice of spatial encoding. In parallel, recent developments such as DeepGeo [369] have introduced geometry-aware graph networks that focus on shape parametrisation by learning mesh embeddings, replacing conventional techniques such as free-form deformation (FFD). While DeepGeo is currently limited to geometric encoding, its underlying principles suggest a promising direction for future research: the integration of shape parametrisation and aerodynamic prediction within a unified, learnable framework.

Uncertainty quantification remains a critical consideration, particularly in transonic flows where shock-boundary layer interactions can cause highly sensitive aerodynamic responses. Although the models are trained within a predefined design space, their predictions in sparsely sampled or extrapolated regions may suffer from unknown errors. This sensitivity is investigated in Chapter 5, which developed a multi-fidelity Bayesian neural network that leverages heterogeneous datasets to capture both epistemic and aleatory uncertainties. In practice, Bayesian inference across fidelity levels yields probabilistic predictions with credible intervals that are especially valuable in conceptual design, model validation, and robust optimisation. By examining the confidence intervals around predicted quantities, it is possible to assess the reliability of these estimates and identify regions of the flight envelope that require further refinement. In addition, these uncertainty bounds play an important role in robust optimisation, where they help avoid solutions that appear optimal under a single deterministic prediction, but carry a high risk of underperformance when potential variations and modelling inaccuracies are considered.

At the beginning of the research, the exclusive use of large high-fidelity CFD datasets appeared to ensure accuracy, yet it quickly proved impractical for broader aerospace applications due to computational cost and convergence difficulties, while alternative high-fidelity sources (e.g. wind tunnels or flight tests) are even more resource-intensive. These constraints motivate strategies that combine fidelities to maximise information while minimising reliance on costly evaluations: consistent data fusion integrates high- and low-fidelity sources; data augmentation enhances coverage when low-fidelity data do not adequately span the design space; and, more importantly, the resulting uncertainty estimates, particularly confidence bands, become central to guiding new data acquisition. By highlighting where the surrogate is least certain, uncertainty can turn data scarcity into a targeted sampling task. For instance, in an active-learning loop, acquisition rules (e.g. predictive variance or expected improvement) propose candidate points, of which only a selected subset is promoted to new high-fidelity runs, optionally complemented by cheaper low-fidelity samples. In parallel, adaptive design of experiments concentrates sampling near physically significant regions (e.g. shock movement, buffet onset, stall) while keeping more benign portions of the envelope less densely covered. Together, these methods can transform data generation from brute-force accumulation of high-fidelity samples into a selective, uncertainty-aware process that balances cost, reliability, and physical relevance.

In terms of practical utility and industrial applicability, the developed ML-ROMs can offer advantages for early-stage design, control, and digital twin applications. Their combination of computational speed and predictive accuracy makes them well-suited for rapid design space exploration and the identification of promising regions for optimisation. Within simulation environments, they serve as lightweight surrogates, enabling interactive assessment of aerodynamic behaviour. Moreover, their capacity to capture unsteady flow dynamics with high temporal resolution allows for integration into flight control systems. Real-time estimation of aerodynamic loads from surface pressure perturbations can facilitate onboard prediction of nonlinear instabilities, such as LCO, thereby informing adaptive control strategies, flutter margin tracking, and structural health monitoring.

Beyond accuracy and application relevance, it is important to estimate the break-even point N, defined as the minimum number of predictions beyond which ML-ROMs become more efficient than repeated high-fidelity CFD runs. This threshold is formulated as:

$$T_{\text{CFD}} \times N = T_{\text{CFD,train}} + T_{\text{ROM,train}} + T_{\text{ROM,opt}} + T_{\text{ROM,inf}} \times N$$

Here, T_{CFD} is the cost of one CFD run, $T_{\text{CFD,train}}$ is the total cost of generating training data, $T_{\text{ROM,train}}$ is the ROM training time, $T_{\text{ROM,opt}}$ is the time for hyperparameter optimisation, and $T_{\text{ROM,inf}}$ is the inference time per ROM evaluation. The values reported below are order-of-magnitude estimates and are strictly valid under the conditions of this study: (i) CFD costs measured on an Intel Skylake HPC (3 nodes × 40 cores) with RANS/URANS set-ups and meshes detailed in the relevant chapters; (ii) ROM training and inference on a workstation with an Intel XEON W-2255 CPU and NVIDIA RTX A4000 GPU; (iii) hyperparameter optimisation budgets where indicated; and (iv) "one prediction" meaning a single steady evaluation or, for unsteady fields, one full signal as specified. Under these assumptions, the break-even thresholds obtained from the measured times are approximately:

- **Unsteady global coefficients**: $N \approx 140$. Derived from the cost figures in Chapter 4 (e.g. $\sim 4{,}000$ CPU-h per URANS run; ROM optimisation/training $\ll 1$ GPU-h; ~ 1 s per evaluation).
- **Steady global coefficients with UQ**: $N \approx 58$. Based on dataset generation costs (500 CPU-h per run) and ROM training/inference times (≈ 1 GPU-h training; $\sim 10^{-5}$ GPU-h per prediction), see Chapter 5.
- **Steady vector field prediction**: $N \approx 70$. Using per-run CFD costs of ~ 500 CPU-h and per-sample ROM inference ~ 1 s, with training budgets reported in Chapter 7.
- **Unsteady vector field reconstruction**: $N \approx 12$ per full signal of $\sim 10^3$ timesteps, reflecting per-timestep inference ~ 0.15 s (full signal < 2 min) versus $\sim 6{,}000$ CPU-h per URANS run, see Chapter 8.
- **Generalisation across geometries**: $N \approx 250$, using the Chapter 9 training budgets (~ 45 GPU-h), with per-sample inference ~ 0.1 s, and CFD campaign costs (~ 155 CPU-h per-run).

These thresholds are problem-dependent and will vary with solver settings (e.g. turbulence model, mesh resolution, time step), HPC hardware, dataset size, hyperparameter budget, and whether multi-fidelity data are used. They should therefore be interpreted as contextual indicators for the specific test cases and workflows documented in Chapters 3–9, not as universal constants. Nonetheless, under the above conditions they indicate that ML-ROMs can deliver substantial speed-ups, particularly in iterative design, uncertainty quantification-, and control loops where many evaluations are required.

In summary, this thesis demonstrates that ML can serve as a valuable complement to high-fidelity simulation, bridging the gap between efficiency and accuracy. The studies show that, under carefully controlled data conditions, neural networks can compress complex flowfields into low-dimensional latent spaces, accelerate modal space exploration through fast surrogate evaluation, and explicitly represent uncertainty. An important limitation, however, is the reliance on extensive, high-quality training data, which are expensive to generate and limits the range of conditions under which these models can confidently operate. Nonetheless, the research makes significant progress in overcoming critical key computational bottlenecks in CFD-based design. By tailoring ROM architectures to problem-specific requirements, whether scalar aerodynamic loads or high-dimensional unsteady flowfields, this work establishes a solid foundation for future academic research and industrial implementation.

10.2 Research Limitations

Although the methods presented in this thesis have shown promise in predicting complex, nonlinear aerodynamic phenomena, several limitations reduce their broader applicability and point to opportunities for future improvement.

First, while the study spans a relevant range of Mach numbers and angles of attack, the transonic regimes considered do not cover the full spectrum of off-design or extreme conditions encountered in operational settings. Scenarios involving high angles of attack, large-scale flow separation, or strong compressibility effects with vortex shedding remain outside the current training domain. Consequently, the predictive performance of the proposed ROMs may degrade when extrapolated to flight envelopes far from the sampled conditions. In addition, although some test cases include 3D configurations and unstructured meshes, the diversity of geometries remains limited. Most models are trained on variations of baseline wing shapes with relatively simple planform changes. The ROMs are thus constrained to the input conditions and geometries seen during training. Expecting a single model to generalize across fundamentally different geometries is extremely challenging, not only due to the complex aerodynamic differences involved, but also because of the preprocessing steps required to render such data compatible in terms of mesh structure, resolution, and topology. Geometric generalisation should therefore be approached carefully, either through specialised models trained on specific families of shapes or through the development of more flexible representations, such as learned embeddings.

A further limitation is the dependency on high-fidelity simulation data for both training and validation, whether in the form of flowfield snapshots, indicial responses, or multi-fidelity datasets. While reduced sampling strategies help mitigate costs, each new flight condition or configuration requires additional CFD computations, and extrapolations beyond the sampled parameter space carry no formal guarantees of accuracy. Strong nonlinearities, especially near shock formation, also make these models sensitive to the choice of training data. Although ensemble and multi-fidelity approaches can partially address this, the precise thresholds beyond which ROMs become unreliable remain an open question.

The generalisability of the proposed ROMs beyond simplified fluid-structure coupling is also limited. Although 2D and 3D geometries with simplified structural dynamics have been studied, truly complex systems involving large structural deflections, composites, or active control surfaces remain largely unexplored. Modelling these effects would require additional high-fidelity aeroelastic data and more advanced coupling frameworks, which in turn would increase the same cost and resource constraints that originally motivated ROM development. Thus, the trade-off between the complexity of structural modelling and the computational advantages of faster aerodynamic surrogates must be carefully evaluated.

Several limitations emerge from the ML frameworks themselves. Despite their high accuracy, deep learning architectures lack explicit physical interpretability. Their latent variables do not correspond to identifiable flow modes or structures, making them unsuitable for mechanistic analysis or interpretability-driven certification pathways. This lack of transparency limits their application in contexts where explainability and traceability are essential.

Uncertainty quantification methods also introduce interpretability challenges. In particular, the distinction between aleatoric and epistemic sources of uncertainty remains difficult in practice, and calibration of confidence intervals is often limited by the availability of ground truth data. This hinders the integration of ROMs into safety-critical workflows, where formal and interpretable uncertainty estimates are required.

Scalability to large-scale, industrial meshes presents an additional constraint. While graph-based models naturally accommodate unstructured grids, their application to meshes comprising several hundred thousand nodes remains computationally demanding. Training on such graphs requires substantial memory and processing resources, particularly during backpropagation. Without the use of mesh coarsening techniques or hierarchical learning strategies, their use in full-scale industrial settings may remain constrained.

Despite the efficiency of ROMs during the online phase, their high offline training cost remains a major limitation. This restricts their suitability for time-sensitive scenarios or applications requiring only a limited number of evaluations. Moreover, accurately capturing high-frequency unsteady flow phenomena often necessitates small timesteps, which further increases the offline cost. Numerical stability also poses a concern, especially in coupled aeroelastic systems, where small errors in aerodynamic load predictions can propagate through the structural solver and degrade the overall reliability of the simulation.

Therefore, the work in this thesis must be viewed in light of limitations related to flow regime coverage, reliance on high-fidelity data, limited scope in geometric and structural modelling, numerical coupling complications, interpretability of deep learning models, challenges in uncertainty quantification, and computational scalability. Future research could address these issues by expanding training datasets, integrating advanced turbulence models, exploring more complex geometries, and developing stronger multi-fidelity or uncertainty quantification frameworks. By systematically extending the scope of each ROM component, the proposed methods could mature into more robust and widely applicable tools for high-speed aircraft design and analysis.

10.3 Further Research

The investigations presented in Chapters 3–9 demonstrated the ability of data-driven methods to capture complex transonic aerodynamic phenomena. Building on these results, several opportunities exist to further enhance model accuracy, generalisability, and computational efficiency. The following outlines major directions for future research.

- **Wider Flow Regimes and Nonlinear Flow Physics.** Many of the current test cases have investigated moderate to high Mach numbers under limited ranges of angles of attack. It would be beneficial to include extreme regimes where stronger shock-induced separation, dynamic stall and buffet dominate. Such expanded coverage of the flight envelope would provide the data-driven surrogate models with more challenging, nonlinear phenomena from which to learn, further improving their generalisation and robustness.
- **Refined Aeroelastic and Fluid-Structure Coupling.** Chapters 3 and 4 demonstrated the feasibility of embedding ROMs in aeroelastic solvers to predict flutter boundaries and transient structural responses. Looking ahead, the incorporation of more advanced structural models, including large deflections, flexible couplings, and nonlinear material behaviour, would significantly improve the accuracy of these methods. Future studies could also explore coupling ROM-based

aerodynamics with detailed finite element or multi-body solvers for realistic configurations such as connected wings or fuselage-mounted control surfaces. This will require extending the parametrisation of the surrogate models to account for time-varying structural dynamics beyond small perturbations.

- **Enhanced Parametric Space Exploration and Global Optimisation.** Chapter 9 underscored the importance of handling large sets of shape variations efficiently, employing graph-convolutional autoencoders. A logical next step is to investigate how these surrogates perform in global optimisation loops or multi-objective scenarios (e.g., minimizing drag and maximizing lift-to-drag ratio under structural constraints). Methods such as Bayesian optimisation and evolutionary algorithms could be combined with the ROM predictions for accelerated design space exploration, ultimately guiding the design to highly nonlinear optimal shapes with fewer high-fidelity CFD runs.

- **Extended Uncertainty Quantification and Multi-Fidelity Data Fusion.** Chapter 5 highlighted a multi-fidelity Bayesian neural network approach to account for uncertainties in transonic aerodynamic loads. Extensions could include:

  - Incorporating physics-based constraints (e.g., shock location bounds, positivity of turbulent viscosity) into the multi-fidelity frameworks for more physically interpretable confidence intervals.
  - Adopting adaptive sampling or active learning strategies that pinpoint where additional high-fidelity runs can most effectively reduce model uncertainty.
  - Evaluate the hybrid integration of multiple lower-fidelity levels in geometric deep learning, such as panel methods or coarser RANS simulations, with high-fidelity LES data or wind tunnel measurements to continuously refine a GCN-based surrogate model over time.

  These enhancements would be particularly relevant for probabilistic load prediction and robust structural design, where explicit confidence measures are of critical importance.

- **Deeper Integration of Graph-Based Models for Complex Geometries.** Throughout Chapters 7–9, GNNs excelled at handling unstructured grids and complex wing or wing–fuselage shapes. This capacity sets the stage for dealing with more realistic, industry-scale meshes. Additionally, combining these GNN-based surrogates with geometric parametrisation tools like free-form deformation or CAD-based morphing would streamline shape optimisation tasks that rely on flexible grid manipulations.

- **Leveraging Generative Foundation Models for Sequence Prediction.** Transformer architectures that excel in sequence modelling can be trained on sequences of mesh fields (or latent graph embeddings) to predict the evolution of vector fields over thousands of timesteps without drift. Their attention mechanism identifies key flow features, such as shocks or separations, and classifies those regions as “tokens of interest”, enabling both accurate forecasting and improved physical interpretability.

- **Self-supervised and Contrastive Pre-training to Reduce Data Costs.** Even with the data-efficient methodology proposed here, high-quality labelled CFD snapshots remain expensive. Self-supervised objectives (e.g., masked-node reconstruction on unstructured graphs) can pre-train encoders on plentiful unlabelled coarse-grid solutions before fine-tuning on scarce high-fidelity data. Contrastive learning between low- and high-fidelity representations can further align latent spaces, reducing the number of expensive RANS/LES runs required for a given accuracy target.

- **Automated Data Augmentation for Rare Flow Events.** Synthetic data can be efficiently generated by applying perturbations such as randomised shock positions, spanwise phase shifts or local mesh morphing to low-fidelity solutions. Using style-transfer networks, these can then be remapped into a more realistic, high-fidelity representation, thereby populating the data space with rare events (e.g., deep stall or buffet) that would otherwise be difficult to obtain.

- **Fast and Scalable Predictions on Exascale Hardware.** Upcoming GPU-dense and mixed-precision architectures (e.g., NVIDIA Grace-Hopper) will enable large-scale models with a capacity of billions of parameters to be trained in hours rather than weeks. Combined with techniques such as dynamic mesh coarsening and implicit graph operators, these models can process full aircraft configurations with tens of millions of cells in near real-time. This opens the door to high-speed applications like aeroelastic mode tracking and onboard flowfield prediction.

These advances will reduce the computational cost of high-fidelity simulations by orders of magnitude and enable more insightful, pattern-recognition analyses. This will allow engineers to identify changes in flow topology, early signs of aeroelastic instabilities, and rare but critical events well before traditional solvers can compute a single solution.

Appendix A

Models Architecture

This appendix presents the architectures of the ML models used for each test case in the thesis. It provides a detailed breakdown of the network structures, hyperparameter configurations, and optimisation strategies tailored to specific aerodynamic applications. The reported models span regression, fully-connected, and graph-based neural networks, including multi-fidelity approaches for uncertainty quantification. These architectures were designed to balance computational efficiency and predictive accuracy, supporting the CFD analyses conducted in this work.

A.1 Data-Driven Aerodynamic Modelling of 2D Unsteady Transonic Loads

Table A.1 presents the hyperparameter configurations for the ML model used to predict aerodynamic loads. The regression model is designed to provide a computationally efficient baseline by using a single fully-connected output layer. The learning rate and batch size were optimised to balance convergence speed and numerical stability. Due to the nature of linear regression, no hidden layers or nonlinear activation functions are employed.

Hyperparameter	**Value**
Learning rate	$3.0 \cdot 10^{-4}$
Hidden layers	None
Neurons per layer	Single output layer
Batch size	300
Activation function	Linear
Loss function	MSE
Optimizer	Adam

TABLE A.1: Hyperparameter configurations for the implemented model.

A.2 Parametric Nonlinear Volterra Series via Machine Learning for 3D Unsteady Transonic Loads

Table A.2 presents the final hyperparameter configurations after the Bayesian optimisation process for the FCNN model used to predict linear and nonlinear Volterra kernels across the three test cases. The architectures vary in terms of learning rate, number of hidden layers, and neurons per layer and reflect the increasing complexity of the datasets from the synthetic model to the 3D wing case. Nonlinear kernels generally require deeper architectures and slightly lower learning rates compared to linear kernels, as they involve modelling more complex relationships. The activation function, PReLU, was chosen consistently across all models due to its effectiveness in handling nonlinearity.

Test Case	**Hyperparameter**	**Linear Kernels**	**Nonlinear Kernels**
Synthetic Model	Learning rate	$2.5 \cdot 10^{-5}$	$9.0 \cdot 10^{-5}$
	Hidden layers	3	4
	Neurons per layer	[148, 108, 76]	[140, 116, 76, 164]
	Batch size	4	3
	Activation function	PReLU	PReLU
2D Aerofoil	Learning rate	$6.0 \cdot 10^{-5}$	$3.0 \cdot 10^{-5}$
	Hidden layers	4	5
	Neurons per layer	[172, 68, 116, 84]	[92, 124, 60, 108, 76]
	Batch size	4	4
	Activation function	PReLU	PReLU
3D Wing	Learning rate	$8.5 \cdot 10^{-5}$	$1.0 \cdot 10^{-5}$
	Hidden layers	5	6
	Neurons per layer	[164, 140, 92, 76, 204]	[196, 140, 164, 156, 92, 100]
	Batch size	4	4
	Activation function	PReLU	PReLU

TABLE A.2: Final hyperparameter configurations of the FCNN models for linear and nonlinear kernels prediction for the three test cases.

A.3 Multi-Fidelity Bayesian Network for Uncertainty Quantification in Aerodynamic Loads

A.3.1 Finite Wing Test Case

This section outlines the architecture used for the test case after the Bayesian hyperparameter optimisation. The design parameters for the optimisation are chosen accordingly to the multi-fidelity framework, and are described in Section 5.2.2. The final model is composed by 114,194 trainable weights for the pre-trained model with LF samples, 57,026 trainable weights for the pre-trained model with the first TL phase using MF samples, and 354 trainable weights during the last TL phase using HF samples.

For the first TL phase with MF samples, the model was trained using a batch size of 32 and a learning rate of 0.001. The maximum number of epochs was set to 8000, setting an early stopping criterion with a patience of 500. During the final TL phase with HF samples, the batch size was reduced to 1 to better capture high-fidelity details. This training followed a similar structure, with a maximum of 8000 epochs and an early stopping patience of 500. However, only one layer was unfrozen for fine-tuning, ensuring that the model retained the knowledge learned in previous stages while adapting to the HF dataset. The optimised hyperparameters for the four models developed are summarized in Table A.3.

Model	Hyperparameter	Best value
LF BNN	N_{layers}	6
	N_{units}	[176, 176, 144, 176, 176] + [2] output
	lr	0.0016
	μ_{prior}	0
	σ_{prior}	0.0126
MF BNN	N_{layers}	3
	N_{units}	[160, 80] + [2] output
	lr	0.0014
	μ_{prior}	0
	σ_{prior}	0.0526
HF BNN	N_{layers}	4
	N_{units}	[128, 160, 176] + [2] output
	lr	0.0014
	μ_{prior}	0
	σ_{prior}	0.0596
MF-Baynet	N_{layers}	6
	N_{units}	[176, 176, 144, 176, 176] + [2] output
	$lr(LF)$	0.0016
	$lr(LF \rightarrow MF)$	0.001
	$lr(MF \rightarrow HF)$	0.001
	μ_{prior}	0
	σ_{prior}	0.0126
	N_{frz} (LF → MF)	3
	N_{frz} (MF → HF)	5

TABLE A.3: Hyperparameters of the Low-Fidelity (LF), Mid-Fidelity (MF), High-Fidelity (HF) BNN models, and the Transfer Learning (TL) model.

A.3.2 Full-configuration Test Case

This section outlines the architecture used for the full-configuration test case after the Bayesian hyperparameter optimisation. The overall procedure closely follows what was done for the finite wing test case described in A.3.1, with the main difference being the larger number of layers and trainable weights required to capture the increased complexity of the full-configuration problem.

As summarized in Table A.4, the final network architecture for the full-configuration test case consists of six layers, for a total of 174,932 trainable weights. The model is first trained on the LF dataset using a batch size of 32, a learning rate of 0.0016, a maximum of 5000 epochs, and an early stopping criterion with a patience of 300. Subsequently, the model transitions to the MF dataset through TL. In this phase, three out of the six layers are kept frozen, while the remaining layers are fine–tuned with a batch size of 16, a learning rate of 0.0081, a maximum of 8000 epochs, and an early stopping patience of 500. Freezing part of the model ensures that knowledge gathered from the LF training is preserved, while the unfrozen layers adapt to the higher–fidelity data.

Model	Hyperparameter	Best value
MF-BayNet	N_{layers}	6
	N_{units}	[224, 144, 160, 112, 96] + [10] output
	$lr(LF \rightarrow MF)$	0.0081
	μ_{prior}	0
	σ_{prior}	0.0351
	$N_{frz}(LF \rightarrow MF)$	3

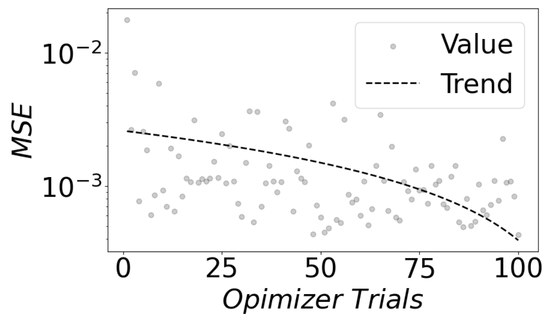
TABLE A.4: Optimized hyperparameters of the MF-BayNet model for the full-configuration test case.

A.4 Deep Learning for Steady Transonic Flowfields

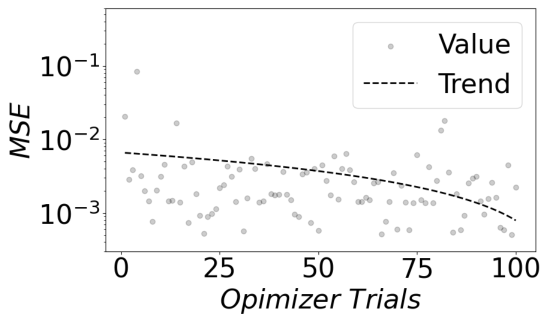
The Bayesian optimisation history of the FCNN hyperparameters tuning is shown in Figure A.1. The plot depicts the evolution of `MSE` across multiple trials for each test case. Transparent points represent the `MSE` value at the end of the training process at each trial. The black dashed line denotes the trend of the error during the optimisation. The graph highlights a consistent reduction in error throughout the optimisation process, emphasizing the effectiveness of the tuning procedure in identifying the combination of hyperparameters that yield minimal `MSE` on the validation dataset.

Table A.5 shows the hyperparameters of the neural network after the optimisation process, where a relation between the number of surface mesh points assigned in the input layer and the neural network parameters is highlighted.

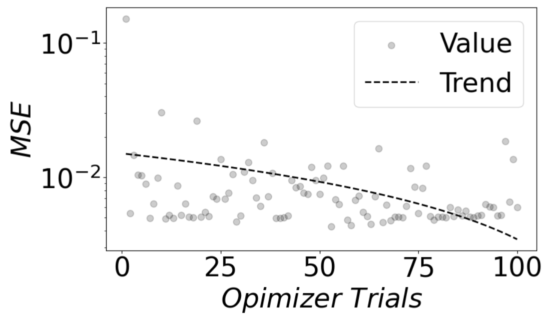
The optimum learning rate and activation function are similar between BSCW and ONERA M6 test case. This suggests that an optimal region for certain hyperparameters may exist for test cases characterized by similar physical phenomena. Hence, these hyperparameters have been fixed in order to improve the hyperparameter optimisation performance by reducing the design space.



(A) BSCW



(B) ONERA M6



(C) CRM

FIGURE A.1: Hyperparameter optimisation history of the FCNN for each test case.

Hyperparameter	**Optimal Value**		
	BSCW	ONERA M6	CRM
Learning rate	10^{-4}	10^{-4}	10^{-4}
Number of Hidden Layers	7	6	4
Batch size	6	6	6
Activation function	LeakyReLu	LeakyReLu	LeakyReLu
Number of parameters	170,756	158,180	68,392
Input surface points	130,816	149,700	78,829

TABLE A.5: Optimal hyperparameters for the three test cases.

A.5 Autoencoder Graph Convolutional Networks for Steady Transonic Flowfields on Unstructured Grids

This section provides a comprehensive overview of the optimised architectures and highlights the systematic reduction of loss throughout the optimisation trials for each test case.

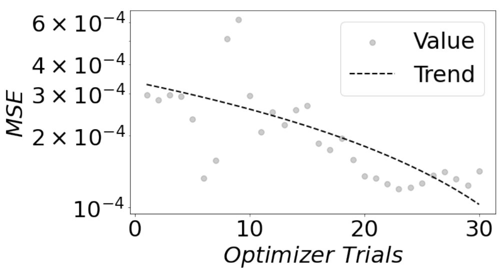
Table A.6 provides detailed information about the optimised architecture designed specifically for the wing–only test case. This architecture consists of 17 layers and a total of 711,493 parameters, carefully balanced to capture the complexities of this aerodynamic setup. Similarly, Table A.7 displays the optimised architecture for the wing–fuselage test case. With 15 layers and a total of 633,731 parameters, this configuration is

	Block	Layer	Activation	Output Size
Input				m × 86840 × 5
Encoding	Block 0	GCN	PReLU	m × 86840 × 64
	Block 1	GCN	PReLU	m × 86840 × 112
		GCN	PReLU	m × 86840 × 192
		GCN	PReLU	m × 86840 × 256
	Pooling 1			m × 28600 × 256
	Block 2	GCN	PReLU	m × 28600 × 256
		GCN	PReLU	m × 28600 × 288
	Pooling 2			m × 9600 × 288
Reduced Space	Block 3	GCN	PReLU	m × 9600 × 496
		GCN	PReLU	m × 9600 × 288
Decoding	Unpooling 2			m × 28600 × 288
	Block 4	GCN	PReLU	m × 28600 × 256
		GCN	PReLU	m × 28600 × 256
	Unpooling 1			m × 86840 × 256
	Block 5	GCN	PReLU	m × 86840 × 256
		GCN	PReLU	m × 86840 × 192
		GCN	PReLU	m × 86840 × 160
Output	Block 6	GCN	PReLU	m × 86840 × 1
		GCN	PReLU	m × 86840 × 1
		GCN	PReLU	m × 86840 × 1
		GCN	PReLU	m × 86840 × 1
	Concatenate Block 6			
	Prediction			m × 86840 × 4

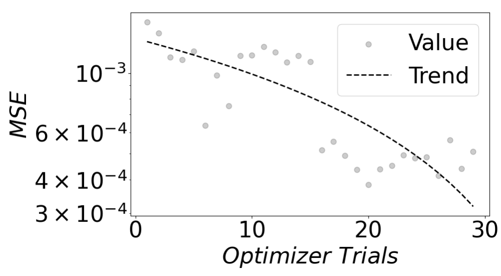
TABLE A.6: Optimal architecture for Test Case I - Wing–only Model.

tailored to accurately model the interaction between the wing and fuselage, capturing the subtle aerodynamic interactions between these components.

Figure A.2 illustrates the optimisation history of GB-AE-GCN hyperparameters using Bayesian optimisation. Each trial is represented by a set of transparent points indicating the MSE at the end of training. The dashed black line indicates the trend of error reduction during optimisation. The graph underscores a continual decrease in error during the optimisation, underscoring the efficacy of the tuning process in discovering the hyperparameters combination that minimizes MSE on the validation dataset.



(A) Test Case I - Wing–only Model



(B) Test Case II - Wing–fuselage Model

FIGURE A.2: Hyperparameter optimisation history of the GB-AE-GCN model for each test case.

	Block	Layer	Activation	Output Size
Input				m × 78829 × 5
Encoding	Block 0	GCN	PReLU	m × 78829 × 224
	Block 1	GCN	PReLU	m × 78829 × 192
		GCN	PReLU	m × 78829 × 192
	Pooling 1			m × 26000 × 192
	Block 2	GCN	PReLU	m × 26000 × 240
		GCN	PReLU	m × 26000 × 304
	Pooling 2			m × 8000 × 304
Reduced Space	Block 3	GCN	PReLU	m × 8000 × 432
		GCN	PReLU	m × 8000 × 304
Decoding	Unpooling 2			m × 26000 × 304
	Block 4	GCN	PReLU	m × 26000 × 240
		GCN	PReLU	m × 26000 × 192
	Unpooling 1			m × 78829 × 192
	Block 5	GCN	PReLU	m × 78829 × 192
		GCN	PReLU	m × 78829 × 64
Output	Block 6	GCN	PReLU	m × 78829 × 1
		GCN	PReLU	m × 78829 × 1
		GCN	PReLU	m × 78829 × 1
		GCN	PReLU	m × 78829 × 1
	Concatenate Block 6			
	Prediction			m × 78829 × 4

TABLE A.7: Optimal architecture for Test Case II - Wing–fuselage Model.

A.6 Spatio-temporal Graph Convolutional Autoencoder for Unsteady Transonic Flowfields

This section outlines the architecture and training process for both the ARMA and feedforward models used in this study, as detailed in Tables A.8 and A.9. The ARMA model in Table A.8 combines autoregressive components with GCN layers and STGCN temporal layer to capture both spatial and temporal dynamics, featuring 5,775,023 trainable weights. In contrast, the feedforward model in Table A.9 avoids using previous predictions, which helps prevent error accumulation over time. This model has 1,962,111 trainable weights. Both models utilize a pre-trained AE for dimensionality reduction. Specifically, the Encoding A and Decoding layers in both architectures are optimised based on the pre-trained AE, while Encoding B mirrors the structure of Encoding A, ensuring consistent feature extraction across different model variants. Additionally, the concatenation block in both models concatenates the encodings from the previous three timesteps, enabling the temporal layer to effectively capture and process the sequential dependencies within the data.

During the backpropagation phase, the Adam optimizer [181] was employed to fine-tune the neural network weights and minimize the `MAE` loss function. The learning rate was set to 0.001. A batch size m of 1 was found to yield the most accurate results. The training process was carried out over 50 epochs.

	Layer Type	Output Size		Layer Type	Output Size
Encoding A	Input	$m \times 3 \times 86840 \times 8$	**Encoding B**	Input	$m \times 3 \times 86840 \times 1$
	GCN	$m \times 3 \times 86840 \times 256$		GCN	$m \times 3 \times 86840 \times 256$
	GCN	$m \times 3 \times 86840 \times 224$		GCN	$m \times 3 \times 86840 \times 224$
	GCN	$m \times 3 \times 86840 \times 96$		GCN	$m \times 3 \times 86840 \times 96$
	Pooling 1	$m \times 3 \times 28600 \times 96$		Pooling 1	$m \times 3 \times 28600 \times 96$
	GCN	$m \times 3 \times 28600 \times 64$		GCN	$m \times 3 \times 28600 \times 64$
	Pooling 2	$m \times 3 \times 9600 \times 64$		Pooling 2	$m \times 3 \times 9600 \times 64$
	GCN	$m \times 3 \times 9600 \times 368$		GCN	$m \times 3 \times 9600 \times 368$

Concatenate Block – Output: $m \times 3 \times 9600 \times 736$

Temporal Layer – Output: $m \times 9600 \times 368$

	Layer Type	Output Size
Decoding	GCN	$m \times 9600 \times 368$
	Unpooling 2	$m \times 28600 \times 368$
	GCN	$m \times 28600 \times 64$
	Unpooling 1	$m \times 86840 \times 64$
	GCN	$m \times 86840 \times 96$
	GCN	$m \times 86840 \times 224$
	GCN	$m \times 86840 \times 256$
	Output	$m \times 86840 \times 1$

TABLE A.8: Layer structure and output dimensions for the ARMA model, detailing the two encodings, temporal, and decoding layers used for predicting pressure distribution.

	Layer Type	Output Size
Encoding A	Input	$m \times 3 \times 86840 \times 8$
	GCN	$m \times 3 \times 86840 \times 256$
	GCN	$m \times 3 \times 86840 \times 224$
	GCN	$m \times 3 \times 86840 \times 96$
	Pooling 1	$m \times 3 \times 28600 \times 96$
	GCN	$m \times 3 \times 28600 \times 64$
	Pooling 2	$m \times 3 \times 9600 \times 64$
	GCN	$m \times 3 \times 9600 \times 368$

Temporal Layer – Output: $m \times 9600 \times 368$

	Layer Type	Output Size
Decoding	GCN	$m \times 9600 \times 368$
	Unpooling 2	$m \times 28600 \times 368$
	GCN	$m \times 28600 \times 64$
	Unpooling 1	$m \times 86840 \times 64$
	GCN	$m \times 86840 \times 96$
	GCN	$m \times 86840 \times 224$
	GCN	$m \times 86840 \times 256$
	Output	$m \times 86840 \times 1$

TABLE A.9: Layer structure and output dimensions for the Feedforward model, detailing the encoding, temporal, and decoding layers used for predicting pressure distribution.

A.7 Graph-Convolutional Autoencoder for Modal Space Analysis

This section provides a comprehensive overview of the architecture of the two proposed approaches. Table A.10 outlines the architecture for Approach 1: GCN-MM-AE, which consists of 9 layers and 1,556,500 parameters. This model integrates GCN layers into an autoencoder structure, enabling dimensionality reduction on irregular, unstructured meshes. While effective in capturing key aerodynamic features, this approach employs a straightforward design with a relatively high number of parameters, leading to higher computational costs compared to the second approach.

Table A.11 details the optimised architecture for Approach 2: GB-AE-GCN, comprising 17 layers and 656,739 parameters. This model includes additional GCN layers and advanced encoding-decoding blocks for improved nonlinearity and learning performance. Despite the increased architectural complexity, this approach reduces the total number of parameters, achieving a more efficient balance between computational cost and prediction accuracy.

	Block	Layer	Activation	Output Size
Input				m × 45943 × 9
Encoding	Block 1	GCN	PReLU	m × 45943 × 216
		GCN	PReLU	m × 45943 × 432
	Pooling			m × 5000 × 432
Reduced Space	Block 2	GCN	PReLU	m × 5000 × 432
		GCN	PReLU	m × 5000 × 864
		GCN	PReLU	m × 5000 × 432
		GCN	PReLU	m × 5000 × 432
Decoding	Unpooling			m × 45943 × 432
	Block 3	GCN	PReLU	m × 45943 × 216
		GCN	PReLU	m × 45943 × 216
Output	Concatenate × 4			m × 45943 × 216 × 4
	Block 4	GCN	PReLU	m × 45943 × 216 × 4
		GCN	–	m × 45943 × 1 × 4
	Prediction			m × 45943 × 4

TABLE A.10: Architecture for Approach 1: GCN-MM-AE.

	Block	Layer	Activation	Output Size
Input				m × 45943 × 9
Encoding	Block 0	GCN	PReLU	m × 45943 × 224
	Block 1	GCN	PReLU	m × 45943 × 208
		GCN	PReLU	m × 45943 × 256
		GCN	PReLU	m × 45943 × 256
	Pooling 1			m × 11486 × 256
	Block 2	GCN	PReLU	m × 11486 × 224
	Pooling 2			m × 2871 × 224
Reduced Space	Block 3	GCN	PReLU	m × 2871 × 512
		GCN	PReLU	m × 2871 × 512
Decoding	Unpooling 2			m × 11486 × 512
	Block 4	GCN	PReLU	m × 11486 × 224
	Unpooling 1			m × 45943 × 224
	Block 5	GCN	PReLU	m × 45943 × 256
		GCN	PReLU	m × 45943 × 256
		GCN	PReLU	m × 45943 × 208
Output	Block 6	GCN	PReLU	m × 45943 × 112
		GCN	PReLU	m × 45943 × 112
		GCN	PReLU	m × 45943 × 112
		GCN	PReLU	m × 45943 × 112
	Concatenate Block 6			
	Prediction			m × 45943 × 4

TABLE A.11: Optimised architecture for Approach 2: GB-AE-GCN.

Bibliography

[1] Abadi. TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems. Technical report, University of Southampton, 2016.

[2] S. Abou-Kebeh, R. Gil-Pita, and M. Rosa-Zurera. Application of deep learning to identify flutter flight testing signals parameters and analysis of real f-18 flutter flight test data. *Aerospace*, 12(1):34, 2025.

[3] T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 2623–2631, 2019.

[4] J. J. Alonso, M. S. Eldred, P. Constantine, K. Duraisamy, C. Farhat, G. Iaccarino, and J. Jakeman. Scalable environment for quantification of uncertainty and optimization in industrial applications (sequoia). In *19th AIAA Non-Deterministic Approaches Conference*, page 1327, 2017.

[5] K. Amiri, M. R. Soltani, and A. Haghiri. Steady flow quality assessment of a modified transonic wind tunnel. *Scientia Iranica*, 20(3):500–507, 2013.

[6] D. Amsallem, J. Cortial, K. Carlberg, and C. Farhat. A method for interpolating on manifolds structural dynamics reduced-order models. *International journal for numerical methods in engineering*, 80(9):1241–1258, 2009.

[7] D. Amsallem and C. Farhat. An interpolation method for adapting reduced-order models and application to aeroelasticity. *SIAM Journal on Scientific Computing*, 31(2):1109–1134, 2008.

[8] D. Amsallem and C. Farhat. Interpolation method for adapting reduced-order models and application to aeroelasticity. *AIAA journal*, 46(7):1803–1813, 2008.

[9] D. Amsallem and C. Farhat. An online method for interpolating linear parametric reduced-order models. *SIAM Journal on Scientific Computing*, 33(5):2169–2198, 2011.

[10] D. Amsallem, M. J. Zahr, and C. Farhat. Nonlinear model order reduction based on local reduced-order bases. *International Journal for Numerical Methods in Engineering*, 92(10):891–916, 2012.

[11] E. Andrés-Pérez and C. Paulete-Periáñez. On the application of surrogate regression models for aerodynamic coefficient prediction. *Complex & Intelligent Systems*, 7:1991–2021, 2021.

[12] M. Anhichem, S. Timme, J. Castagna, A. J. Peace, and M. Maina. Data fusion of wing pressure distributions using scalable gaussian processes. *AIAA Journal*, 62(5):1946–1961, 2024.

[13] H. Arbabi and I. Mezić. Ergodic theory, dynamic mode decomposition, and computation of spectral properties of the koopman operator. *SIAM Journal on Applied Dynamical Systems*, 16(4):2096–2126, 2017.

[14] H. Asada and S. Kawai. Pod and dmd of three-dimensional transonic aircraft buffet using large-scale les data. In *AIAA SCITECH 2024 Forum*, page 0494, 2024.

[15] V. Asouti, M. Kontou, and K. Giannakoglou. Radial basis function surrogates for uncertainty quantification and aerodynamic shape optimization under uncertainties. *Fluids*, 8(11):292, 2023.

[16] K. J. Asztalos, S. T. Dawson, and D. R. Williams. Modeling the flow state sensitivity of actuation response on a stalled airfoil. *AIAA Journal*, 59(8):2901–2915, 2021.

[17] K. J. Badcock, S. Timme, S. Marques, H. Khodaparast, M. Prandina, J. E. Mottershead, A. Swift, A. Da Ronch, and M. A. Woodgate. Transonic aeroelastic simulation for instability searches and uncertainty analysis. *Progress in Aerospace Sciences*, 47(5):392–423, 2011.

[18] J. Bai, J. Zhu, Y. Song, L. Zhao, Z. Hou, R. Du, and H. Li. A3t-gcn: Attention temporal graph convolutional network for traffic forecasting. *ISPRS International Journal of Geo-Information*, 10(7):485, 2021.

[19] M. Balajewicz and E. Dowell. Reduced-order modeling of flutter and limit-cycle oscillations using the sparse volterra series. *Journal of Aircraft*, 49(6):1803–1812, 2012.

[20] M. Balajewicz, F. Nitzsche, and D. Feszty. Application of multi-input volterra theory to nonlinear multi-degree-of-freedom aerodynamic systems. *AIAA Journal*, 48(1):56–62, 2010.

[21] M. Balajewicz, F. Nitzsche, and D. Feszty. Application of multi-input volterra theory to nonlinear multi-degree-of-freedom aerodynamic systems. *AIAA journal*, 48(1):56–62, 2010.

[22] P. Baqué and P. Fua. Multi-fidelity optimization of a fixed-wing drone using Geometric Convolutional Neural Networks . 2020.

[23] P. Baqué, E. Remelli, F. Fleuret, and P. Fua. Geodesic Convolutional Shape Optimization. 2 2018.

[24] R. E. Bartels and B. Stanford. Economical unsteady high fidelity aerodynamics in a structural optimization with a flutter constraint. In *35th AIAA Applied Aerodynamics Conference*, page 4358, 2017.

[25] T. Ben Asher and D. E. Raveh. Aeroelastic system identification for flutter prediction via multi-output autoregressive modeling. *Journal of Aircraft*, 61(2):470–484, 2024.

[26] O. O. Bendiksen. Review of unsteady transonic aerodynamics: Theory and applications. *Progress in Aerospace Sciences*, 47(2):135–167, 2011.

[27] S. Bengio, O. Vinyals, N. Jaitly, and N. Shazeer. Scheduled sampling for sequence prediction with recurrent neural networks. *Advances in neural information processing systems*, 28, 2015.

[28] 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, 1994.

[29] J. Bergstra, R. Bardenet, Y. Bengio, and B. Kégl. Algorithms for hyper-parameter optimization. In *Advances in Neural Information Processing Systems 24 (NeurIPS 2011)*, pages 2546–2554. Curran Associates, 2011.

[30] G. Berkooz, P. Holmes, and J. L. Lumley. The proper orthogonal decomposition in the analysis of turbulent flows. *Annual Review of Fluid Mechanics*, 25:539–575, 1993.

[31] S. Bhatnagar, Y. Afshar, S. Pan, K. Duraisamy, and S. Kaushik. Prediction of aerodynamic flow fields using convolutional neural networks. *Computational Mechanics*, 64:525–545, 2019.

[32] E. D. V. Bigarella and J. L. F. Azevedo. Advanced eddy-viscosity and reynolds-stress turbulence model simluations of aerospace applications. *AIAA journal*, 45(10):2369–2390, 2007.

[33] Billings. *Nonlinear System Identification*, chapter 2. John Wiley & Sons, Chichester, UK, 1 edition, 2013.

[34] S. A. Billings. *Nonlinear System Identification: NARMAX Methods in the Time, Frequency, and Spatio-Temporal Domains*. Wiley, Chichester, 2013.

[35] B. Bischl, M. Binder, M. Lang, T. Pielok, J. Richter, S. Coors, J. Thomas, T. Ullmann, M. Becker, A.-L. Boulesteix, et al. Hyperparameter optimization: Foundations, algorithms, best practices, and open challenges. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 13(2):e1484, 2023.

[36] C. M. Bishop. Training with noise is equivalent to tikhonov regularization. *Neural computation*, 7(1):108–116, 1995.

[37] J. Blazek. *Computational fluid dynamics: principles and applications*. Butterworth-Heinemann, 2015.

[38] S. Boluki, S. Z. Dadaneh, E. R. Dougherty, and X. Qian. Bayesian proper orthogonal decomposition for learnable reduced-order models with uncertainty quantification. *IEEE Transactions on Artificial Intelligence*, 5(3):1162–1173, 2023.

[39] J. Borée. Extended proper orthogonal decomposition: a tool to analyse correlated events in turbulent flows. *Experiments in fluids*, 35(2):188–192, 2003.

[40] L. Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996.

[41] M. Brenner, J. Eldredge, and J. Freund. Perspective on machine learning for advancing fluid mechanics. *Physical Review Fluids*, 4(10):100501, 2019.

[42] L. Brevault, M. Balesdent, and A. Hebbal. Overview of gaussian process based multi-fidelity techniques with variable relationship between fidelities, application to aerospace systems. *Aerospace Science and Technology*, 107:106339, 2020.

[43] M. M. Bronstein, J. Bruna, T. Cohen, and P. Veličković. Geometric deep learning: Grids, groups, graphs, geodesics, and gauges. *arXiv preprint arXiv:2104.13478*, 2021.

[44] M. M. Bronstein, J. Bruna, T. Cohen, and P. Veličković. Geometric Deep Learning Grids, Groups, Graphs, Geodesics, and Gauges. 2021.

[45] M. M. Bronstein, J. Bruna, Y. LeCun, A. Szlam, and P. Vandergheynst. Geometric deep learning: going beyond euclidean data. *IEEE Signal Processing Magazine*, 34(4):18–42, 2017.

[46] M. M. Bronstein, J. Bruna, Y. Lecun, A. Szlam, and P. Vandergheynst. Geometric Deep Learning: Going beyond Euclidean data. *IEEE Signal Processing Magazine*, 34(4):18–42, 2017.

[47] S. L. Brunton, B. R. Noack, and P. Koumoutsakos. Machine learning for fluid mechanics. *Annual Review of Fluid Mechanics*, 52:477–508, 2020.

[48] S. L. Brunton, J. L. Proctor, and J. N. Kutz. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proceedings of the National Academy of Sciences*, 113(15):3932–3937, 2016.

[49] P. Bühlmann and S. Van De Geer. *Statistics for high-dimensional data: methods, theory and applications*. Springer Science & Business Media, 2011.

[50] K. P. Burnham and D. R. Anderson. *Model Selection and Multimodel Inference: A Practical Information-Theoretic Approach*. Springer, New York, 2 edition, 2002.

[51] R. E. Caflisch. Monte carlo and quasi-monte carlo methods. *Acta numerica*, 7:1–49, 1998.

[52] R. J. Campello, G. Favier, and W. C. Do Amaral. Optimal expansions of discrete-time volterra models using laguerre functions. *Automatica*, 40(5):815–822, 2004.

[53] A. C. N. Carloni and J. L. F. Azevedo. Development of transonic unsteady aerodynamic reduced-order models using system identification techniques. *arXiv preprint arXiv:2307.09644*, 2023.

[54] M. A. Carrizales, G. Dussart, V. Portapas, A. Pontillo, and M. Lone. Verification of a low fidelity fast simulation framework through rans simulations. *CEAS Aeronautical Journal*, 11:161–176, 2020.

[55] R. Castellanos, J. B. Varela, A. Gorgues, and E. Andrés. An assessment of reduced-order and machine learning models for steady transonic flow prediction on wings. In *ICAS 2022*, 2022.

[56] G. Catalani, S. Agarwal, X. Bertrand, F. Tost, M. Bauerheim, and J. Morlier. Neural fields for rapid aircraft aerodynamics simulations. *Scientific Reports*, 14(1):25496, 2024.

[57] L. R. Chai. Uncertainty estimation in bayesian neural networks and links to interpretability. *Master's Thesis, Massachusetts Institute of Technology*, 2018.

[58] S. Chakraborty. Transfer learning based multi-fidelity physics informed deep neural network. *Journal of Computational Physics*, 426:109942, 2021.

[59] K. Champion, B. Lusch, J. N. Kutz, and S. L. Brunton. Data-driven discovery of coordinates and governing equations. *Proceedings of the National Academy of Sciences*, 116(45):22445–22451, 2019.

[60] A. Chatterjee. An introduction to the proper orthogonal decomposition. *Current science*, pages 808–817, 2000.

[61] T. Chen, E. Fox, and C. Guestrin. Stochastic gradient hamiltonian monte carlo. In *International conference on machine learning*, pages 1683–1691. PMLR, 2014.

[62] T. Chen, Z. Ren, G. Lin, Z. Wu, and B.-L. Ye. Real-time computational optimal control of an mhd flow system with parameter uncertainty quantification. *Journal of the Franklin Institute*, 357(5):2830–2850, 2020.

[63] W. Chen, K. Chiu, and M. D. Fuge. Aerodynamic design optimization and shape exploration using generative adversarial networks. In *AIAA Scitech 2019 Forum*. American Institute of Aeronautics and Astronautics Inc, AIAA, 2019.

[64] C. Cheng, Z. Peng, W. Zhang, and G. Meng. Volterra-series-based nonlinear system modeling and its engineering applications: A state-of-the-art review. *Mechanical Systems and Signal Processing*, 87:340–364, 2017.

[65] J. Cheng, L. Dong, and M. Lapata. Long short-term memory-networks for machine reading. *arXiv preprint arXiv:1601.06733*, 2016.

[66] X. Cheng, F. Shi, M. Zhao, G. Li, H. Zhang, and S. Chen. Temporal attention convolutional neural network for estimation of icing probability on wind turbine blades. *IEEE Transactions on Industrial Electronics*, 69(6):6371–6380, 2021.

[67] K. Cho, B. Van Merriënboer, D. Bahdanau, and Y. Bengio. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259*, 2014.

[68] E. Choi, M. T. Bahadori, L. Song, W. F. Stewart, and J. Sun. Gram: graph-based attention model for healthcare representation learning. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 787–795, 2017.

[69] I. A. Chowdhury. State-of-the-art cfd simulation: A review of techniques, validation methods, and application scenarios. *J. Recent Trends Mech*, 9:45–53, 2024.

[70] P. Cinnella. Numerical study of transonic shock/boundary layer interactions on an oscillating airfoil using a third-order scheme and nonlinear turbulence models. In *Computational Fluid Dynamics 2000: Proceedings of the First International Conference on Computational Fluid Dynamics, ICCFD, Kyoto, Japan, 10–14 July 2000*, pages 157–162. Springer, 2001.

[71] G. Coppotelli, R. G. Sbarra, L. Onofri, M. Righi, G. Immordino, and A. Da Ronch. Experimental characterization of the flutter behavior of a very flexible wing. In *AIAA SCITECH 2025 Forum*, page 1019, 2025.

[72] T. J. Cowan, A. S. A. Jr., and K. K. Gupta. Accelerating computational fluid dynamics based aeroelastic predictions using system identification. *Journal of Aircraft*, 38(1):81–87, 2001.

[73] J. D. Crouch, A. Garbaruk, D. Magidov, and A. Travin. Origin of transonic buffet on aerofoils. *Journal of Fluid Mechanics*, 628:357–369, 2009.

[74] A. Crovato, H. S. Almeida, G. Vio, G. H. Silva, A. P. Prado, C. Breviglieri, H. Guner, P. H. Cabral, R. Boman, V. E. Terrapon, et al. Effect of levels of fidelity

on steady aerodynamic and static aeroelastic computations. *Aerospace*, 7(4):42, 2020.

[75] H. J. Cunningham, J. T. Batina, and R. M. Bennett. Modern wing flutter analysis by computational fluid dynamics methods. *Journal of Aircraft*, 25(10):962–968, 1988.

[76] A. Da Ronch, J. Drofelnik, M. P. van Rooij, J. C. Kok, M. Panzeri, and A. Voß. Aerodynamic and aeroelastic uncertainty quantification of NATO STO AVT-251 unmanned combat aerial vehicle. *Aerospace Science and Technology*, 91:627–639, 2019.

[77] A. Da Ronch, M. Ghoreyshi, and K. Badcock. On the generation of flight dynamics aerodynamic tables by computational fluid dynamics. *Progress in Aerospace Sciences*, 47(8):597–620, 2011.

[78] A. Da Ronch, G. Immordino, and J. W. Kim. A preliminary investigation into icing accretion around a wavy leading-edge wing. In *AIAA SCITECH 2023 Forum*, page 2459, 2023.

[79] A. Da Ronch, A. J. McCracken, and K. J. Badcock. Assessing the impact of aerodynamic modelling on manoeuvring aircraft. In *AIAA Atmospheric Flight Mechanics Conference*, page 0732, 2014.

[80] A. Damianou. *Deep Gaussian processes and variational propagation of uncertainty*. PhD thesis, University of Sheffield, 2015.

[81] A. Damianou and N. D. Lawrence. Deep gaussian processes. In *Artificial intelligence and statistics*, pages 207–215. PMLR, 2013.

[82] A. C. Damianou and N. D. Lawrence. Deep gaussian processes. *ArXiv*, abs/1211.0358, 2012.

[83] B. DANSBERRY, M. DURHAM, R. BENNETT, J. RIVERA, W. SILVA, C. WIESEMAN, and D. TURNOCK. Experimental unsteady pressures at flutter on the supercritical wing benchmark model. In *34th Structures, structural dynamics and materials conference*, page 1592, 1993.

[84] A. Das, P. Marzocca, R. Das, and O. Levinski. Transonic shock buffet flowfield assessment using various dynamic mode decomposition techniques. *Journal of Aircraft*, 60(4):1038–1049, 2023.

[85] S. Daulton, X. Wan, D. Eriksson, M. Balandat, M. A. Osborne, and E. Bakshy. Bayesian optimization over discrete and mixed spaces via probabilistic reparameterization. *Advances in Neural Information Processing Systems*, 35:12760–12774, 2022.

[86] B. Davoudi and K. Duraisamy. A hybrid blade element momentum model for flight simulation of rotary wing unmanned aerial vehicles. In *AIAA Aviation 2019 Forum*, page 2823, 2019.

[87] S. T. M. Dawson, M. S. Hemati, M. O. Williams, and C. W. Rowley. Characterizing and correcting for the effect of sensor noise in the dynamic mode decomposition, 2015.

[88] R. De Maesschalck, D. Jouan-Rimbaud, and D. L. Massart. The mahalanobis distance. *Chemometrics and intelligent laboratory systems*, 50(1):1–18, 2000.

[89] N. De Paula and F. D. Marques. Multi-variable volterra kernels identification using time-delay neural networks: application to unsteady aerodynamic loading. *Nonlinear Dynamics*, 97:767–780, 2019.

[90] N. C. de Paula, F. D. Marques, and W. A. Silva. Volterra kernels assessment via time-delay neural networks for nonlinear unsteady aerodynamic loading identification. *AIAA Journal*, 57(4):1725–1735, 2019.

[91] Z. Deng, Y. Chen, Y. Liu, and K. C. Kim. Time-resolved turbulent velocity field reconstruction using a long short-term memory (lstm)-based artificial intelligence framework. *Physics of Fluids*, 31(7):075108, 2019.

[92] S. Depeweg, J. M. Hernández-Lobato, F. Doshi-Velez, and S. Udluft. Uncertainty decomposition in bayesian neural networks with latent variables. *arXiv preprint arXiv:1706.08495*, 2017.

[93] A. Der Kiureghian and O. Ditlevsen. Aleatory or epistemic? does it matter? *Structural safety*, 31(2):105–112, 2009.

[94] T. J. Dodd and R. F. Harrison. A new solution to volterra series estimation. *IFAC Proceedings Volumes*, 35(1):67–72, 2002.

[95] J. Dominique, J. Berghe, C. Schram, and M. Mendez. Artificial neural networks modelling of wall pressure spectra beneath turbulent boundary layers. *arXiv preprint arXiv:2201.03262*, 2022.

[96] E. Dowell. Reduced-order modeling: a personal journey. *Nonlinear Dynamics*, 111(11):9699–9720, 2023.

[97] E. H. Dowell, R. Scanlan, F. Sisto, H. Curtiss Jr, and H. Saunders. A modern course in aeroelasticity. 1981.

[98] J. Du, X. Li, S. Dong, Z. Liu, and G. Chen. A novel attention enhanced deep neural network for hypersonic spatiotemporal turbulence prediction. *Physics of Fluids*, 36(5), 2024.

[99] X. Du, P. He, and J. R. Martins. Rapid airfoil design optimization via neural networks-based parameterization and surrogate modeling. *Aerospace Science and Technology*, 113:106701, 2021.

[100] K. Duraisamy, G. Iaccarino, and H. Xiao. Turbulence modeling in the age of data. *Annual Review of Fluid Mechanics*, 51:357–377, 2019.

[101] N. Durrande, D. Ginsbourger, and O. Roustant. Additive kernels for gaussian process modeling. *arXiv preprint arXiv:1103.4023*, 2011.

[102] G. Duthé, I. Abdallah, S. Barber, and E. Chatzi. Graph Neural Networks for Aerodynamic Flow Reconstruction from Sparse Sensing. 1 2023.

[103] D. K. Duvenaud, D. Maclaurin, J. Iparraguirre, R. Bombarell, T. Hirzel, A. Aspuru-Guzik, and R. P. Adams. Convolutional networks on graphs for learning molecular fingerprints. *Advances in neural information processing systems*, 28, 2015.

[104] T. D. Economon, F. Palacios, S. R. Copeland, T. W. Lukaczyk, and J. J. Alonso. Su2: An open-source suite for multiphysics simulation and design. *Aiaa Journal*, 54(3):828–846, 2016.

[105] H. Eivazi, H. Veisi, M. H. Naderi, and V. Esfahanian. Deep neural networks for nonlinear model order reduction of unsteady flows. *Physics of Fluids*, 32(10), 2020.

[106] O. U. Espinosa Barcenas, J. G. Quijada Pioquinto, E. Kurkina, and O. Lukyanov. Surrogate aerodynamic wing modeling based on a multilayer perceptron. *Aerospace*, 10(2):149, 2023.

[107] S. Falkner, A. Klein, and F. Hutter. Bohb: Robust and efficient hyperparameter optimization at scale. In *International conference on machine learning*, pages 1437–1446. PMLR, 2018.

[108] C. Fallet, A. Garbo, A. Kiener, and P. Bekemeyer. Reduced order model for steady aerodynamics applications based on navier-stokes residual vector minimization. In *AIAA AVIATION 2023 Forum*, page 3715, 2023.

[109] Z. Feng. Flutter phenomenon and safety implications in transonic flow. *Highlights in Science, Engineering and Technology*, 76:198–204, 2023.

[110] M. G. Fernández-Godino. Review of multi-fidelity models. *arXiv preprint arXiv:1609.07196*, 2016.

[111] N. Fonzi, S. L. Brunton, and U. Fasel. Data-driven nonlinear aeroelastic models of morphing wings for control. *Proceedings of the Royal Society A*, 476(2239):20200079, 2020.

[112] N. Fonzi, S. L. Brunton, and U. Fasel. Data-driven modeling for transonic aeroelastic analysis. *Journal of Aircraft*, 61(2):625–637, 2024.

[113] N. Fonzi, V. Cavalieri, A. De Gaspari, S. Ricci, and A. Preprint. Advances of the Python-based Fluid-Structure Interaction capabilities included in SU2 A Preprint Advances of the Python FSI capabilities in SU2. Technical report, 2021.

[114] A. I. Forrester, A. Sóbester, and A. J. Keane. Multi-fidelity optimization via surrogate modelling. *Proceedings of the royal society a: mathematical, physical and engineering sciences*, 463(2088):3251–3269, 2007.

[115] T. Franz. *Reduced-order modeling for steady transonic flows via manifold learning*. PhD thesis, Deutsches Zentrum für Luft-und Raumfahrt, 2016.

[116] T. Franz, R. Zimmermann, S. Görtz, and N. Karcher. Interpolation-based reduced-order modelling for steady transonic flows via manifold learning. *International Journal of Computational Fluid Dynamics*, 28(3-4):106–121, 2014.

[117] J. H. Friedman, J. L. Bentley, and R. A. Finkel. An algorithm for finding best matches in logarithmic expected time. *ACM Transactions on Mathematical Software (TOMS)*, 3(3):209–226, 1977.

[118] K. Fukami, K. Fukagata, and K. Taira. Super-resolution reconstruction of turbulent flows with machine learning. *Journal of Fluid Mechanics*, 870:106–120, 2019.

[119] K. Fukami, K. Fukagata, and K. Taira. Assessment of supervised machine learning methods for fluid flows. *Theoretical and Computational Fluid Dynamics*, 34(4):497–519, 2020.

[120] K. Fukami, T. Nakamura, and K. Fukagata. Convolutional neural network based hierarchical autoencoder for nonlinear mode decomposition of fluid field data. *Physics of Fluids*, 32(9), 2020.

[121] E. C. Garrido-Merchán and D. Hernández-Lobato. Dealing with categorical and integer-valued variables in bayesian optimization with gaussian processes. *Neurocomputing*, 380:20–35, 2020.

[122] R. G. Ghanem and P. D. Spanos. Spectral stochastic finite-element formulation for reliability analysis. *Journal of Engineering Mechanics*, 117(10):2351–2372, 1991.

[123] H. Ghorbani. Mahalanobis distance and its application for detecting multivariate outliers. *Facta Universitatis, Series: Mathematics and Informatics*, pages 583–595, 2019.

[124] G. Giangaspero, D. MacManus, and I. Goulos. Surrogate models for the prediction of the aerodynamic performance of exhaust systems. *Aerospace Science and Technology*, 92:77–90, 2019.

[125] N. F. Giannelis, G. A. Vio, and O. Levinski. A review of recent developments in the understanding of transonic shock buffet. *Progress in Aerospace Sciences*, 92:39–84, 2017.

[126] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl. Neural Message Passing for Quantum Chemistry. Technical report, 2017.

[127] M. Gönen and E. Alpaydın. Multiple kernel learning algorithms. *The Journal of Machine Learning Research*, 12:2211–2268, 2011.

[128] M. Gori, G. Monfardini, and F. Scarselli. A new model for learning in graph domains. In *Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005.*, volume 2, pages 729–734. IEEE, 2005.

[129] A. Graves. Practical variational inference for neural networks. *Advances in neural information processing systems*, 24, 2011.

[130] J. Gu, Z. Wang, J. Kuen, L. Ma, A. Shahroudy, B. Shuai, T. Liu, X. Wang, G. Wang, J. Cai, et al. Recent advances in convolutional neural networks. *Pattern recognition*, 77:354–377, 2018.

[131] M. Guo, A. Manzoni, M. Amendt, P. Conti, and J. S. Hesthaven. Multi-fidelity regression using artificial neural networks: efficient approximation of parameter-dependent output quantities. *Computer methods in applied mechanics and engineering*, 389:114378, 2022.

[132] X. Guo, W. Li, and F. Iorio. Convolutional neural networks for steady flow approximation. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 481–490, 2016.

[133] R. Halder, M. Damodaran, and B. C. Khoo. Deep learning based reduced order model for airfoil-gust and aeroelastic interaction. *AIAA Journal*, 58(10):4304–4321, 2020.

[134] K. C. Hall, J. P. Thomas, and W. S. Clark. Computation of unsteady nonlinear flows in cascades using a harmonic balance technique. *AIAA journal*, 40(5):879–886, 2002.

[135] K. C. Hall, J. P. Thomas, and E. H. Dowell. Proper orthogonal decomposition technique for transonic unsteady aerodynamic flows. *AIAA journal*, 38(10):1853–1862, 2000.

[136] D. K. Hammond, P. Vandergheynst, and R. Gribonval. Wavelets on graphs via spectral graph theory. *Applied and Computational Harmonic Analysis*, 30(2):129–150, 2011.

[137] R. Han, Y. Wang, Y. Zhang, and G. Chen. A novel spatial-temporal prediction method for unsteady wake flows based on hybrid deep neural network. *Physics of Fluids*, 31(12):127101, 2019.

[138] X. Han, H. Gao, T. Pfaff, J.-X. Wang, and L.-P. Liu. Predicting physics in mesh-reduced space with temporal attention. *arXiv preprint arXiv:2201.09113*, 2022.

[139] Z.-H. Han and S. Görtz. Hierarchical kriging model for variable-fidelity surrogate modeling. *AIAA journal*, 50(9):1885–1896, 2012.

[140] L. He, W. Qian, T. Zhao, and Q. Wang. Multi-fidelity aerodynamic data fusion with a deep neural network modeling method. *Entropy*, 22(9):1022, 2020.

[141] A. Hebbal, L. Brevault, M. Balesdent, E.-G. Talbi, and N. Melab. Multi-objective optimization using deep gaussian processes: application to aerospace vehicle design. In *AIAA Scitech 2019 Forum*, page 1973, 2019.

[142] J. Heeg and P. Chwalowski. Investigating the transonic flutter boundary of the benchmark supercritical wing. In *58th AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, page 0191, 2017.

[143] J. Heeg, P. Chwalowski, D. E. Raveh, M. J. Dalenbring, and A. Jirasek. Plans and example results for the 2nd aiaa aeroelastic prediction workshop. In *56th AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, page 0437, 2015.

[144] J. Heeg, P. Chwalowski, D. E. Raveh, A. Jirasek, and M. Dalenbring. Overview and data comparisons from the 2nd aeroelastic prediction workshop. In *34th AIAA Applied Aerodynamics Conference*, page 3121, 2016.

[145] J. Heeg, P. Chwalowski, D. Schuster, and M. Dalenbring. Overview of the Aeroelastic Prediction Workshop. In *51st AIAA Aerospace Sciences Meeting*. American Institute of Aeronautics and Astronautics Inc, AIAA, 2013.

[146] J. Heeg and D. J. Piatak. Experimental data from the benchmark supercritical wing wind tunnel test on an oscillating turntable. In *54th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, page 1802, 2013.

[147] H. Helmbold. Der unverwundene ellipsenflugel als tragende flanche. *Jahrbuch*, pages I111–I113, 1942.

[148] M. S. Hemati, C. W. Rowley, E. A. Deem, and L. N. Cattafesta. De-biasing the dynamic mode decomposition for applied koopman spectral analysis of noisy datasets. *Theoretical and Computational Fluid Dynamics*, 31(4):349–368, 2017.

[149] D. Hines and P. Bekemeyer. Graph neural networks for the prediction of aircraft surface pressure distributions. *Aerospace Science and Technology*, 137:108268, 2023.

[150] G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *science*, 313(5786):504–507, 2006.

[151] C. Hirsch. *Numerical computation of internal and external flows: The fundamentals of computational fluid dynamics*. Elsevier, 2007.

[152] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[153] P. Holmes, J. L. Lumley, G. Berkooz, and C. W. Rowley. *Turbulence, Coherent Structures, Dynamical Systems and Symmetry*. Cambridge University Press, Cambridge, 2 edition, 2012.

[154] P. J. Holmes, J. L. Lumley, G. Berkooz, J. C. Mattingly, and R. W. Wittenberg. Low-dimensional models of coherent structures in turbulence. *Physics Reports*, 287(4):337–384, 1997.

[155] S. Hosder, R. W. Walters, and M. Balch. Point-collocation nonintrusive polynomial chaos method for stochastic computational fluid dynamics. *AIAA journal*, 48(12):2721–2730, 2010.

[156] A. A. Howard, B. Jacob, and P. Stinis. Multifidelity kolmogorov-arnold networks. *ArXiv*, abs/2410.14764, 2024.

[157] D. Huang, R. Bai, S. Zhao, P. Wen, S. Wang, and S. Chen. Bayesian neural network based method of remaining useful life prediction and uncertainty quantification for aircraft engine. In *2020 IEEE International Conference on Prognostics and Health Management (ICPHM)*, pages 1–8. IEEE, 2020.

[158] F. Hutter, H. H. Hoos, and K. Leyton-Brown. Sequential model-based optimization for general algorithm configuration. In C. A. Coello Coello, editor, *Learning and Intelligent Optimization*, volume 6683 of *Lecture Notes in Computer Science*, pages 507–523. Springer, 2011.

[159] G. Immordino, A. Da Ronch, and M. Righi. Deep–learning framework for aircraft aerodynamics prediction. In *AIAA AVIATION 2023 Forum*, page 3846, 2023.

[160] G. Immordino, A. Da Ronch, and M. Righi. Steady-state flowfield prediction in transonic regime via deep-learning framework. *AIAA Journal*, 2023.

[161] G. Immordino, A. Da Ronch, and M. Righi. Steady-state transonic flowfield prediction via deep-learning framework. *AIAA Journal*, pages 1–17, 2024.

[162] G. Immordino, A. Da Ronch, and M. Righi. Parametric nonlinear volterra series via machine learning: Transonic aerodynamics. *Journal of Aircraft*, 0(0):1–18, 2025.

[163] G. Immordino, A. Vaiuso, A. Da Ronch, and M. Righi. Predicting transonic flowfields in non-homogeneous unstructured grids using autoencoder graph convolutional networks. *arXiv preprint arXiv:2405.04396*, 2024.

[164] G. Immordino, A. Vaiuso, A. Da Ronch, and M. Righi. Predicting transonic flowfields in non–homogeneous unstructured grids using autoencoder graph convolutional networks. *Journal of Computational Physics*, page 113708, 2025.

[165] G. Immordino, A. Vaiuso, A. Da Ronch, and M. Righi. Spatio-temporal graph convolutional autoencoder for transonic wing pressure distribution forecasting. *Aerospace Science and Technology*, page 110516, 2025.

[166] B. W. Israelsen and D. A. Smith. Generalized laguerre reduction of the volterra kernel for practical identification of nonlinear dynamic systems. *arXiv preprint arXiv:1410.0741*, 2014.

[167] X. Jia, C. Gong, W. Ji, and C. Li. An accuracy-enhanced transonic flow prediction method fusing deep learning and a reduced-order model. *Physics of Fluids*, 36(5), 2024.

[168] X. Jia, C. Li, W. Ji, and C. Gong. A hybrid reduced-order model combing deep learning for unsteady flow. *Physics of Fluids*, 34(9), 2022.

[169] X. Jin, P. Cheng, W.-L. Chen, and H. Li. Prediction model of velocity field around circular cylinder over various reynolds numbers by fusion convolutional neural networks based on pressure on the cylinder. *Physics of Fluids*, 30(4):047105, 2018.

[170] Z. Jin, B. Zheng, C. Kim, and G. X. Gu. Leveraging graph neural networks and neural operator techniques for high-fidelity mesh-based physics simulations. *APL Machine Learning*, 1(4), 2023.

[171] Z. Jin, B. Zheng, C. Kim, and G. X. Gu. Leveraging graph neural networks and neural operator techniques for high-fidelity mesh-based physics simulations. *APL Machine Learning*, 1(4), 12 2023.

[172] G. R. Joldes, H. A. Chowdhury, A. Wittek, B. Doyle, and K. Miller. Modified moving least squares with polynomial bases for scattered data approximation. *Applied mathematics and computation*, 266:893–902, 2015.

[173] P. Juangphanich, J. Rush, and N. Scannell. Predicting two-dimensional airfoil performance using graph neural networks. Technical report, NASA, 2023.

[174] M. Kammeyer. Wind tunnel facility calibrations and experimental uncertainty. In *20th AIAA Advanced Measurement and Ground Testing Technology Conference*, page 2715, 1998.

[175] M. Kang, L. K. Hwang, and B. Kwon. Machine learning flow regime classification in three-dimensional printed tubes. *Physical Review Fluids*, 5(8):081901, 2020.

[176] M. Karpel. Design for active and passive flutter suppression and gust alleviation. Technical Report NASA CR-3482, NASA, 1981.

[177] A. Kashefi, D. Rempe, and L. J. Guibas. A Point-Cloud Deep Learning Framework for Prediction of Fluid Flow Fields on Irregular Geometries. 2020.

[178] J. Kawahara, C. J. Brown, S. P. Miller, B. G. Booth, V. Chau, R. E. Grunau, J. G. Zwicker, and G. Hamarneh. Brainnetcnn: Convolutional neural networks for brain networks; towards predicting neurodevelopment. *NeuroImage*, 146:1038–1049, 2017.

[179] B. Kerleguer, C. Cannamela, and J. Garnier. A bayesian neural network approach to multi-fidelity surrogate modeling. *International Journal for Uncertainty Quantification*, 14(1), 2024.

[180] T. Kim, M. Hong, K. G. Bhatia, and G. SenGupta. Aeroelastic model reduction for affordable computational fluid dynamics-based flutter analysis. *AIAA journal*, 43(12):2487–2495, 2005.

[181] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[182] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[183] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. *5th International Conference on Learning Representations, ICLR 2017 - Conference Track Proceedings*, pages 1–14, 2017.

[184] B. F. Klose, C. Morsbach, M. Bergmann, E. J. Munoz Lopez, A. Hergt, and E. Kügeler. The unsteady shock–boundary layer interaction in a compressor cascade—part ii: High-fidelity simulation. *Journal of Turbomachinery*, 147(9), 2025.

[185] T. Koh and E. Powers. Second-order volterra filtering and its application to nonlinear system identification. *IEEE Transactions on acoustics, speech, and signal processing*, 33(6):1445–1455, 1985.

[186] P. Z. Korondi, M. Marchi, L. Parussini, and C. Poloni. Multi-fidelity design optimisation strategy under uncertainty with limited computational budget. *Optimization and Engineering*, 22(2):1039–1064, 2021.

[187] J. Kou and T. Xiao. Artificial intelligence and machine learning in aerodynamics. *Metascience in Aerospace*, 1(2):190–218, 2024.

[188] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60:84 – 90, 2012.

[189] J. N. Kutz. Deep learning in fluid dynamics. *Journal of Fluid Mechanics*, 814:1–4, 2017.

[190] J. N. Kutz, S. L. Brunton, B. W. Brunton, and J. L. Proctor. *Dynamic Mode Decomposition: Data-Driven Modeling of Complex Systems*. SIAM, Philadelphia, PA, 2016.

[191] B. Lakshminarayanan, A. Pritzel, and C. Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. *Advances in neural information processing systems*, 30, 2017.

[192] R. Landon. Naca 0012 oscillatory and transient pitching. *AGARD report*, 702:45–59, 1982.

[193] Larrañaga A, Brunton S L, Martínez Z, Chapela S, and Porteiro J. Data-driven prediction of the performance of enhanced surfaces from an extensive CFD-generated parametric search space. *Machine Learning: Science and Technology*, 4:025012, 4 2023.

[194] T. Lassila, A. Manzoni, A. Quarteroni, and G. Rozza. Model order reduction in fluid dynamics: challenges and perspectives. *Reduced Order Methods for modeling and computational reduction*, pages 235–273, 2014.

[195] B. Launder. Cfd for aerodynamic turbulent flows: progress and problems. *Aeronautical Journal*, 104(1036):353–365, 2000.

[196] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural Computation*, 1(4):541–551, 1989.

[197] B. H. K. Lee. Self-sustained shock oscillations on airfoils at transonic speeds. *Progress in Aerospace Sciences*, 37(2):147–196, 2001.

[198] J. Leishman. Indicial lift approximations for two-dimensional subsonic flow as obtained from oscillatory measurements. *Journal of Aircraft*, 30(3):340–351, 1993.

[199] J. G. Leishman and T. S. Beddoes. State-space model for unsteady airfoil behavior and dynamic stall. In *AIAA Aerospace Sciences Meeting*, number AIAA-1989-1319, 1989.

[200] M. Leschziner. Modelling turbulent separated flow in the context of aerodynamic applications. *Fluid dynamics research*, 38(2-3):174–210, 2006.

[201] D. Levin, K. K. Bastos, and E. H. Dowell. Convolution and volterra series approach to reduced-order modeling of unsteady aerodynamic loads. *AIAA Journal*, 60(3):1663–1678, 2022.

[202] H. Li and K. Ekici. Aeroelastic modeling of the agard 445.6 wing using the harmonic-balance-based one-shot method. In *AIAA Scitech 2019 Forum*. American Institute of Aeronautics and Astronautics Inc, AIAA, 2019.

[203] J. Li, Y. Li, T. Liu, D. Zhang, and Y. Xie. Multi-fidelity graph neural network for flow field data fusion of turbomachinery. *Energy*, 285:129405, 2023.

[204] J. Li, T. Liu, G. Zhu, Y. Li, and Y. Xie. Uncertainty quantification and aerodynamic robust optimization of turbomachinery based on graph learning methods. *Energy*, 273:127289, 2023.

[205] K. Li, J. Kou, and W. Zhang. Deep neural network for unsteady aerodynamic and aeroelastic modeling across multiple mach numbers. *Nonlinear Dynamics*, 96:2157–2177, 2019.

[206] S. Li, J. Qin, M. He, and R. Paoli. Fast evaluation of aircraft icing severity using machine learning based on XGBoost. *Aerospace*, 7(4), 4 2020.

[207] S. Li, Z. Sun, Y. Zhu, and C. Zhang. Physics-constrained and flow-field-message-informed graph neural network for solving unsteady compressible flows. *Physics of Fluids*, 36(4), 2024.

[208] T. Li, J. Yan, X. Chen, Z. Wang, Q. Zhang, E. Zhou, C. Gong, and J. Liu. Accelerating aerodynamic design optimization based on graph convolutional neural network. *International Journal of Modern Physics C*, 35(1):2450007–3433, 2024.

[209] Z. Li, F. Liu, W. Yang, S. Peng, and J. Zhou. A survey of convolutional neural networks: analysis, applications, and prospects. *IEEE transactions on neural networks and learning systems*, 2021.

[210] Z. Li, S. Zhang, H. Li, K. Tian, Z. Cheng, Y. Chen, and B. Wang. On-line transfer learning for multi-fidelity data fusion with ensemble of deep neural networks. *Advanced Engineering Informatics*, 53:101689, 2022.

[211] N. Lianrui and W. Ziniu. Matched volterra reduced-order model for an airfoil undergoing periodic translation, 2023.

[212] P. Liao, W. Song, P. Du, and H. Zhao. Multi-fidelity convolutional neural network surrogate model for aerodynamic optimization based on transfer learning. *Physics of Fluids*, 33(12), 2021.

[213] J. Ling, A. Kurzawski, and J. Templeton. Reynolds averaged turbulence modelling using deep neural networks with embedded invariance. *Journal of Fluid Mechanics*, 807:155–166, 2016.

[214] C. Liu, C. Xie, Y. Meng, and L. Bai. Experimental and numerical flutter analysis using local piston theory with viscous correction. *Aerospace*, 10(10):870, 2023.

[215] H. Liu and Y. Zhao. Efficient training data generation for reduced-order modeling in a transonic flight regime. *International Journal of Aerospace Engineering*, 2018(1):4083538, 2018.

[216] Y. Liu, S. Chen, F. Wang, and F. Xiong. Sequential optimization using multi-level cokriging and extended expected improvement criterion. *Structural and Multidisciplinary Optimization*, 58:1155–1173, 2018.

[217] L. Ljung. *System Identification: Theory for the User*. Prentice Hall, Upper Saddle River, NJ, 2 edition, 1999.

[218] S. M. R. Loghmanian, R. Yusof, and M. Khalid. Nonlinear dynamic system identification using volterra series: Multi-objective optimization approach. In *2011 Fourth International Conference on Modeling, Simulation and Applied Optimization*, pages 1–5. IEEE, 2011.

[219] D. J. Lucia, P. S. Beran, and W. A. Silva. Reduced-order modeling: new approaches for computational physics. *Progress in aerospace sciences*, 40(1-2):51–117, 2004.

[220] D. J. Lucia, P. S. Beran, and W. A. Silva. Reduced-order modeling, new approaches for computational physics. *Progress in Aerospace Sciences*, 40(1–2):51–117, 2004.

[221] H. Lyu, N. Sha, S. Qin, M. Yan, Y. Xie, and R. Wang. Advances in neural information processing systems. *Advances in neural information processing systems*, 32, 2019.

[222] D. J. C. MacKay. A Practical Bayesian Framework for Backpropagation Networks. *Neural Computation*, 4(3):448–472, 05 1992.

[223] B. Malouin, J.-Y. Trépanier, M. Gariépy, et al. Interpolation of transonic flows using a proper orthogonal decomposition method. *International Journal of Aerospace Engineering*, 2013, 2013.

[224] A. Mannarino and E. H. Dowell. Reduced-order models for computational-fluid-dynamics-based nonlinear aeroelastic problems. *Aiaa Journal*, 53(9):2671–2685, 2015.

[225] A. Mannarino and P. Mantegazza. Nonlinear aeroelastic reduced order modeling by recurrent neural networks. *Journal of Fluids and Structures*, 48:103–121, 2014.

[226] Z. Mao, A. D. Jagtap, and G. E. Karniadakis. Physics-informed neural networks for high-speed flows. *Computer Methods in Applied Mechanics and Engineering*, 360:112789, 2020.

[227] P. Marzocca, W. A. Silva, and L. Librescu. Nonlinear open-/closed-loop aeroelastic analysis of airfoils via volterra series. *AIAA journal*, 42(4):673–686, 2004.

[228] J. Masci, D. Boscaini, M. M. Bronstein, and P. Vandergheynst. Geodesic Convolutional Neural Networks on Riemannian Manifolds. *Proceedings of the IEEE International Conference on Computer Vision*, 2015-Febru:832–840, 2015.

[229] D. Massegur, D. Clifford, A. Da Ronch, R. Lombardi, and M. Panzeri. Low-dimensional models for aerofoil icing predictions. *Aerospace*, 10(5):444, 2023.

[230] D. Massegur and A. Da Ronch. Graph convolutional multi-mesh autoencoder for steady transonic aircraft aerodynamics. *Machine Learning: Science and Technology*, 5(2):025006, 6 2024.

[231] D. Massegur and A. Da Ronch. Recurrent graph convolutional multi-mesh autoencoder for unsteady transonic aerodynamics. *Journal of Fluids and Structures*, 131:104202, 2024.

[232] D. Massegur Sampietro and A. Da Ronch. Graph convolutional multi-mesh autoencoder for steady transonic aircraft aerodynamics. *Machine Learning: Science and Technology*, 2023.

[233] D. Massegur Sampietro and A. Da Ronch. Recurrent multi-mesh convolutional autoencoder framework for spatio-temporal aerodynamic modelling. In *AIAA AVIATION 2023 Forum*, page 3845, 2023.

[234] D. Massegur Sampietro, G. Immordino, A. Vaiuso, A. Da Ronch, and M. Righi. Graph-convolutional autoencoder frameworks for aerodynamic shape predictions of the agard wing. In *AIAA SCITECH 2025 Forum*, page 0885, 2025.

[235] S. F. McCormick. *Multigrid Methods*. Society for Industrial and Applied Mathematics, 1987.

[236] S. F. McCormick. *Multigrid methods*. SIAM, 1987.

[237] W. J. McCroskey. Unsteady airfoils. *Annual review of fluid mechanics*, 14(1):285–311, 1982.

[238] M. Meliani, N. Bartoli, T. Lefebvre, M.-A. Bouhlel, J. R. Martins, and J. Morlier. Multi-fidelity efficient global optimization: Methodology and application to airfoil shape design. In *AIAA aviation 2019 forum*, page 3236, 2019.

[239] X. Meng, H. Babaee, and G. E. Karniadakis. Multi-fidelity bayesian neural networks: Algorithms and applications. *Journal of Computational Physics*, 438:110361, 2021.

[240] T. Mitchell. Machine learning mcgraw-hill international. 1997.

[241] F. M. A. Mitrotta, A. Pirrera, T. Macquart, J. E. Cooper, A. Pereira do Prado, and P. Higino Cabral. Development of a nonlinear structural stability constraint for aeroelastic optimization. In *AIAA SCITECH 2024 Forum*, page 2412, 2024.

[242] A. Mohan, D. Daniel, M. Chertkov, and D. Livescu. Compressed convolutional lstm: An efficient deep learning framework to model high fidelity 3d turbulence. *arXiv preprint arXiv:1903.00033*, 2019.

[243] A. Moni, W. Yao, and H. Malekmohamadi. Data-driven reduced-order modeling for nonlinear aerodynamics using an autoencoder neural network. *Physics of Fluids*, 36(1), 2024.

[244] F. Monti, D. Boscaini, J. Masci, E. Rodolà, J. Svoboda, and M. M. Bronstein. Geometric deep learning on graphs and manifolds using mixture model CNNs. *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, 2017-Janua:5425–5434, 2017.

[245] H. Morino, H. Yamaguchi, T. Kumano, S. Jeong, and S. Obayashi. Efficient aeroelastic analysis using unstructured cfd method and reduced-order unsteady aerodynamic model. In *50th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference 17th AIAA/ASME/AHS Adaptive Structures Conference 11th AIAA No*, page 2326, 2009.

[246] B. Mufti, A. Bhaduri, S. Ghosh, L. Wang, and D. N. Mavris. Shock wave prediction in transonic flow fields using domain-informed probabilistic deep learning. *Physics of Fluids*, 36(1), 2024.

[247] B. Mufti, C. Perron, and D. N. Mavris. Nonlinear reduced-order modeling of compressible flow fields using deep learning and manifold learning. *Physics of Fluids*, 37(3), 2025.

[248] T. Nakamura-Zimmerer, M. T. Stringer, B. K. Colbert, and J. B. Scoggins. Structured covariance gaussian networks for orion crew module aerodynamic uncertainty quantification. In *AIAA SCITECH 2023 Forum*, page 1184, 2023.

[249] R. Neal. Bayesian learning via stochastic dynamics. *Advances in neural information processing systems*, 5, 1992.

[250] R. M. Neal. *Bayesian learning for neural networks*, volume 118. Springer Science & Business Media, 2012.

[251] M.-F. Negoita and M.-V. Hothazie. A machine learning-based approach for predicting aerodynamic coefficients using deep neural networks and cfd data. *INCAS Bulletin*, 16(4), 2024.

[252] T. Nguyen and R. Grishman. Graph convolutional networks with argument-aware pooling for event detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[253] F. Ogoke, K. Meidani, A. Hashemi, and A. B. Farimani. Graph convolutional networks applied to unstructured flow field data. *Machine Learning: Science and Technology*, 2(4):045020, 2021.

[254] M. Ohlberger and S. Rave. Reduced basis methods, a tutorial. *GAMM-Mitteilungen*, 38(1):101–123, 2015.

[255] N. Omata and S. Shirayama. A novel method of low-dimensional representation for temporal behavior of flow fields using deep autoencoder. *Aip Advances*, 9(1):015006, 2019.

[256] F. K. Owen and A. K. Owen. Measurement and assessment of wind tunnel flow quality. *Progress in aerospace sciences*, 44(5):315–348, 2008.

[257] B. Pacini, A. Yildirim, B. Davoudi, J. R. Martins, and K. Duraisamy. Towards efficient aerodynamic and aeroacoustic optimization for urban air mobility vehicle design. In *AIAA AVIATION 2021 FORUM*, page 3026, 2021.

[258] J. Parekh, P. Bekemeyer, S. Helm, D. G. François, and C. Grabe. Surrogate-based design space exploration and exploitation for efficient airfoil optimization under uncertainties using transition models. *Aerospace Science and Technology*, 154:109532, 2024.

[259] B. Parsonage and C. Maddock. A multi-fidelity model management framework for multi-objective aerospace design optimisation. *Frontiers in Aerospace Engineering*, 2:1046177, 2023.

[260] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, et al. Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830, 2011.

[261] A. Pedrioli, P. Capone, M. Righi, E. Garcia-Sanchez, L. Pinsard, and J. Vieira Gomes. Model-si: Modeling and simulation-multi-fidelity surrogate model of an evtol for certification. In *AIAA SCITECH 2024 Forum*, page 1624, 2024.

[262] B. Peherstorfer, P. S. Beran, and K. E. Willcox. Multifidelity monte carlo estimation for large-scale uncertainty propagation. In *2018 AIAA Non-Deterministic Approaches Conference*, page 1660, 2018.

[263] H. Pei, B. Wei, K. C.-C. Chang, Y. Lei, and B. Yang. Geom-gcn: Geometric graph convolutional networks. *arXiv preprint arXiv:2002.05287*, 2020.

[264] J.-Z. Peng, S. Chen, N. Aubry, Z.-H. Chen, and W.-T. Wu. Time-variant prediction of flow over an airfoil using deep neural network. *Physics of Fluids*, 32(12):123602, 2020.

[265] D. A. Peters, D. D. Boyd, and C. J. He. Finite-state induced-flow model for rotors in hover and forward flight. *Journal of the American Helicopter Society*, 34(4):5–17, 1989.

[266] T. Pfaff, M. Fortunato, A. Sanchez-Gonzalez, and P. W. Battaglia. Learning Mesh-Based Simulation with Graph Networks. 10 2020.

[267] T. Pfaff, M. Fortunato, A. Sanchez-Gonzalez, and P. W. Battaglia. Learning mesh-based simulation with graph networks. In *International Conference on Learning Representations*, 2021.

[268] V. Pham, M. Tyan, T. A. Nguyen, and J.-W. Lee. Extended hierarchical kriging method for aerodynamic model generation incorporating multiple low-fidelity datasets. *Aerospace*, 11(1):6, 2023.

[269] D. J. Piatak and C. S. Cleckner. Oscillating turntable for the measurement of unsteady aerodynamic phenomena. *Journal of aircraft*, 40(1):181–188, 2003.

[270] S. Portet. A primer on model selection using the akaike information criterion. *Infectious Disease Modelling*, 5:111–128, 2020.

[271] J. L. Proctor, S. L. Brunton, and J. N. Kutz. Dynamic mode decomposition with control. *SIAM Journal on Applied Dynamical Systems*, 15(1):142–161, 2016.

[272] J. L. Proctor, S. L. Brunton, and J. N. Kutz. Dynamic mode decomposition with control. *SIAM Journal on Applied Dynamical Systems*, 15(1):142–161, 2016.

[273] P. Z. G. Qian, H. Wu, and C. F. J. Wu. Gaussian process models for computer experiments with qualitative and quantitative factors. *Technometrics*, 50(3):383–396, 2008.

[274] J. Qin, H. Yu, and J. Wu. On the investigation of shock wave/boundary layer interaction with a high-order scheme based on lattice boltzmann flux solver. *Advances in Aerodynamics*, 6(1):6, 2024.

[275] J. Qiu, J. Tang, H. Ma, Y. Dong, K. Wang, and J. Tang. Deepinf: Social influence prediction with deep learning. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 2110–2119, 2018.

[276] G. Quaranta, P. Masarati, and P. Mantegazza. A conservative mesh-free approach for fluid structure problems in coupled problems. In *International conference for coupled problems in science and engineering, Santorini, Greece*, pages 24–27, 2005.

[277] S. Rahman. Wiener–hermite polynomial expansion for multivariate gaussian probability measures. *Journal of Mathematical Analysis and Applications*, 454(1):303–334, 2017.

[278] J. C. Rains, D. Huang, and Y. Wang. Residual dynamic mode decomposition with control for nonlinear aeroservoelastic applications. In *AIAA SciTech 2024 Forum*, page 2264, 2024.

[279] M. Raissi, P. Perdikaris, and G. E. Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, 2019.

[280] D. Rajaram, T. G. Puranik, A. Renganathan, W. J. Sung, O. J. Pinon-Fischer, D. N. Mavris, and A. Ramamurthy. Deep gaussian process enabled surrogate models for aerodynamic flows. In *AIAA Scitech 2020 Forum*, page 1640, 2020.

[281] H. Rakotonirina, P. Honeine, O. Atteia, and A. VAN EXEM. A generative deep neural network as an alternative to co-kriging. *Available at SSRN 4725658*, 2024.

[282] D. E. Raveh, Y. M. Yossef, and Y. Levy. Analyses for the second aeroelastic prediction workshop using the EZNSS code. In *AIAA Journal*, volume 56, pages 387–402. American Institute of Aeronautics and Astronautics Inc., 2018.

[283] S. A. Renganathan, R. Maulik, and V. Rao. Machine learning for nonintrusive model order reduction of the parametric inviscid transonic flow past an airfoil. *Physics of Fluids*, 32(4), 2020.

[284] A. H. Ribeiro, K. Tiels, J. Umenberger, T. B. Schön, and L. A. Aguirre. On the smoothness of nonlinear system identification. *Automatica*, 121:109158, 2020.

[285] M. D. Ribeiro, M. Stradtner, and P. Bekemeyer. Unsteady reduced order model with neural networks and flight-physics-based regularization for aerodynamic applications. *Computers & Fluids*, 264:105949, 2023.

[286] M. Righi and M. Berci. On the subsonic lift of elliptical wings: Generation of indicial functions via cfd simulations. *Journal of Aeroelasticity and Structural Dynamics*, 7(1), 2019.

[287] M. Righi, G. Coppotelli, G. Immordino, A. Da Ronch, L. Onofri, R. Sbarra, and G. Romano. Experimental characterization of flutter and lco of a very flexible wing. In *AIAA SCITECH 2024 Forum*, page 0831, 2024.

[288] M. Righi, S. Düzel, D. Anderegg, A. Da Ronch, D. Massegur Sampietro, I. Soukhmane, and G. Immordino. Rom-based uncertainties quantification of flutter speed prediction of the bscw wing. In *AIAA SCITECH 2022 Forum*, page 0179, 2022.

[289] M. Ripepi, M. J. Verveld, N. Karcher, T. Franz, M. Abu-Zurayk, S. Görtz, and T. Kier. Reduced-order models for aerodynamic applications, loads and mdo. *CEAS Aeronautical Journal*, 9(1):171–193, 2018.

[290] E. A. I. Roadmap. Easa concept paper: Guidance for level 1 and 2 machine learning applications. 2024.

[291] F. Romor, M. Tezzele, M. Mrosek, C. Othmer, and G. Rozza. Multi-fidelity data fusion through parameter space reduction with applications to automotive engineering. *International Journal for Numerical Methods in Engineering*, 124(23):5293–5311, 2023.

[292] M. Ross, M. T. Smith, and M. Álvarez. Learning nonparametric volterra kernels with gaussian processes. *Advances in Neural Information Processing Systems*, 34:24099–24110, 2021.

[293] C. W. Rowley and S. T. M. Dawson. Model reduction for flow analysis and control. *Annual Review of Fluid Mechanics*, 49:387–417, 2017.

[294] V. Rozov and C. Breitsamter. Data-driven prediction of unsteady pressure distributions based on deep learning. *Journal of Fluids and Structures*, 104:103316, 2021.

[295] S. H. Rudy, S. L. Brunton, J. L. Proctor, and J. N. Kutz. Data-driven discovery of partial differential equations. *Science Advances*, 3(4):e1602614, 2017.

[296] C. Sabater and S. Görtz. Gradient-based aerodynamic robust optimization using the adjoint method and gaussian processes. *Advances in Evolutionary and Deterministic Methods for Design, Optimization and Control in Engineering and Sciences*, pages 211–226, 2021.

[297] C. Sabater, P. Stürmer, and P. Bekemeyer. Fast predictions of aircraft aerodynamics using deep-learning techniques. *AIAA Journal*, pages 1–13, 2022.

[298] E. Saetta, R. Tognaccini, and G. Iaccarino. Machine learning to predict aerodynamic stall. *International Journal of Computational Fluid Dynamics*, 36(7):641–654, 2022.

[299] H. Salehinejad, S. Sankar, J. Barfett, E. Colak, and S. Valaee. Recent advances in recurrent neural networks. *arXiv preprint arXiv:1801.01078*, 2017.

[300] H. Salimbeni and M. Deisenroth. Doubly stochastic variational inference for deep gaussian processes. *Advances in neural information processing systems*, 30, 2017.

[301] A. L. Samuel. Some studies in machine learning using the game of checkers. ii—recent progress. *IBM Journal of research and development*, 11(6):601–617, 1967.

[302] A. Sanchez-Gonzalez, J. Godwin, T. Pfaff, R. Ying, J. Leskovec, and P. Battaglia. Learning to simulate complex physics with graph networks. In *International conference on machine learning*, pages 8459–8468. PMLR, 2020.

[303] A. Sanchez-Gonzalez, J. Godwin, T. Pfaff, R. Ying, J. Leskovec, and P. W. Battaglia. Learning to Simulate Complex Physics with Graph Networks. Technical report, 2020.

[304] A. Sanchez-Gonzalez, N. Heess, J. T. Springenberg, J. Merel, M. Riedmiller, R. Hadsell, and P. Battaglia. Graph Networks as Learnable Physics Engines for Inference and Control. Technical report, 2018.

[305] A. Savino. Dust user manual, 2024. `https://public.gitlab.polimi.it/DAER/dust/-/blob/master/doc/DUST_user_manual.pdf`.

[306] P. J. Schmid. Dynamic mode decomposition of numerical and experimental data. *Journal of Fluid Mechanics*, 656:5–28, 2010.

[307] P. J. Schmid. Dynamic mode decomposition of numerical and experimental data. *Journal of fluid mechanics*, 656:5–28, 2010.

[308] P. J. Schmid. Dynamic mode decomposition and its variants. *Annual Review of Fluid Mechanics*, 54:225–254, 2022.

[309] V. Schmitt. Pressure distributions on the onera m6-wing at transonic mach numbers, experimental data base for computer program assessment. *AGARD AR-138*, 1979.

[310] D. M. Schuster, D. D. Liu, and L. J. Huttsell. Computational aeroelasticity: success, progress, challenge. *Journal of Aircraft*, 40(5):843–856, 2003.

[311] J. B. Scoggins, T. J. Wignall, T. Nakamura-Zimmerer, and K. L. Bibb. Multihierarchy gaussian process models for probabilistic aerodynamic databases using uncertain nominal and off-nominal configuration data. In *AIAA SCITECH 2023 Forum*, page 1185, 2023.

[312] M. Seeger. Gaussian processes for machine learning. *International journal of neural systems*, 14(02):69–106, 2004.

[313] A. Segalini and P. H. Alfredsson. A simplified vortex model of propeller and wind-turbine wakes. *Journal of Fluid Mechanics*, 725:91–116, 2013.

[314] Y. Seo, M. Defferrard, P. Vandergheynst, and X. Bresson. Structured sequence modeling with graph convolutional recurrent networks. In *Neural Information Processing: 25th International Conference, ICONIP 2018, Siem Reap, Cambodia, December 13-16, 2018, Proceedings, Part I 25*, pages 362–373. Springer, 2018.

[315] A. Sharma, A. Gandhi, and A. Kumar. Exploring multi-fidelity bayesian neural network for nuclear data evaluation. In *Proceedings of the DAE Symp. on Nucl. Phys*, volume 66, page 1218, 2022.

[316] M. Sharma, P. Kushwaha, P. Kumari, P. Kumari, and R. Yadav. Machine learning techniques in data fusion: A review. In *International Conference on Communication and Intelligent Systems*, pages 391–405. Springer, 2022.

[317] S. W. Shaw and C. Pierre. Modal analysis-based reduced-order models for nonlinear structures&mdash; an invariant manifold approach. *The shock and vibration digest*, 31(1):3–16, 1999.

[318] X. SHI, Z. Chen, H. Wang, D.-Y. Yeung, W.-k. Wong, and W.-c. WOO. Convolutional lstm network: A machine learning approach for precipitation nowcasting. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015.

[319] S. B. Shiki, C. Hansen, and S. d. Silva. Practical applications for nonlinear system identification using discrete-time volterra series. *Journal of the Brazilian Society of Mechanical Sciences and Engineering*, 45(2):87, 2023.

[320] W. SILVA. A methodology for using nonlinear aerodynamics in aeroservoelastic analysis and design. In *32nd Structures, Structural Dynamics, and Materials Conference*, page 1110, 1991.

[321] W. Silva. Reduced-order models based on linear and nonlinear aerodynamic impulse responses. In *40th Structures, Structural Dynamics, and Materials Conference and Exhibit*, page 1262, 1999.

[322] L. Sirovich. Turbulence and the dynamics of coherent structures. part i. coherent structures. *Quarterly of Applied Mathematics*, 45(3):561–571, 1987.

[323] J. Snoek, H. Larochelle, and R. P. Adams. Practical bayesian optimization of machine learning algorithms. *Advances in neural information processing systems*, 25, 2012.

[324] S. Snyder, T. J. Wignall, J. S. Green, S. Kumar, M. W. Lee, T. Nakamura-Zimmerer, J. B. Scoggins, and R. A. Williams. Aerofusion: Data fusion and uncertainty quantification for entry vehicles. In *AIAA SCITECH 2023 Forum*, page 1182, 2023.

[325] J. Song, Y. Yu, and D. Huang. Parametrized global linearization models for flutter prediction. In *AIAA SCITECH 2024 Forum*, page 2266, 2024.

[326] S. R. Stahlschmidt, B. Ulfenborg, and J. Synnergren. Multimodal deep learning for biomedical data fusion: a review. *Briefings in Bioinformatics*, 23(2):bbab569, 2022.

[327] B. K. Stanford and S. J. Massey. Uncertainty quantification of the fun3d-predicted nasa crm flutter boundary. In *SciTech Forum*, number NF1676L-24499, 2017.

[328] B. K. Stanford and S. Roy. Sizing and topology design of an aeroelastic wingbox under uncertainty. In *International Forum on Aeroelasticity and Structural Dynamics*, number NF1676L-31616, 2019.

[329] B. Sudret. Global sensitivity analysis using polynomial chaos expansions. *Reliability engineering & system safety*, 93(7):964–979, 2008.

[330] D. Sun, Z. Wang, F. Qu, and J. Bai. A deep learning based prediction approach for the supercritical airfoil at transonic speeds. *Physics of Fluids*, 33(8):086109, 2021.

[331] Y. Sun, U. Sengupta, and M. Juniper. Physics-informed deep learning for simultaneous surrogate modeling and PDE-constrained optimization of an airfoil geometry. *Computer Methods in Applied Mechanics and Engineering*, 411, 6 2023.

[332] P. A. Surve, P. Ramu, and D. Ghate. A multi-fidelity aeroelastic optimization of an aircraft wing using co-kriging. In *National Conference on Multidisciplinary Analysis and Optimization*, pages 57–63. Springer, 2021.

[333] B. Swaminathan, J. G. Manathara, and A. Vinayagam. Application of dynamic mode decomposition with control (dmdc) for aircraft parameter estimation. *IFAC-PapersOnLine*, 55(1):789–794, 2022.

[334] S. B. Taieb, G. Bontempi, A. F. Atiya, and A. Sorjamaa. A review and comparison of strategies for multi-step ahead time series forecasting based on the nn5 forecasting competition. *Expert systems with applications*, 39(8):7067–7083, 2012.

[335] S. B. Taieb, R. J. Hyndman, et al. *Recursive and direct multi-step forecasting: the best of both worlds*, volume 19. Department of Econometrics and Business Statistics, Monash Univ., 2012.

[336] K. Taira, S. L. Brunton, S. T. Dawson, C. W. Rowley, T. Colonius, B. J. McKeon, O. T. Schmidt, S. Gordeyev, V. Theofilis, and L. S. Ukeiley. Modal analysis of fluid flows: An overview. *Aiaa Journal*, 55(12):4013–4041, 2017.

[337] K. Taira, M. S. Hemati, S. L. Brunton, Y. Sun, K. Duraisamy, S. Bagheri, S. T. Dawson, and C.-A. Yeh. Modal analysis of fluid flows: Applications and outlook. *AIAA journal*, 58(3):998–1022, 2020.

[338] N. Takeishi, Y. Kawahara, and T. Yairi. Subspace dynamic mode decomposition for stochastic koopman analysis. *Physical Review E*, 96(3):033310, 2017.

[339] S. Tamura and T. Yumitori. Accurate aeroelastic analysis of the wing structure with control surface. In *AIAA SCITECH 2024 Forum*, page 2446, 2024.

[340] B.-T. Tan, K. E. Willcox, and M. Damodaran. Applications of proper orthogonal decomposition for inviscid transonic aerodynamics. 2003.

[341] Y. Tao. *Uncertainty Quantification in Large Scale Systems Design*. PhD thesis, Carleton University, 2022.

[342] J. D. Taylor and D. F. Hunsaker. Low-fidelity method for rapid aerostructural optimisation and design-space exploration of planar wings. *The Aeronautical Journal*, 125(1289):1209–1230, 2021.

[343] N. Thuerey, K. Weißenow, L. Prantl, and X. Hu. Deep learning methods for reynolds-averaged navier-stokes simulations of airfoil flows. *arXiv*, 2018.

[344] H. Tijdeman. On the motion of shock waves on an airfoil with oscillating flap. In *Symposium Transsonicum II*, pages 49–56. Springer, 1976.

[345] S. Timme and K. J. Badcock. Global instability of wing shock-buffet onset. *Journal of Fluid Mechanics*, 885:A27, 2020.

[346] E. Tipton. Stratified sampling using cluster analysis: A sample selection strategy for improved generalizations from experiments. *Evaluation review*, 37(2):109–139, 2013.

[347] A. Torregrosa, L. García-Cuevas, P. Quintero, and A. Cremades. On the application of artificial neural network for the development of a nonlinear aeroelastic model. *Aerospace Science and Technology*, 115:106845, 2021.

[348] D. Tran, M. Dusenberry, M. Van Der Wilk, and D. Hafner. Bayesian layers: A module for neural network uncertainty. *Advances in neural information processing systems*, 32, 2019.

[349] P. Tsilifis, P. Pandita, S. Ghosh, V. Andreoli, T. Vandeputte, and L. Wang. Bayesian learning of orthogonal embeddings for multi-fidelity gaussian processes. *Computer Methods in Applied Mechanics and Engineering*, 386:114147, 2021.

[350] J. H. Tu. *Dynamic mode decomposition: Theory and applications*. PhD thesis, Princeton University, 2013.

[351] J. H. Tu, C. W. Rowley, and J. N. Kutz. On dynamic mode decomposition, theory and applications. *Journal of Computational Dynamics*, 1(2):391–421, 2014.

[352] M. Tugnoli, D. Montagnani, M. Syal, G. Droandi, and A. Zanotti. Mid-fidelity approach to aerodynamic simulations of unconventional vtol aircraft configurations. *Aerospace Science and Technology*, 115:106804, 2021.

[353] K. Vafa. *Training and inference for deep Gaussian processes*. PhD thesis, 2016.

[354] A. Vaiuso, G. Immordino, M. Righi, and A. Da Ronch. Multi-fidelity transonic aerodynamic loads estimation using bayesian neural networks with transfer learning. *Aerospace Science and Technology*, 163:110301, 2025.

[355] J. Vassberg, E. Tinoco, M. Mani, O. Brodersen, B. Eisfeld, R. Wahls, J. Morrison, T. Zickuhr, K. Laflin, and D. Mavriplis. Summary of dlr-f6 wing-body data from the third aiaa cfd drag prediction workshop. *RTO AVT-147 Paper*, 57, 2007.

[356] J. C. Vassberg, E. N. Tinoco, M. Mani, O. P. Brodersen, B. Eisfeld, R. A. Wahls, J. H. Morrison, T. Zickuhr, K. R. Laflin, and D. J. Mavriplis. Abridged summary of the third aiaa computational fluid dynamics drag prediction workshop. *Journal of aircraft*, 45(3):781–798, 2008.

[357] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[358] F. Vetrano, F. Mastroddi, and R. Ohayon. POD approach for unsteady aerodynamic model updating. *CEAS Aeronautical Journal*, 6(1):121–136, 2015.

[359] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*, pages 1096–1103, 2008.

[360] G. A. Vio, G. Dimitriadis, J. E. Cooper, K. J. Badcock, M. A. Woodgate, and A. M. Rampurawala. Aeroelastic system identification using transonic cfd data for a wing/store configuration. *Aerospace Science and Technology*, 11(2–3):146–154, 2007.

[361] V. Volterra. *Sopra le funzioni che dipendono da altre funzioni*. Tipografia della R. Accademia dei Lincei, 1887.

[362] H. Wan, M. Ghoreyshi, and J. Seidel. Analysis of flow past an aircraft at high angle of attack based on mode decomposition. In *AIAA AVIATION 2023 Forum*, page 3946, 2023.

[363] C. Wang, X. Qiang, M. Xu, and T. Wu. Recent advances in surrogate modeling methods for uncertainty quantification and propagation. *Symmetry*, 14(6):1219, 2022.

[364] H. Wang, Y. Cao, Z. Huang, Y. Liu, P. Hu, X. Luo, Z. Song, W. Zhao, J. Liu, J. Sun, et al. Recent advances on machine learning for computational fluid dynamics: A survey. *arXiv preprint arXiv:2408.12171*, 2024.

[365] J. Wang, G. Huang, W. Lu, and P. E. Sullivan. Dynamic mode decomposition analysis of flow separation in a diffuser to inform flow control strategies. *Journal of Fluids Engineering*, 142(2):021502, 2020.

[366] Q. Wang, C. E. Cesnik, and K. Fidkowski. Multivariate recurrent neural network models for scalar and distribution predictions in unsteady aerodynamics. In *AIAA Scitech 2020 Forum*, page 1533, 2020.

[367] X. Wang, J. Kou, and W. Zhang. A new dynamic stall prediction framework based on symbiosis of experimental and simulation data. *Physics of Fluids*, 33(12):127119, 2021.

[368] Z. Wang, Z. Zhang, D. Lee, P. Chen, D. Liu, and M. Mignolet. Flutter analysis with structural uncertainty by using cfd-based aerodynamic rom. In *49th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference, 16th AIAA/ASME/AHS Adaptive Structures Conference, 10th AIAA Non-Deterministic Approaches Conference, 9th AIAA Gossamer Spacecraft Forum, 4th AIAA Multidisciplinary Design Optimization Specialists Conference*, page 2197, 2008.

[369] Z. Wei, A. Yang, J. Li, M. Bauerheim, R. P. Liem, and P. Fua. Deepgeo: Deep geometric mapping for automated and effective parameterization in aerodynamic shape optimization. In *AIAA AVIATION FORUM AND ASCEND 2024*, page 3839, 2024.

[370] A. Weiner and R. Semaan. Robust dynamic mode decomposition methodology for an airfoil undergoing transonic shock buffet. *AIAA Journal*, 61(10):4456–4467, 2023.

[371] J. Wen, W. Zhu, X. Jia, F. Ma, and Q. Liu. Spectral domain graph convolutional deep neural network for predicting unsteady and nonlinear flows. *Physics of Fluids*, 35(9), 2023.

[372] N. Wiener. The homogeneous chaos. *American Journal of Mathematics*, 60(4):897–936, 1938.

[373] D. N. Wilke. Multifidelity surrogate models: A new data fusion perspective. *arXiv preprint arXiv:2404.14456*, 2024.

[374] M. O. Williams, I. G. Kevrekidis, and C. W. Rowley. A data–driven approximation of the koopman operator: Extending dynamic mode decomposition. *Journal of Nonlinear Science*, 25:1307–1346, 2015.

[375] A. G. Wilson, D. A. Knowles, and Z. Ghahramani. Gaussian process regression networks. *arXiv preprint arXiv:1110.4411*, 2011.

[376] M. Winter and C. Breitsamter. *Reduced-order modeling of unsteady aerodynamic loads using radial basis function neural networks*. Deutsche Gesellschaft für Luft-und Raumfahrt-Lilienthal-Oberth eV, 2014.

[377] B. Y. Wong and B. C. Khoo. Inductive transfer-learning of high-fidelity aerodynamics from inviscid panel methods. *Advances in Aerodynamics*, 7(1):1, 2025.

[378] K. Worden, G. Manson, and G. Tomlinson. A harmonic probing algorithm for the multi-input volterra series. *Journal of Sound and Vibration*, 201(1):67–84, 1997.

[379] J. Wray and G. G. Green. Calculation of the volterra kernels of non-linear dynamic systems using an artificial neural network. *Biological cybernetics*, 71(3):187–195, 1994.

[380] H. WU, X. LIU, W. AN, and H. LYU. A generative deep learning framework for airfoil flow field prediction with sparse data. *Chinese Journal of Aeronautics*, 35(1):470–484, 1 2022.

[381] M.-Y. Wu, X.-J. He, X.-H. Sun, T.-S. Tong, Z.-H. Chen, and C. Zheng. Efficient aerodynamic shape optimization using transfer learning based multi-fidelity deep neural network. *Physics of Fluids*, 36(11):116109, 2024.

[382] X. Wu, W. Zhang, and S. Song. Robust aerodynamic shape design based on an adaptive stochastic optimization framework. *Structural and Multidisciplinary Optimization*, 57:639–651, 2018.

[383] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and S. Y. Philip. A comprehensive survey on graph neural networks. *IEEE transactions on neural networks and learning systems*, 32(1):4–24, 2020.

[384] S. Xiang, F. Nie, and C. Zhang. Learning a mahalanobis distance metric for data clustering and classification. *Pattern recognition*, 41(12):3600–3612, 2008.

[385] H.-R. Xie, Z.-Q. Wang, Y.-B. Li, Q. Li, W.-T. Wu, J.-L. Han, J.-Z. Peng, and Y. He. Fast spatiotemporal sequence graph convolutional network-based transient flow prediction around different airfoils. *Physics of Fluids*, 36(10), 2024.

[386] D. Xiu and G. E. Karniadakis. The wiener–askey polynomial chaos for stochastic differential equations. *SIAM journal on scientific computing*, 24(2):619–644, 2002.

[387] B. Xu, T. Wang, Y. Yuan, Z. Zhao, and H. Liu. A simplified free vortex wake model of wind turbines for axial steady conditions. *Applied Sciences*, 8(6):866, 2018.

[388] L. Xu, G. Zhou, F. Zhao, Z. Guo, and K. Zhang. A data-driven reduced order modeling for fluid flow analysis based on series forecasting intelligent algorithm. *IEEE Access*, 10:60163–60176, 2022.

[389] W. Yamazaki and D. J. Mavriplis. Derivative-enhanced variable fidelity surrogate modeling for aerodynamic functions. *AIAA journal*, 51(1):126–137, 2013.

[390] L. Yang and A. Shami. On hyperparameter optimization of machine learning algorithms: Theory and practice. *Neurocomputing*, 415:295–316, 2020.

[391] W. Yao and M.-S. Liou. A nonlinear modeling approach using weighted piecewise series and its applications to predict unsteady flows. *Journal of Computational Physics*, 318:58–84, 2016.

[392] X. Yao, R. Huang, H. Hu, and H. Liu. Transonic aerodynamic–structural coupling characteristics predicted by nonlinear data-driven modeling approach. *AIAA Journal*, 62(3):1159–1178, 2024.

[393] E. C. Yates Jr. NASA Technical Memorandum I00492. Technical report, NASA, Hampton, 8 1987.

[394] S. Yazdani and M. Tahani. Data-driven discovery of turbulent flow equations using physics-informed neural networks. *Physics of Fluids*, 36(3), 2024.

[395] R. Yondo, E. Andrés, and E. Valero. A review on design of experiments and surrogate models in aircraft real-time and many-query aerodynamic analyses. *Progress in Aerospace Sciences*, 96:23–61, 2018.

[396] B. Yu, H. Yin, and Z. Zhu. Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting. *arXiv preprint arXiv:1709.04875*, 2017.

[397] R. Zahn, M. Winter, M. Zieher, and C. Breitsamter. Application of a long short-term memory neural network for modeling transonic buffet aerodynamics. *Aerospace Science and Technology*, 113:106652, 2021.

[398] M. J. Zahr and C. Farhat. Progressive construction of a parametric reduced-order model for pde-constrained optimization. *International Journal for Numerical Methods in Engineering*, 102(5):1111–1135, 2015.

[399] B. W. Zan, Z. H. Han, C. Z. Xu, M. Q. Liu, and W. Z. Wang. High-dimensional aerodynamic data modeling using a machine learning method based on a convolutional neural network. *Advances in Aerodynamics*, 4(1), 12 2022.

[400] C. Zhang, H. Chen, X. Xu, Y. Duan, and G. Wang. Aerodynamic shape optimization using a physics-informed hot-start method combined with modified metric-based proper orthogonal decomposition. *Physics of Fluids*, 36(8), 2024.

[401] W. Zhang, P. Xuhao, K. Jiaqing, and W. Xu. Heterogeneous data-driven aerodynamic modeling based on physical feature embedding, 2024.

[402] X. Zhang, F. Xie, T. Ji, Z. Zhu, and Y. Zheng. Multi-fidelity deep neural network surrogate model for aerodynamic shape optimization. *Computer Methods in Applied Mechanics and Engineering*, 373:113485, 2021.

[403] Y. Zhang. *Efficient uncertainty quantification in aerospace analysis and design*. Missouri University of Science and Technology, 2013.

[404] Y. Zhang, S. Tao, W. Chen, and D. W. Apley. A latent variable approach to gaussian process modeling with qualitative and quantitative factors. *Technometrics*, 62(3):291–302, 2020.

[405] Y. Zhang, Z. Wang, and Z.-H. Han. A parallel variable-fidelity algorithm for efficient constrained multi-objective aerodynamic design optimization. *Physics of Fluids*, 36(8), 2024.

[406] Y. Zhang, D. Zhang, and H. Jiang. Review of challenges and opportunities in turbulence modeling: A comparative analysis of data-driven machine learning approaches. *Journal of Marine Science and Engineering*, 11(7):1440, 2023.

[407] Z. Zhang, P. Cui, and W. Zhu. Deep learning on graphs: A survey. *IEEE Transactions on Knowledge and Data Engineering*, 34(1):249–270, 2020.

[408] H. Zhao, Z.-H. Gao, and L. Xia. Efficient aerodynamic analysis and optimization under uncertainty using multi-fidelity polynomial chaos-kriging surrogate model. *Computers & Fluids*, 246:105643, 2022.

[409] Q. Zhao, X. Han, R. Guo, and C. Chen. A computationally efficient hybrid neural network architecture for porous media: Integrating cnns and gnns for improved permeability prediction. *arXiv preprint arXiv:2311.06418*, 2023.

[410] W. Zhao, L. Lv, J. Zhao, W. Xiao, J. Chen, and X. Wu. Uncertainty quantification for multidimensional correlated flow field responses. *Journal of Verification, Validation and Uncertainty Quantification*, 9(2), 2024.

[411] S. Zhiwei, W. Chen, Y. Zheng, B. Junqiang, L. Zheng, X. Qiang, and F. Qiujun. Non-intrusive reduced-order model for predicting transonic flow with varying geometries. *Chinese Journal of Aeronautics*, 33(2):508–519, 2020.

[412] H. Zhou, F. Xie, T. Ji, X. Zhang, C. Zheng, and Y. Zheng. Fast transonic flow prediction enables efficient aerodynamic design. *Physics of Fluids*, 35(2):026109, 2023.

[413] J. Zhou, G. Cui, S. Hu, Z. Zhang, C. Yang, Z. Liu, L. Wang, C. Li, and M. Sun. Graph neural networks: A review of methods and applications. *AI open*, 1:57–81, 2020.

[414] F. Zhuang, Z. Qi, K. Duan, D. Xi, Y. Zhu, H. Zhu, H. Xiong, and Q. He. A comprehensive survey on transfer learning. *Proceedings of the IEEE*, 109(1):43–76, 2020.

[415] R. Zimmermann and S. Görtz. Improved extrapolation of steady turbulent aerodynamics using a non-linear pod-based reduced order model. *The Aeronautical Journal*, 116(1184):1079–1100, 2012.

[416] R. Zimmermann, A. Vendl, and S. Görtz. Reduced-order modeling of steady flows subject to aerodynamic constraints. *AIAA journal*, 52(2):255–266, 2014.

[417] D. Zügner, A. Akbarnejad, and S. Günnemann. Adversarial attacks on neural networks for graph data. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 2847–2856, 2018.

[418] K. Zuo, Z. Ye, S. Bu, X. Yuan, and W. Zhang. Fast simulation of airfoil flow field via deep neural network. *Aerospace Science and Technology*, 150:109207, 2024.