Titre: Title:	AI-Accelerated Predictions of Aerodynamic Coefficients for Helicopter Blades
Auteur: Author:	Simon Paquette-Greenbaum
Date:	2025
Type:	Mémoire ou thèse / Dissertation or Thesis
Référence: Citation:	Paquette-Greenbaum, S. (2025). AI-Accelerated Predictions of Aerodynamic Coefficients for Helicopter Blades [Mémoire de maîtrise, Polytechnique Montréal]. PolyPublie. https://publications.polymtl.ca/65705/

URL de PolyPublie: PolyPublie URL:	https://publications.polymtl.ca/65705/
Directeurs de recherche: Advisors:	David Vidal, & Éric Laurendeau
Programme: Program:	Génie mécanique

POLYTECHNIQUE MONTRÉAL
affiliée à l'Université de Montréal

AI-Accelerated Predictions of Aerodynamic Coefficients for Helicopter Blades

SIMON PAQUETTE-GREENBAUM
Département de génie mécanique

Mémoire présenté en vue de l'obtention du diplôme de *Maîtrise ès sciences appliquées*
Génie mécanique

Mai 2025

POLYTECHNIQUE MONTRÉAL
affiliée à l'Université de Montréal

Ce mémoire intitulé :

AI-Accelerated Predictions of Aerodynamic Coefficients for Helicopter Blades

présenté par **Simon PAQUETTE-GREENBAUM**
en vue de l'obtention du diplôme de *Maîtrise ès sciences appliquées*
a été dûment accepté par le jury d'examen constitué de :

Sébastien LECLAIRE, président
David VIDAL, membre et directeur de recherche
Éric LAURENDEAU, membre et codirecteur de recherche
Youssef DIOUANE, membre

DEDICATION

To my family

ACKNOWLEDGEMENTS

This work was supported by CAE Inc., the Natural Sciences and Engineering Research Council of Canada (NSERC), and the Consortium for Research and Innovation in Aerospace in Quebec (CRIAQ) *PARTENAR-IA* program. Compute resources were provided by the Digital Research Alliance of Canada and Calcul Québec.

RÉSUMÉ

Les simulateurs de vol s'appuient fréquemment sur des méthodes de simulation de giravions de fidélité moyenne qui nécessitent un couplage non linéaire avec des tables de recherche. La génération de ces tables contenant les coefficients aérodynamiques de portance (C_L), de traînée (C_D) et de moment (C_M) est souvent d'un coût prohibitif, nécessitant des simulations coûteuses en termes de calcul et plusieurs processus de travail complexes. Ce coût est contourné frequemment en interpolant entre des points de données, au détriment d'une erreur d'interpolation. Cette recherche présente le développement d'un modèle de substitution basé sur des réseaux neuronaux capables de prédictions discrètes, précises et rapides des coefficients aérodynamiques, éliminant effectivement cette erreur. Ce modèle est conçu pour se généraliser à l'ensemble des géométries et des conditions d'écoulement environnantes qu'un rotor pourrait rencontrer, y compris les régimes standard, inversé, de décrochage et transsonique. Premièrement, ce travail décrit le processus de génération d'une base de données de Mécanique des Fluides Numérique (CFD) à haute-fidélité. Les plages de variables sont définies pour couvrir l'intégralité de l'espace de l'écoulement : angle d'attaque de $-180°$ à $180°$, Mach de 0.2 à 0.9, et Reynolds de 2.5×10^6 à 20.0×10^6. De plus, l'espace des profils de rotor est exploré à l'aide de profils générés par des B-Splines rationnelles non uniformes. La base de données CFD est générée grâce à un cadre de simulation robuste utilisant le solveur CFD de volumes finis CHAMPS. Deuxièmement, le processus d'entraînement du modèle IA est décrit en détail. L'architecture du modèle repose sur un perceptron multicouche, avec une stratégie complète de prétraitement des données incluant une réduction de dimensionnalité via l'Analyse en Composantes Principales. En outre, l'algorithme "Tree-structured Parzen Estimator" est utilisé pour naviguer l'espace des hyperparamètres architecturaux, aboutissant à des modèles à la fois précis en termes de prédiction et efficaces en termes de calcul. Enfin, les modèles obtenus sont évalués sur des bases de données *test* dédiées. Sur un jeu de test comprenant 10% de points échantillonnés aléatoirement à partir du domaine d'entraînement, mais exclus pendant l'apprentissage, le meilleur modèle atteint un coefficient de détermination (R^2) de 0.9962 et une erreur absolue moyenne (MAE) de 0.0302 pour C_L, $R^2 = 0.9987$ et MAE = 0.0115 pour C_D, et $R^2 = 0.9958$ avec MAE = 0.0128 pour C_M. Ces résultats montrent que le modèle se généralise bien à la fois à des géométries et à des conditions d'écoulement inédites. Le modèle le plus performant est également testé sur des cas plus complexes afin de tenir compte de ces limites. Enfin, l'accélération produite par ce modèle par rapport à la CFD conventionnelles est mesurée, où les prédictions du meilleur modèle nécessitent environ 0.54×10^9 fois moins de ressources de calcul.

ABSTRACT

Flight simulators and aerodynamic design optimization frequently rely on medium-fidelity rotorcraft simulation methods, which often necessitate nonlinear coupling with lookup table databases. The generation of these lookup tables containing lift (C_L), drag (C_D), and moment (C_M) aerodynamic coefficients is often prohibitively expensive, requiring expensive high-fidelity simulations and several complex workflows. This cost is often circumvented by interpolating between lookup table data points at the expense of an incurred interpolation error. This work presents the development of a surrogate model based on neural networks capable of discrete, accurate, and rapid predictions of aerodynamic coefficients, effectively eliminating the interpolation error typically associated with lookup tables. This model is designed to generalize across the full state space of possible geometries and surrounding flow conditions a rotor might experience, including standard, reversed, stall, and transonic. Firstly, this work outlines the process of generating a high-fidelity Computational Fluid Dynamics (CFD) database. Relevant variable ranges are defined to cover the full state space of flow conditions: angle of attack from $-180°$ to $180°$, Mach number from 0.2 to 0.9, and Reynolds number from 2.5×10^6 to 20.0×10^6. In addition, the design space of rotor blade-section profiles is explored using complex airfoils generated via Non-Uniform Rational B-Splines. The dataset of high-fidelity CFD data points is generated with a robust simulation framework using the CHAMPS finite volume CFD solver. Secondly, the development and training process of the AI-based surrogate model is described in detail. The Multi-Layer Perceptron architecture forms the basis of the model, with a comprehensive data pre-processing strategy that includes dimensionality reduction using Principal Component Analysis. Furthermore, the Tree-structured Parzen Estimator algorithm is employed to efficiently navigate the architectural hyperparameter space, leading to models that are both accurate in prediction and computationally efficient. Finally, resulting models are evaluated on dedicated testing datasets. On a dataset consisting of 10% randomly sampled points from the training domain, held out during training, the best-performing model achieves a coefficient of determination (R^2) of 0.9962 and a Mean Absolute Error (MAE) of 0.0302 for C_L predictions, $R^2 = 0.9987$ and MAE = 0.0115 for C_D, and $R^2 = 0.9958$ with MAE = 0.0128 for C_M. This demonstrates that the model generalizes well to both unseen geometries and flow conditions. The best-performing model is also tested on more complex cases to address this framework's limitations. Lastly, the acceleration produced by this model with respect to conventional CFD simulations is measured, where the surrogate predictions of the best model require an estimated 0.54×10^9 times fewer computational resources.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ACRONYMS AND SYMBOLS

Acronyms

AI	Artificial Intelligence
CFD	Computational Fluid Dynamics
CFL	Courant–Friedrichs–Lewy
CHAMPS	CHApel Multi-Physics Simulation
CNN	Convolutional Neural Networks
CPU	Central Processing Unit
DoE	Design of Experiments
EI	Expected Improvement
ELU	Exponential Linear Unit
EMA	Exponential Moving Average
FEM	Finite Element Method
FLOP	Floating-Point Operations
FVM	Finite Volume Method
GPU	Graphics Processing Unit
GPR	Gaussian Process Regression
HPC	High-Performance Computing
ISA	International Standard Atmosphere
LHS	Latin Hypercube Sampling
LU-SGS	Lower-Upper Symmetric Gauss-Seidel
MAE	Mean Absolute Error
Mila	Montreal Institute for Learning Algorithms
ML	Machine Learning
MLP	Multi-Layer Perceptron
MPI	Message Passing Interface
MSE	Mean Squared Error
MUSCL	Monotonic Upstream-centered Scheme for Conservation Laws
NaN	Not a Number
NN	Neural-Networks
NURBS	Non-Uniform Rational B-Splines
NL-VLM	Non-Linear Vortex Lattice Method
OLC	Outer Limit Cycle

PCA	Principal Component Analysis
PI	Probability of Improvement
POD	Proper Orthogonal Decomposition
RANS	Reynolds-Averaged Navier–Stokes
ReLU	Rectified Linear Unit
RMSE	Root Mean Square Error
RMS	Root Mean Square
SA	Spalart-Allmaras
SFD	Selective Frequency Damping
TPE	Tree-structured Parzen Estimator
VTOL	Vertical Take-Off and Landing
WLS	Weighted Least Squares

Symbols

α	Angle of attack
α	ELU scale hyperparameter
α	Scaling factor for AdEMAMAMix $\boldsymbol{m_2}$ moment
β_1	Decay coefficient for AdEMAMAMix $\boldsymbol{m_1}$ moment
β_2	Decay coefficient for AdEMAMAMix $\boldsymbol{m_2}$ moment
β_3	Decay coefficient for AdEMAMAMix $\boldsymbol{\nu}$ moment
β_W	Relative wind inflow angle
c	Geometry chord length
C^2	Geometry curvature continuity
C_D	Drag coefficient
$C_{D_{CFD}}$	CHAMPS CFD drag coefficient prediction
$C_{D_{ML}}$	ML surrogate model drag coefficient prediction
$\bar{C}_{D_{ML}}$	Mean ML surrogate model drag coefficient prediction
C_f	Skin friction coefficient
C_i	Simulation convergence condition
C_L	Lift coefficient
$C_{L_{CFD}}$	CHAMPS CFD lift coefficient prediction
$C_{L_{ML}}$	ML surrogate model lift coefficient prediction
$\bar{C}_{L_{ML}}$	Mean ML surrogate model lift coefficient prediction
C_M	Moment coefficient
$C_{M_{CFD}}$	CHAMPS CFD moment coefficient prediction

$C_{M_{ML}}$	ML surrogate model moment coefficient prediction
$\bar{C}_{M_{ML}}$	Mean ML surrogate model moment coefficient prediction
$\text{CONV}_{\%}$	Simulation convergence rate
C_x	Aerodynamic force coefficient in x-direction
D	Drag force
δ	Entropy correction coefficient
Δs	Grid cell normal wall-spacing
Δs_{min}	Minimum grid cell normal wall-spacing
e^x	Exponential function
ϵ	Infinitesimally small positive quantity
η	Learning rate
$f(\alpha)$	Custom probability distribution for angle of attack
$f(x)$	Multi-objective function
g	Gravitational acceleration near Earth's surface
$\boldsymbol{g}$	Parameter-wise loss function gradient
$g(x)$	Lower quantile hyperparameter kernel density estimator
γ	Hyperparameter quantile threshold
γ	Specific heat ratio for air
h	Altitude
i	Grid node index along wall surface
j	Grid node index normal to wall surface
$J_c(x)$	Complexity optimization objective
$J_l(x)$	Loss optimization objective
k	Number of eigenvalues
K	Venkatakrishnan's limiter coefficient
l	Number of levels per dimension
L	ISA tropospheric lapse rate
L	Lift force
$\mathcal{L}$	Loss function
L^2	Euclidean norm
L_i	Layer
$\mathcal{L}_{\text{val}}$	Validation loss
$l(x)$	Upper quantile hyperparameter kernel density estimator
λ	Weight decay factor
M	Mach number
M	Pitching moment

$\boldsymbol{m_1}$	First AdEMAMAMix mean gradient moment
$\hat{\boldsymbol{m}}_\mathbf{1}$	Corrected first AdEMAMAMix mean gradient moment
$\boldsymbol{m_2}$	Second AdEMAMAMix mean gradient moment
M_R	Resulting Mach number
M_{tip}	Rotor tip Mach number
M_W	Relative wind Mach number
M_x	Chord tangent Mach number component
M_y	Chord normal Mach number component
μ	Mean
μ_{ref}	Sutherland's reference viscosity for air
μ_∞	Freestream dynamic viscosity
n	Number of Geometry Cartesian coordinates
N	Number of DoE dimensions
N	Number of grid cells
N	Stochastic batch size
N_{con}	Number of neural connections
N_i	Number of input components
N_l	Number of hidden layers
N_n	Number of neurons in first hidden layer
$\boldsymbol{\nu}$	AdEMAMAMix gradient variance moment
$\hat{\boldsymbol{\nu}}$	Corrected AdEMAMAMix gradient variance moment
ν	Kinematic viscosity
ν_t	Eddy viscosity
ω	Relaxation factor
P_i	NURBS control point
ϕ	Activation function
q_∞	Freestream dynamic pressure
R	Specific gas constant
Re	Reynolds number
R^2	Coefficient of determination
Re_x	Reynolds number at x-distance from flat-plate leading-edge
ρ	Density
ρ_0	Density for air at sea-level
ρ_∞	Freestream density
S	Geometry surface area
S	Sutherland's constant

σ	Standard deviation
Σ	Sum
T	Temperature
T_0	ISA sea-level atmospheric temperature
T_∞	Freestream temperature
T_{ref}	Sutherland's reference temperature for air
τ_{wall}	Wall shear stress
$\boldsymbol{\theta}$	Model parameter
U_∞	Freestream velocity
u_τ	Friction velocity
$\mathbf{w}$	NURBS control point weight
w_i	Weight
$\mathbf{W}_k$	Matrix of k eigenvectors
$\mathbf{x}$	Input vector
$\mathbf{x}$	NURBS control point Cartesian x-coordinate
$\mathbf{X}$	Input dataset
x'	Normalized input
x^*	Proposed set of hyperparameters
x_i	Geometry Cartesian x-coordinate
x_i	Input
$\hat{\mathbf{X}}_k$	PCA transformed input dataset with k components
x_{max}	Feature-wise input maximum
x_{min}	Feature-wise input minimum
x_{opt}	Optimal set of hyperparameters
y	Output
$\mathbf{y}$	NURBS control point Cartesian y-coordinate
$\mathbf{y}$	Output vector
$\mathbf{Y}$	Output dataset
y^+	Dimensionless wall distance
y'	Normalized output
y_i	Geometry Cartesian y-coordinate
$\hat{y}_i$	Predicted output
y_i	Real output
y_{wall}	Dimensional wall distance

LIST OF APPENDICES

CHAPTER 1 INTRODUCTION

1.1 Context

1.1.1 Urban Flight

The need for urban flight is becoming more apparent as the human species continues to congregate in urban centers and metropolitan areas which continue to become increasingly dense and distributed vertically. This trend highlights a growing need for vertical transportation options capable of navigating tightly packed cities. Vertical Take-Off and Landing (VTOL) aircraft appear to be an ideal candidate for this task as they have the ability to take off from and land on the limited area available on the roofs of tall buildings. VTOL aircraft could consist of drones and air taxis, which would serve the humans of tomorrow by reducing road congestion and by shortening the time required to transport humans and goods alike.

While the concept of urban flight through VTOL aircraft seems very promising, several challenges must be overcome before it becomes a widespread reality. Safety, regulatory frameworks, infrastructure, and public opinion are but a few examples. The technological limitations that plague urban flight simulators are also a massive challenge that prevents this concept from coming to fruition. Urban flight can only become a reality if pilots are trained using urban VTOL flight simulators that perform adequately.

Accurately simulating the physics of a VTOL aircraft during urban flight remains a significant technological challenge. Real-time and sufficiently accurate calculations of aerodynamics are required to achieve the high level of realism and responsiveness needed for pilot training. Typically, classical Computational Fluid Dynamics (CFD) methods are used to predict urban airflow, the physical flight response of aircraft, the forces acting on aircraft, etc. However, further advancements in CFD are needed to meet the specific requirements of a full-fledged flight simulator for VTOL aircraft placed in an complex urban environment.

1.1.2 Limitations of Classical Methods

CFD is a branch of fluid mechanics that enables engineers and scientists to model fluid flow using numerical methods and computational techniques. By solving governing equations numerically, CFD provides valuable predictions of a fluid's behavior, including its flow patterns, structural interactions, and properties, given specific initial and boundary conditions. Widely applied across numerous research areas, CFD has become an indispensable tool in engineer-

ing disciplines such as aerodynamics, hydrodynamics, thermoacoustics, and environmental sciences, among others.

In the case of the discipline of aerodynamics, CFD is indispensable in the prediction of aerodynamic dimensionless coefficients, which serve in the design of cars, aircraft, and wind turbines, to name a few. Aerodynamic dimensionless coefficients are used to quantify the forces and moments acting on a structure given its surrounding flow environment. For example, the Lift Coefficient (C_L), the Drag Coefficient (C_D), and the Moment Coefficient (C_M) coefficients can be used in the performance prediction of vehicles placed in a fluid flow.

Despite its widespread utility, CFD has notable limitations, particularly in computational complexity. Classical CFD techniques, such as the Finite Volume Method (FVM) and Finite Element Method (FEM), often come with high computational costs, making them unsuitable for all applications. In the context of flight simulators, these methods are impractical due to the computational demands of domain discretization, which leads to large systems of equations that must be solved iteratively. Current computational constraints make real-time simulations challenging, limiting the feasibility of classical CFD for such applications.

Hence, given the complexity of simulating the 3D aerodynamics of VTOL aircraft, medium-fidelity aerodynamic simulation methods, such as the Non-Linear Vortex Lattice Method (NL-VLM), are commonly used in rotorcraft early-stage design optimization and flight simulators. However, methods like NL-VLM require coupling with non-linear databases to adequately represent complex and non-linear flow phenomena.

1.1.3 HAMAC Project

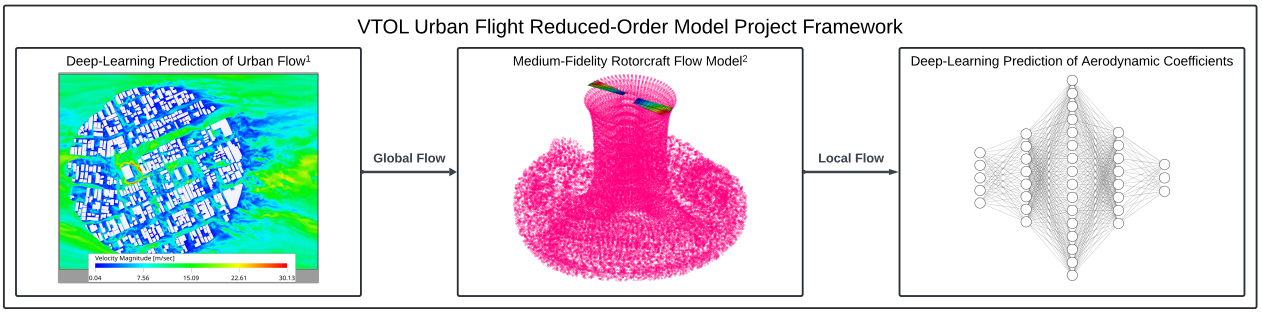


Figure 1.1 HAMAC project framework [1] [2]

The research and development presented in this memoir are conducted within the framework of the *Multi-scale Aerodynamic Modelling of Helicopters/UAVs in Urban Environments* (HAMAC) project. This project aims to develop solutions that enhance the capacity of flight

simulators for urban air mobility through the coupling of rotorcraft models and urban city aerodynamic models. As illustrated in Figure 1.1, the project is divided into three main components. While the first two components fall outside the scope of this memoir, the third serves as its primary focus.

The first component, urban airflow solutions, will be developed using Machine Learning (ML) and Artificial Intelligence (AI) techniques to enable rapid predictions of city flow fields in previously unseen urban environments. The second component focuses on constructing a medium-fidelity multi-rotor and fuselage rotorcraft model using NL-VLM. Finally, the third component involves the non-linear data coupling of the aforementioned rotorcraft model.

Table 1.1 C81 Table Example

Airfoil, Re	m = number of α samples , n = number of M samples		
	M_1	$\cdots$	M_n
α_1	$C_{L_{1,1}}$	$\cdots$	$C_{L_{1,n}}$
$\vdots$	$\vdots$		$\vdots$
α_m	$C_{L_{m,1}}$	$\cdots$	$C_{L_{m,n}}$
α_1	$C_{D_{1,1}}$	$\cdots$	$C_{D_{1,n}}$
$\vdots$	$\vdots$		$\vdots$
α_m	$C_{D_{m,1}}$	$\cdots$	$C_{D_{m,n}}$
α_1	$C_{M_{1,1}}$	$\cdots$	$C_{M_{1,n}}$
$\vdots$	$\vdots$		$\vdots$
α_m	$C_{M_{m,1}}$	$\cdots$	$C_{M_{m,n}}$

This memoir focuses on the development of a surrogate model to replace traditional lookup tables used for non-linear coupling in medium-fidelity rotorcraft simulations. In the rotorcraft community, lookup tables, typically formatted in the C81 style, as outlined in Table 1.1, serve as a standard approach for aerodynamic data storage. Each C81 table corresponds to a specific airfoil and Reynolds number (Re), providing C_L, C_D, and C_M coefficients across a full-factorial combination of Mach number (M) and angle of attack (α) samples. Generating these tables is computationally expensive due to the factorial nature of parameter sampling and the

high cost of obtaining aerodynamic coefficients from high-fidelity CFD simulations. Moreover, medium-fidelity rotorcraft simulations require extensive databases of C81 tables to account for the wide range of Reynolds numbers encountered and the geometric complexity of rotor blades. As a result, creating these databases is often prohibitively expensive. While linear interpolation is commonly used to approximate missing data points and reduce computational demands, it comes at the cost of reduced solution accuracy.

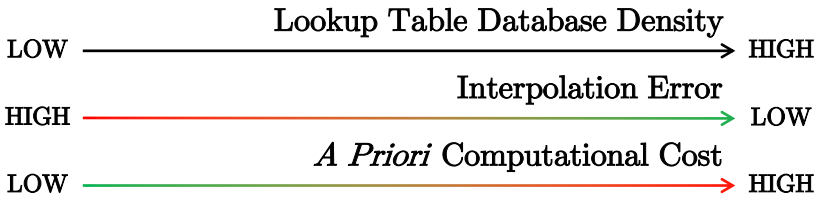


Figure 1.2 Trade-offs in prior generation of lookup tables

Figure 1.2 illustrates the trade-offs researchers and engineers must consider when generating lookup tables for medium-fidelity rotorcraft simulations. Increasing the density of the lookup table database, by incorporating tables for different geometries and Reynolds numbers, as well as including more Mach number and angle of attack samples, reduces linear interpolation error. However, this improvement comes at the cost of significantly higher *a priori* computational expenses.

1.2 Goal

The research presented in this memoir focuses on the development an AI-based surrogate model capable of near real-time predictions of aerodynamic coefficients for a VTOL rotorcraft across all flight conditions. As highlighted in the subsequent literature review, AI-based surrogate models have been successfully applied to similar problems. The proposed model aims to replace the lookup table databases traditionally used in medium-fidelity rotorcraft simulations, eliminating interpolation errors by directly predicting aerodynamic coefficients at each rotor section based on actual geometry and surrounding flow conditions.

However, this research must address several key challenges before possible flight simulator integration. First, computational efficiency is critical to ensure real-time responsiveness, essential for realistic pilot training. Second, predictive accuracy must be carefully managed, as the surrogate model inherently introduces some loss of accuracy compared to the high-fidelity CFD data stored in lookup tables. Finally, the model must demonstrate strong generalization capabilities, enabling it to handle unseen geometries and flow conditions effectively.

CHAPTER 2 LITERATURE REVIEW

This literature review highlights the need for AI-based surrogate modeling of rotorcraft blades and identifies the scientific gap it aims to address. By examining the benefits and drawbacks of medium-fidelity rotorcraft simulation methods that rely on lookup tables, the importance of developing surrogate models as a viable replacement for these tables is made evident. Additionally, and more importantly, the review demonstrates a clear scientific gap in the application of surrogate models for rotorcraft, as is evidenced by a lack of previous studies in the field that address the full state-space surrogate modeling of rotorcraft.

2.1 Medium-Fidelity Aerodynamic Simulation Methods

Medium-fidelity aerodynamic simulation methods that make use of lookup tables are commonplace in early-stage design optimization and flight simulators. Their implementation of lookup tables serves as a means of increasing simulation fidelity while limiting their real-time simulation cost. Although the computational cost associated with these lookup tables can be large, their *a priori* generation shifts the significant computational burden to a pre-simulation phase. This approach eliminates the need for real-time computation of complex non-linear phenomena, thereby enabling rapid medium-fidelity simulations.

Parenteau et al. (2018) [3] presented the NL-VLM, a method that makes usage of non-linear coupling with lookup tables for sectional wing data. In this case, the usage of non-linear coupling with sectional lookup tables produced results with comparable fidelity to a 3D Reynolds-Averaged Navier–Stokes (RANS) method requiring a much greater computational costs. Similarly, Proulx-Cabana et al. (2022) [4] and Cocco et al. (2024) [5] extended the research effort into rotorcraft applications by coupling viscous aerodynamic data with the potential unsteady vortex lattice method. The unsteady coupling in these proposed methods permits increased fidelity in rotorcraft applications compared to classical uncoupled methods such as lifting lines, surface panels, or linear vortex lattice methods.

The critical importance of high-quality lookup table data was made evident in a parametric study conducted by Proulx-Cabana et al. (2024) [6]. The study highlighted the necessity of high-quality lookup table data, noting substantial differences between simulation predictions made with differing quality databases. The study further reinforces the notion of the necessity of non-linear coupling, as in all but the most uncommon of evaluations, the uncoupled linear model consistently underperformed compared to the coupled method, irrespective of the

selected non-linear database.

While these medium-fidelity methods represent an immense reduction in computational time and expenses with respect to their higher-fidelity counterparts, the generation of their required lookup tables can be equally expensive and can, in certain instances, exceed the cost of higher-fidelity methods that do not require the *a priori* generation of databases. In order to avoid excessive computational expenses, researchers have used linear interpolation between lookup table data points. In situations with complex geometries or wildly varying flow conditions, this becomes a less viable solution as the quality of the solution increasingly degrades in function of the interpolation between data points. Furthermore, linear interpolation becomes impossible in aerodynamic design optimization. In the rotorcraft community, the most common form of lookup table is in the C81 style, that features a unique geometry's aerodynamic coefficients for full-factorial combinations of flow conditions. Geometry optimization processes render previously generated lookup tables useless as geometries are continuously updated.

In this context, the utility of surrogate modeling is evident in enabling rapid predictions of aerodynamic coefficients, particularly in scenarios where generating lookup tables using conventional CFD methods is prohibitively costly. The following section reviews literary works focused on surrogate models applied in this domain, specifically on those utilizing supervised learning techniques.

2.2 Supervised Learning of Aerodynamic Coefficients

2.2.1 Historical Perspective

While not the first, Greenman and Roth (1999) [7], within the NASA Ames Research Center, conducted one of the earliest serious attempts at surrogate modeling for aerodynamic coefficient predictions. Numerical CFD data was generated using a 2D Navier-Stokes solver and was subsequently used in a simplistic but effective artificial neural network. Given the time's limitations in computational resources, the generated dataset was very limited in size, containing only 27 data configurations. The study succeeded in developing surrogate models trained on sparse data with algorithms that would be considered outdated within the modern field of machine learning. While the study found the Levenberg-Marquardt training scheme [8] to be optimal, modern knowledge in the field of machine learning [9] has made the backpropagation training algorithm paired with modern optimization schemes the *de facto* choice for researchers and developers alike. Nevertheless, utilizing this surrogate model for relevant shape optimization tasks resulted in an 83% reduction in computational

cost, while only permitting a prediction error of 0.5%. Despite not having access to modern computational resources and machine learning algorithms, the developed surrogate models were capable of rapid and accurate predictions of aerodynamic coefficients. However, these technological constraints still restricted the complexity of the developed surrogate model, confining it to only a limited number of input dimensions.

Papila et al. (1999) [10] conducted further analysis of the predictive capabilities of neural network-based surrogate models for aerodynamic applications. By comparing neural network and polynomial-based techniques, the advantages of machine learning based algorithms become more apparent. While both approaches exhibited comparable performance on modestly sized datasets between 9 to 25 data points, neural network architectures demonstrated significantly greater potential on larger datasets of up to 765 data points. Furthermore, the neural network excelled at fitting sparse datasets, demonstrating the method's interpolative capabilities. This study, despite being an early exploration of the technique, effectively illustrated the advantages of neural network-based surrogate models over traditional response surface methods for predicting aerodynamic coefficients. However, similar to Greenman and Roth (1999) [7], the tested models were constrained by the number of input dimensions. Specifically, the developed models were limited to just two input dimensions, highlighting the technological constraints of the era in applying this methodology effectively.

The impact of dataset density was also analyzed within the NASA Ames Research Center by Rajkumar and Bardina (2003) [11]. The study defined critical characteristics required for effective neural network training datasets. Training datasets must depict all tendencies found within the problem domain and the statistical variance in inputs must be adequately represented across the entirety of the dataset's range. The importance of sufficient data density was demonstrated in this study, where models trained on exceedingly sparse datasets performed poorly with respect to models trained on numerous data points. Additionally, the study examined the performance of various activation functions, concluding that architectures utilizing the sigmoidal function outperformed those employing linear or hyperbolic tangent functions. The Rectified Linear Unit (ReLU) function, that is widely used in modern machine learning architectures, was not tested in this study as it did not gain widespread prominence in the field before the Glorot et al. (2011) [12] paper was published.

Khurana et al. (2008) [13] investigated the application of artificial neural networks as surrogate models for airfoil shape optimization. The proposed surrogate model demonstrated the capability to efficiently predict C_L and was trained on a dataset of 3000 PARSEC [14] generated airfoils under fixed flow conditions, with a Mach number of 0.35 and a Reynolds number of 3M. This surrogate model acted as a replacement for direct numerical optimiza-

tion's inefficient and computationally expensive workflow, where on a generalization testing dataset, a Coefficient of Determination (R^2) of 0.98 and an average of 12.46% error percentage were measured on the best performing model that was presented. This neural network configuration did, however, suffer from a maximum measured error of 294%. From a modern standpoint, unoptimal machine learning methods were selected. The Levenberg-Marquardt training scheme and sigmoid based activation functions were selected, as was the case with similar literary works of this era. Furthermore, the choice of PARSEC as geometry parameterization function was later shown in Masters et al. (2017) [15] to be suboptimal in contexts of shape optimization, where inverse optimization procedures were found to be rarely successful when parameterized with PARSEC design variables.

These early studies highlighted the potential of neural networks as a supervised learning method within the context of aerodynamic applications, specifically the prediction of aerodynamic coefficients. They demonstrated that neural networks have the ability to outperform classical response surface methods in this application. Furthermore, neural network and machine learning algorithms were shown to be capable of fitting sparse datasets. However, the research also highlighted that training on insufficiently dense datasets results in significant model underperformance. While these findings highlight the promise of neural networks, they also reveal limitations of the time, such as the high cost of aerodynamic simulation data, constrained computational resources, and the absence of modern machine learning techniques, such as the Adam optimizer introduced by Kingma and Ba (2014) [16]. As a result, the development of models was largely restricted to simplistic, low-dimensional applications.

2.2.2 Modern Surrogate Modeling Effort

The significant advancement of computational resources throughout the 2010s, generally aligning with Moore's Law (1965) [17], greatly enhanced researchers' capabilities in generating large and complex CFD simulation-based datasets. Simultaneously, the widespread dissemination of modern machine learning algorithms during the same decade, brought forward by foundational works such as Goodfellow et al. (2016) [18], propelled the development of surrogate models for aerodynamic applications to unprecedented levels. The availability of large aerodynamic datasets, combined with modern learning algorithms and architectures, have enabled researchers to develop surrogate models capable of real-world practicality and applicability, even in solving high-dimensional problems.

Multi-fidelity aerodynamic data was used in the development of a deep neural network modeling approach presented by He et al. (2020) [19]. In this study, data fusion techniques were employed as an effective means to generate larger datasets while reducing computational

costs. Both neural network and kriging [20] models were trained on sparse multi-fidelity datasets, which consisted of a relatively large quantity of low-fidelity Euler data and a much smaller set of high-fidelity Navier-Stokes CFD data points. The resulting neural network model significantly outperformed kriging models, achieving an order of magnitude lower prediction error for C_L and C_D coefficients when evaluated on a high-fidelity Navier-Stokes testing dataset. This study highlighted the potential of machine learning methods in capturing complex, non-linear relationships within sparse datasets, surpassing traditional response surface modeling techniques like kriging. However, the study's scope was limited to a single modified NACA 0012 geometry and a narrow input parameter range, restricting its broader applicability.

Convolutional Neural Networks (CNN), a type of machine learning architecture specifically designed to process grid-like or pixelated data structures, were utilized by Chen et al. (2020) [21] predict aerodynamic coefficients C_L, C_D, and C_M. The predictions were based on images of airfoils convolved with the flow conditions: angle of attack and Mach number. The CNN model was trained on a Navier-Stokes dataset consisting of 4200 samples generated from a full-factorial combination of 300 airfoil images, a single Reynolds number, 3 Mach numbers, and 5 angles of attack. The development of the CNN model used an airfoil image resolution of 85×85 pixels. Furthermore, the performance of the CNN was compared to that of a Multi-Layer Perceptron (MLP) architecture, which used vectorized airfoil data at a reduced resolution of 16×16 pixels due to memory constraints. The performance was found to be only slightly better with the CNN architecture. While CNN architecture might have a better prediction performance capability in this configuration, it may remain the case if proper airfoil parameterization techniques were utilized. Furthermore, the study did not compare the computational time required for predictions between the MLP and CNN architectures, focusing instead on a comparison between CNN and CFD predictions. On a dataset composed of 840 test samples, the CNN's prediction time was 0.96 s, whereas for CFD, one evaluation typically required 6 minutes, and while this may represent an immense acceleration, its speed would still be much slower than MLP architectures designed for the same application.

In a study performed by Balla et al. (2021) [22], the effectiveness of artificial neural networks was further tested. Artificial neural networks evaluated in comparison to Proper Orthogonal Decomposition (POD), another algorithm for model order reduction. POD was described as aerodynamic design industrialists' most popular surrogate modeling technique in a review survey published by Yondo et al. in (2018) [23]. This review further described the current challenges in surrogate modeling with POD, being the curse of dimensionality, i.e., creating reduced order models with a large number of inputs, and dealing with the non-linearity of complex aerodynamic functions. Balla et al. (2021) [22] demonstrated that artificial neural

networks outperformed POD in modeling sparse and complex aerodynamic datasets. This study decomposed several 2D CFD problem domains with the aid of Latin Hypercube Sampling (LHS): a dataset for the RAE2822 airfoil at various inflow conditions and a dataset of geometrically parameterized airfoils evaluated under a single inflow condition. In both cases, artificial neural networks outperformed POD in prediction accuracy, with only minimal errors. This remained the case across all evaluated dataset densities, exemplifying the neural network architecture's capacity to fit sparse data effectively. Notably, the artificial neural network models required only a small number of neurons and hidden layers, making them computationally efficient. However, a limitation of the study was its focus on models that addressed either inflow conditions or airfoil geometry parameters, but not both simultaneously. This constraint restricted the models to low-dimensional input spaces, leaving the performance of artificial neural networks in higher-dimensional aerodynamic problems untested.

Du et al. (2021) [24] demonstrated the extent to which surrogate models based on artificial neural networks can produce accurate predictions of aerodynamic performance coefficients. Separate, relatively large MLP architectures were employed to predict C_L and C_D coefficients for subsonic and transonic regimes, using geometry parameters, angle of attack, Mach number, and Reynolds number as inputs. These MLP surrogate models featured multiple layers with varying numbers of neurons and diverse activation functions. The datasets used in this study were derived from 2D aerodynamic solutions generated with a mesh size of 171,904 cells and converged to a relative l^2 residual of 10^{-12} using a high-order turbulent CFD solver. For efficiency, the problem domain was decomposed using Latin Hypercube Sampling (LHS) for independent variables and Gaussian copula methods [25] for dependent variables. In cases where verification accuracy was insufficient, LHS infill sampling was applied to refine the training datasets. This modeling approach resulted in surrogate models capable of state-of-the-art accuracy in predicting C_L and C_D. However, the development of these surrogates required substantial computational resources, with the subsonic and transonic models respectively needing 45,696 and 39,505 CFD training samples. While the specific computational cost of generating these datasets was not disclosed, it is likely to have been prohibitively expensive, making this modeling strategy impractical for most researchers.

These studies demonstrated the capacity of artificial neural networks' capacity to rapidly and accurately predict aerodynamic performance coefficients. They demonstrated that advances in modern computational resources and machine learning techniques have enabled artificial neural networks to fit complex datasets that represent challenging problem domains. Traditional response surface methods, such as kriging and POD, were shown to underperform in capturing complex, non-linear aerodynamic behaviors, particularly in contexts with

high-dimensional parameter spaces. Additionally, effective strategies for problem domain decomposition were explored, particularly LHS, which was found to be an effective means of defining sparse datasets. While these studies show the promising aspects of artificial neural networks, most applications were limited to lower-dimensional input spaces, and in cases where higher-dimensional inputs were used, the models required computationally expensive datasets with extremely high data density.

2.2.3 Surrogate Modeling for Rotorcraft Application

Kriging, also known as Gaussian Process Regression (GPR), was employed as a surrogate modeling technique for rotorcraft lookup tables in a study conducted by Sridharan and Sinsay (2023) [26]. The objective was to develop surrogate models capable of predicting aerodynamic coefficients across a wide range of geometries and flow conditions, covering angle of attack values from -5° to 15° and Mach number values between 0.3 and 0.9. The aerodynamic dataset for this study was generated using a 251×131 moderately fine mesh paired with *ARC2D*, a turbulent, high-fidelity Navier-Stokes CFD solver developed at NASA. A total of 70,021 data points were used to train multiple GPR-based surrogate models, with each model predicting either C_L, C_D, or C_M coefficients for one specific combination of inflow conditions. This solution demonstrated the ability to deliver accurate and rapid predictions of aerodynamic coefficients. However, regions of the dataset with pruned, unconverged training data resulted in relatively larger prediction errors. Furthermore, while the study successfully modeled a complex aerodynamic problem domain, 256 surrogate models were used in the representation of the full-factorial Design of Experiments (DoE), illustrating this approach's inefficiency with respect to modern machine learning algorithms. The usage of several surrogate models simplified their respective inputs' dimensional spaces.

In a similar fashion, Cornelius and Schmitz (2024) [27] employed fully-connected neural networks to fit a complex aerodynamic dataset for surrogate modeling of rotorcraft lookup tables. The dataset comprised full-factorial combinations of various NACA 4-Series airfoils, angles of attack ranging from -20° to 15°, Mach numbers between 0.25 and 0.9, and Reynolds numbers spanning 75k to 8M. Generating this aerodynamic database required approximately 1M CPU hours of computation and resulted in 62,000 data points obtained using 325,000-cell meshes and the NASA *OVERFLOW* CFD code, a high-fidelity turbulent solver with a fourth-order accurate spatial scheme. The neural network surrogate model trained on this dataset made simultaneous predictions of C_L, C_D, and C_M with low error and prediction accuracy surpassing multi-dimensional linear interpolation. The developed surrogate model represents an immense computational acceleration with respect to conventional CFD predictions. How-

ever, its development required a computationally expensive full-factorial DoE. The extremely dense, full-factorial, dataset rendered the neural network's interpolative capabilities under-utilized, with linear interpolation methods exhibiting comparable accuracy. This scenario should never occur when dealing with complex, non-linear aerodynamic datasets lacking extreme data density. Furthermore, the exclusive use of NACA 4-Series eliminated the need for complex geometric parameterization techniques. While this simplified model development, it also limited the generalization capabilities of the surrogate model for airfoils outside the NACA 4-Series family.

While these studies successfully demonstrated machine learning methods utility in surrogate modeling for rotorcraft lookup tables, they did not model across the entire range of possible aerodynamic flow conditions a rotorcraft blade-section might experience. Lookup tables used in rotorcraft simulations typically span across angle of attack ranges of -180° to 180°. Additionally, the reviewed studies only accounted for relatively small Reynolds numbers, whereas in reality, Reynolds numbers can exceed 20×10^6 in extreme cases for corresponding chord lengths of $c = 1$ m. Furthermore, the studies also made use of full-factorial DoE, resulting in extreme data density across small dimensional spaces. Hence, the full interpolative capabilities of machine learning algorithms were left unused due to this extreme data density. Full-factorial DoE quickly become unfeasible with a large number of design dimensions. For a study with a space of N dimensions, each having l levels, the complexity will increase by a factor of $O(l^N)$, making it unfeasible for complex problems such as rotorcraft simulation and optimization.

2.3 Neural Networks

As previously highlighted, artificial neural networks have proven to be highly effective in surrogate modeling for predicting aerodynamic coefficients. Previous studies have demonstrated their ability to model highly complex and non-linear physics applications and have outperformed other forms of classical response surface methods. Furthermore, neural networks have been shown to be effective with sparse and large databases.

Artificial neural networks are a machine learning architecture designed to emulate biological neural functions [28]. They are employed to learn from data, identify essential patterns, and make inferences on unseen inputs based on prior experience. Much like their biological counterparts, artificial neural networks exhibit a remarkable ability to generalize across different patterns, enabling them to apply learned knowledge to new scenarios.

The basic unit of a neural network [29], illustrated in Figure 2.1, serves as the building block of artificial neural networks and is analogous to a biological neuron. This unit consists of multiple inputs, each associated with a corresponding weight, a summation function that computes the weighted sum of the inputs, a bias parameter (not included in Figure 2.1) that shifts the summation, and an activation function that applies a non-linear transformation to the summation. Modeled after the biological neuron, the inputs correspond to dendrites, branches that receive signals from other neurons; the weights represent synapses, which are the junction between neurons where signals of differing strengths are transmitted; and the output corresponds to the axon, which transmits processed signals to other neurons.

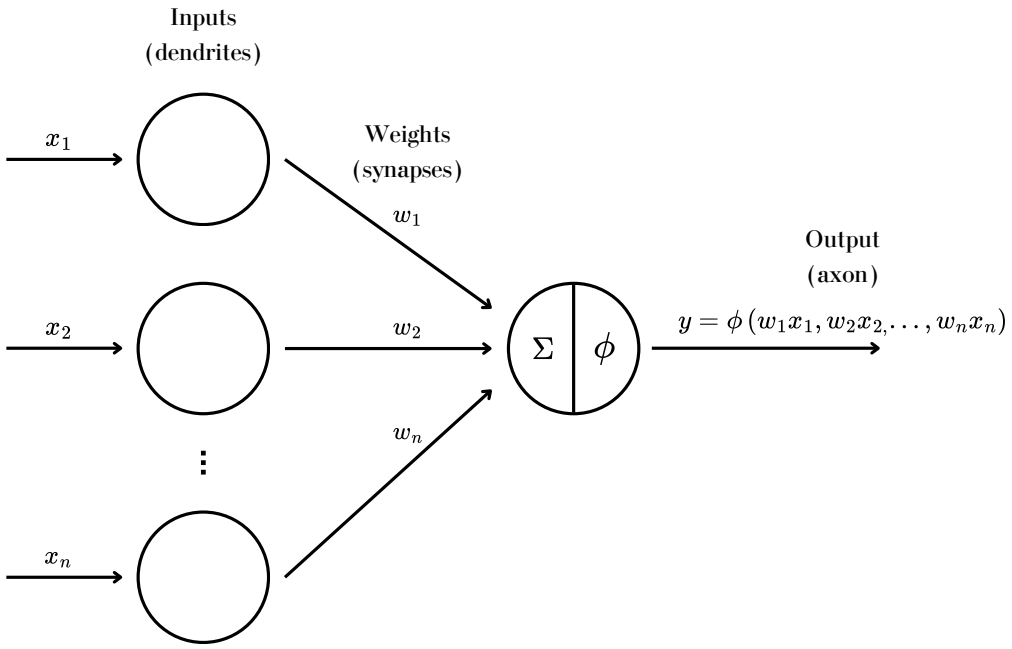


Figure 2.1 Analogy to biological neuron of basic artificial neural network unit

The basic unit of neural networks forms the foundation of machine learning architectures in supervised learning. These architectures utilize this unit in various ways. For example, stacking multiple units allows the network to process the same inputs to generate multiple outputs. Additionally, layering these units enables the product of multiple non-linear transformations, with each layer applying an activation function to enhance the model's representational capacity. The following section details the neural network architecture used in this study, describing its properties, activation functions, and possibilities for composition.

2.3.1 Architectures

The aforementioned basic unit forms the basis of the artificial neural network architectures. Specifically, the multi-layer perceptron (MLP) [18], also known as a fully connected dense neural network, is the most common choice for architecture in the context of surrogate modeling for aerodynamic coefficients, as outlined by previous studies. MLPs consist of one or more hidden layers, each composed of basic neuron units that connect to all inputs from the previous layer. These layers incorporate non-linear activation functions to enhance representational capacity and include bias parameters for shifting outputs. According to the universal approximation theorem, an MLP with at least one sufficiently large hidden layer can approximate any continuous function. This property is crucial, as complex non-linear patterns are often modeled.

For applications with small-sized inputs and outputs, this architecture is a common choice in surrogate modeling and has demonstrated strong performance in rotorcraft lookup table applications [27]. While MLPs are more complex than traditional response surface methods, they can remain relatively lightweight compared to other prominent ML architectures if configured properly. For instance, in surrogate modeling for aerodynamic coefficients, Convolutional Neural Networks (CNN) typically lead to significantly higher model complexity due to differences in data handling. Specifically, MLPs process parameterized geometric data as vectors, whereas CNNs represent geometry as pixelated maps. In summary, MLPs are well-suited for vectorial data inputs, whereas specialized architectures like CNNs or Transformers [30] are respectively preferable for spatial or temporal data.

2.4 Synthesis

The advancement and widespread accessibility of computational resources, coupled with the advent of modern machine learning techniques, have enabled significant progress in the development of AI-CFD based surrogate models for aerodynamic applications. In the context of surrogate modeling of aerodynamic predictions for rotorcraft lookup tables, research has demonstrated that neural networks have the ability to model highly complex and non-linear aerodynamic datasets, outperforming classical regression techniques and response surface methods such as POD and kriging. In addition to the application of neural networks, these studies have explored domain decomposition and geometric parameterization techniques. LHS has emerged as a preferred approach due to its efficiency and ability to achieve low-discrepancy sampling. Furthermore, several geometric parameterization techniques have been tested, following a global trend favoring orthogonality. Furthermore, CNNs have also been

used to parameterize geometry but have shown to be unnecessary in predicting scalar outputs such as aerodynamic coefficients C_L, C_D, and C_M. While these studies highlight the capability of neural networks to deliver rapid and accurate predictions of aerodynamic performance coefficients, these were generally constrained to lower-dimensional input spaces or extreme density in their training datasets. For cases constrained to low-dimensionality in inputs, models were typically only trained on a fixed geometry with varying flow conditions, or fixed flow conditions and varying geometries. For cases with high dimensionality in inputs, the problem domain is typically constrained to small variable ranges, and training points are consequently sampled very densely.

2.4.1 Scientific Gaps

The reviewed literary works reveals several scientific gaps in addressing the application challenges of AI-CFD surrogate modeling for rotorcraft lookup tables:

- Reviewed studies did not account for the entire range of boundary flow conditions typically present for rotorcraft blades;
- Reviewed studies resorted to extreme data density, failing to leverage the full interpolative capabilities of modern machine learning techniques;
- Reviewed studies did not fully address the trade-off between prediction performance and model complexity.

CHAPTER 3 OBJECTIVE

3.1 General Objective

The focus of this memoir is the development of a surrogate model based on neural networks capable of near real-time predictions of the C_L, C_D, and C_M coefficients for rotorcraft blades across all flight conditions. This surrogate model will attempt to eliminate interpolation error associated with lookup table databases traditionally used in medium-fidelity rotorcraft simulations by allowing discrete predictions generalizing towards unseen airfoil geometries and freestream flow conditions.

3.2 Specific Objectives

- Design a neural network model with limited complexity to ensure rapid prediction capability, suitable for integration into real-time flight simulators. A careful balance must be achieved between computational efficiency and predictive accuracy, acknowledging that some loss of accuracy is inherent when replacing high-fidelity CFD data with surrogate predictions.
- Extend existing literature by developing a modeling framework that generalizes across the full state space of freestream conditions and rotorcraft blade geometries. This includes defining a geometric parameterization method and a neural network architecture capable of capturing the underlying aerodynamic behavior across the present diverse scenarios.
- Employ sparse and efficient sampling strategies for training the machine learning model, avoiding exhaustive full-factorial datasets. This involves leveraging the interpolative strengths of modern ML algorithms to enable efficient database generation without compromising generalization capability.

CHAPTER 4 METHODOLOGY

In this study, an AI-surrogate model is developed to predict the lift (C_L), drag (C_D), and moment (C_M) aerodynamic coefficients, which characterize the aerodynamic performance of an airfoil geometry. These dimensionless coefficients describe the aerodynamic forces acting on an airfoil based on the surrounding fluid properties. Specifically, the freestream dynamic pressure q_∞, described in Equation 4.1, where ρ_∞ and U_∞ are respectively the freestream flow density and flow velocity.

$$q_\infty = \frac{\rho_\infty U_\infty^2}{2} \tag{4.1}$$

In the context of surrogate modeling for rotorcraft lookup tables used in medium-fidelity rotorcraft simulations, the lift (C_L), drag (C_D), and moment (C_M) aerodynamic coefficients are crucial. They enable the rapid calculation of the lift force L, drag force D, and pitching moment M, which are essential in computing flight dynamic physics. Given an airfoil's aerodynamic coefficients and surrounding flow conditions, these forces can be determined using Equations 4.2–4.4, where S is the geometry area and c is the geometry chord length.

$$C_L = \frac{L}{q_\infty S} \tag{4.2}$$

$$C_D = \frac{D}{q_\infty S} \tag{4.3}$$

$$C_M = \frac{M}{q_\infty S c} \tag{4.4}$$

As discussed in the literature review, neural networks have proven to be effective for surrogate modeling in this application. However, they require a large dataset of high-fidelity data points that adequately cover the problem domain. This study aims to develop surrogate models for predicting aerodynamic coefficients across the full range of freestream flow conditions encountered by rotorcraft blades of widely varying geometries. Aerodynamic coefficients will vary immensely across different flow regimes, flow phenomena, and differing airfoil geometries. To achieve this, CFD simulations will be used to generate data points for a diverse set of airfoil geometries across the entire envelope of flow conditions present in the problem domain. Additionally, this study will utilize a sparse dataset strategy to cover the problem domain efficiently. This approach enables efficient modeling, as leveraging the immense interpolative capabilities of modern machine learning techniques allows a sparse dataset representation of the problem domain.

The AI-surrogate modeling workflow is summarized in Figure 4.1, where the steps required to develop models optimized for the aforementioned application are outlined. This figure illustrates the key components of the modeling workflow present in this study, which will be further elaborated on in the subsequent methodology sections. The workflow begins by defining the problem domain and decomposing it into a relevant state-space of variables. An appropriate sampling strategy is then employed to generate a set of simulation cases, from which data points are obtained using an optimized CFD simulation strategy. The resulting CFD database serves as the foundation for AI-surrogate model development, where heuristics and machine learning hyperparameter optimization techniques are both used to optimize model predictive accuracy and computational efficiency.

4.1 Problem Domain

Defining the problem domain is a critical step in developing AI-CFD surrogate models. CFD databases are rarely available, and when they do exist, they seldom align with the specific problem being modeled. For rotorcraft, there is no *a priori* availability of sufficiently dense and complex datasets for training AI-based surrogate models. This research aims to explore the potential of neural network-based models as surrogate models for rotorcraft lookup tables, and given the self-imposed target of reduced-order model accuracy, several assumptions about the problem domain are adopted. The objective is to predict the aerodynamic coefficients of rotorcraft blade sections using machine learning techniques requiring a database. This database must encompass data points spanning the full range of flow conditions a rotorcraft blade section might encounter while also capturing the diversity of possible blade-section geometries.

4.1.1 Flow Conditions

The definition of the problem domain's flow conditions necessitates identifying the key non-dimensional numbers governing the flow around rotorcraft blade-sections. The Mach and Reynolds numbers are the primary dimensionless parameters characterizing these flows, describing compressibility effects and the relative significance of inertial versus viscous forces, respectively. These values can be computed from freestream physical quantities, including velocity (U_∞), density (ρ_∞), temperature (T_∞), and dynamic viscosity (μ_∞). Hence, establishing realistic ranges for these freestream parameters is essential for defining the relevant non-dimensional characteristics of the problem domain.

The Mach number acting on rotorcraft blade sections results from the combined velocities

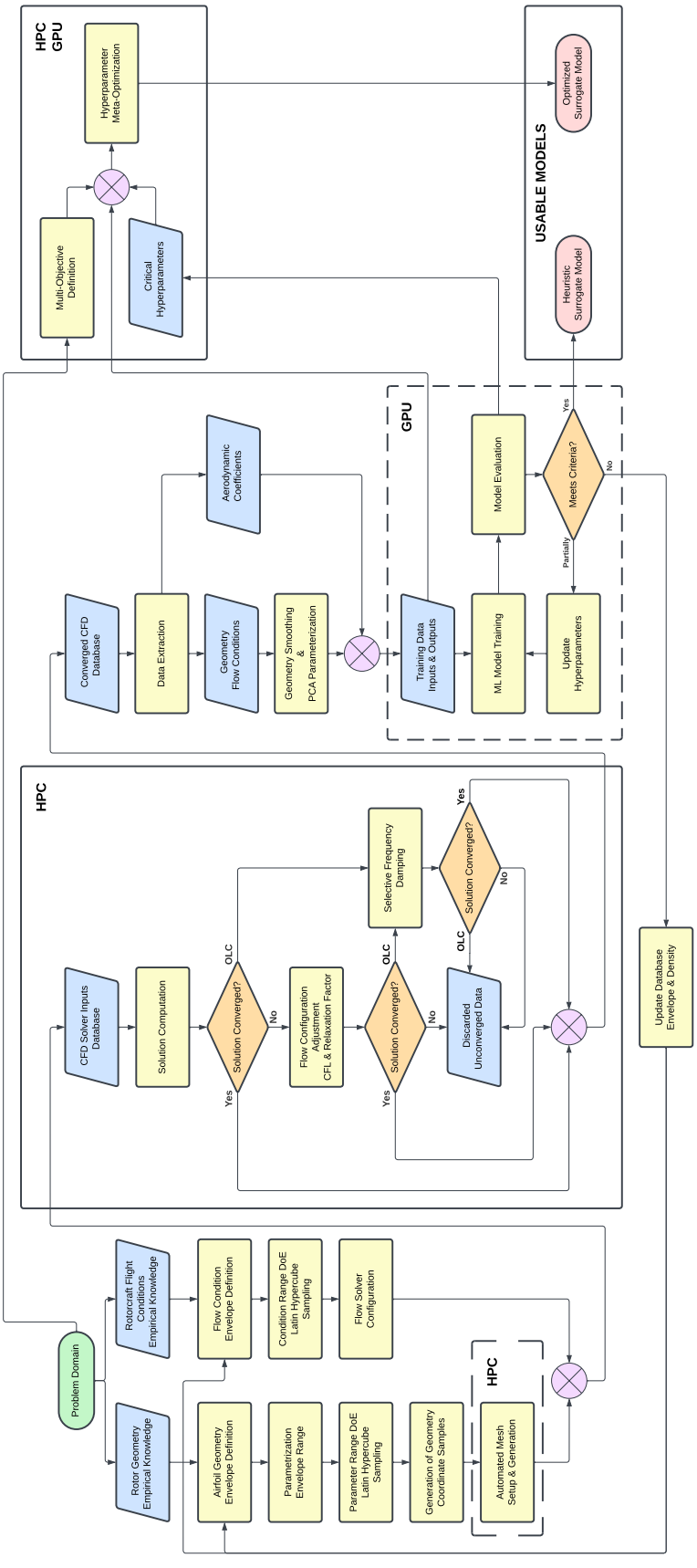


Figure 4.1 AI-CFD surrogate modeling workflow
(HPC: High-Performance Computing, OLC: Outer Limit Cycle, GPU: Graphics Processing Unit)

of the rotor's revolution and the relative velocity induced by the surrounding wind and the rotorcraft's motion. The maximum velocity due to rotor revolution occurs at the blade tip, where speeds are typically constrained to values below $M_{tip} = 1$ to limit compressibility effects. However, in rare cases, the freestream Mach number acting on rotor tips can be supersonic [31]. The maximum Mach number is limited to $M = 0.9$ to remain conservative. While high-speed rotorcraft can be designed for cruise velocities exceeding 300 knots ($M \approx 0.45$) [32], and extreme wind speeds exceeding 231 mph ($M \approx 0.3$) have been recorded [33], the maximum allowable relative wind velocity is conservatively limited to $M = 0.5$, as operational flights would not take place under such extreme conditions.

The resulting combination of rotor Mach number (M_R) and relative wind Mach number (M_W) is further constrained by the total flow's angle of attack (α) and the relative wind's inflow angle (β_W). This relationship will result in reversed, deep stall, and transonic flow configurations. Figure 4.2 describes the resultant combination of the aforementioned variables for a static rotorcraft blade-section, as considered in this study. Here, 'static' indicates that the blade does not undergo pitch, heave, or lead-lag motion.

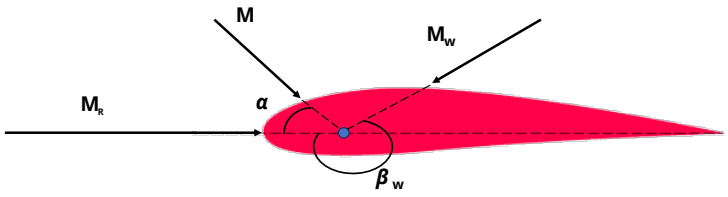


Figure 4.2 Wind and rotor velocity interactions used in finding resulting Mach number

The resultant M with respect to α can be found with the following system of equations. The flow induced by the rotor's revolution aligns with the chord line, meaning that M_R contributes exclusively to the chordwise component of M, with no influence on M_y, its normal component.

$$M_x = M_R + M_W \cdot cos(\beta_W) \tag{4.5}$$

$$M_y = M_W \cdot sin(\beta_W) \tag{4.6}$$

$$M = (M_x^2 + M_y^2)^{1/2} \tag{4.7}$$

$$\alpha = arctan(M_y/M_x) \tag{4.8}$$

The domain of flow conditions in the $M - \alpha$ space is illustrated in Figure 4.3, where the red lines delineate the boundary of physically realistic combinations. This domain is constructed by sampling M_R over the range 0 to 0.9, M_W over 0 to 0.5, and β_W over -180° to 180°, with values outside the range $[0.2, 0.9]$ excluded.

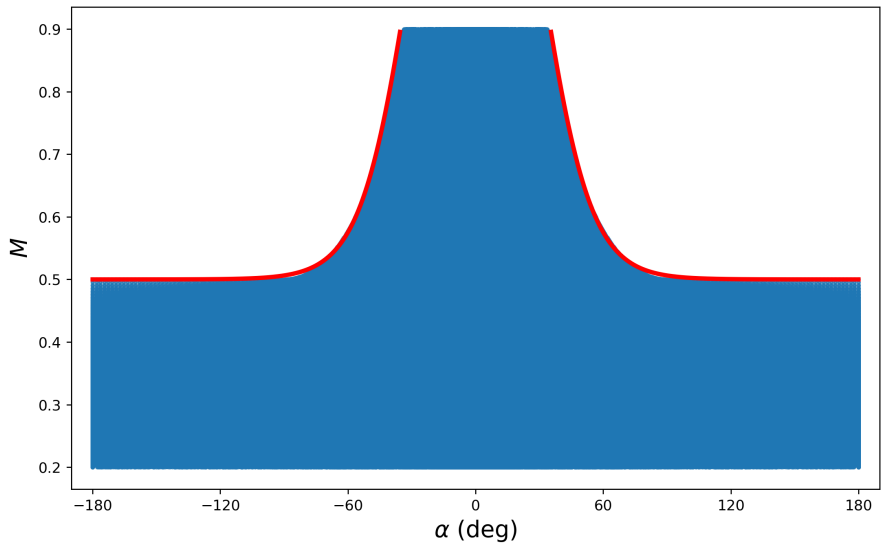


Figure 4.3 Flow condition envelope for Mach versus angle of attack

The freestream temperature (T_∞ in K), used for calculating of the flow's dynamic viscosity, is determined using the International Standard Atmosphere (ISA) model for the troposphere. The relationship between altitude h (m) and temperature T (K) is given by Equation 4.9, as defined by the ISA model. This model employs the standard constants, being the sea-level temperature ($T_0 = 288.15$ K) and the tropospheric lapse rate ($L = -0.0065$ K/m).

$$T(h) = T_0 + Lh \tag{4.9}$$

The flow's dynamic viscosity (μ_∞ in Pa·s) can be calculated using Sutherland's formula, as described in Equation 4.10. Given the assumption that the medium is an ideal gas with the specific thermophysical properties of air, reference constants can be used, where $\mu_{ref} = 1.716 \times 10^{-5}$ Pa·s is the reference viscosity, $T_{ref} = 273.15$ K is the reference temperature, and $S = 110.4$ K is Surtherland's constant.

$$\mu = \mu_{ref}(\frac{T}{T_{ref}})^{3/2}\frac{(T_{ref} + S)}{T + S} \tag{4.10}$$

Furthermore, the freestream density (ρ_∞ in kg/m^3) can be found using the barometric formula for an altitude range with a linear temperature gradient. Equation 4.11 for tropospheric atmosphere can be derived using the ISA model's assumptions, where $T(h)$ is given by Equation 4.9, $T_0 = 288.15$ K is the sea-level temperature, $g = 9.81$ m/s^2 is the gravitational acceleration near Earth's surface, $L = -0.0065$ K/m is the tropospheric lapse rate, $R = 287.05$

J/(kg·K) is air's specific gas constant, and $\rho_0 = 1.225$ kg/m^3 is air's density at sea-level.

$$\rho(h) = \rho_0 \left(\frac{T(h)}{T_0} \right)^{\frac{g}{LR}} \tag{4.11}$$

The relation between freestream velocity (U_∞ in m/s) and the independent variables Mach number and freestream temperature (T_∞ in K) is described in Equation 4.12, where $\gamma = 1.4$ is approximately air's specific heat ratio and $R = 287.05$ J/(kg·K) is air's specific gas constant.

$$U_\infty = M\sqrt{\gamma R T_\infty} \tag{4.12}$$

The Reynolds number is the non-dimensional value that characterizes the flow's relative significance of inertial versus viscous forces. In this instance, it is the dependent variable with respect to freestream velocity (U_∞), density (ρ_∞), and dynamic viscosity (μ_∞), as described in Equation 4.13. Furthermore, in this study, Re is fixed to a reference length of a chord (c) of 1 meter.

$$Re = \frac{\rho_\infty U_\infty c}{\mu_\infty} \tag{4.13}$$

The relationship between the key non-dimensional Reynolds and Mach numbers, which characterize the problem domain's flow conditions, is described by Equations 4.5–4.13. The independent variables M, α, and h determine the freestream physical parameters ρ_∞, U_∞, and μ_∞, which are then used to obtain Re. Consequently, defining the problem domain's flow conditions necessitates a sweep of the low-Earth atmosphere and the M – α relationship.

4.1.2 Airfoil Geometry

The definition of the problem domain's geometries necessitates the inclusion of a diverse range of airfoil shapes that realistically represent rotorcraft blade-sections. While rotorcraft airfoils are typically thin with minimal camber, different rotor designs necessitate a broad spectrum of geometries. Thickness, camber (including negative camber), and leading-edge radius are examples of airfoil characteristics that will typically differ greatly across rotor designs. In order to ensure generalization towards unseen geometries, the sampled database must include airfoils spanning these design variations. The UIUC airfoil database [34] has been a traditional resource but exhibits biases toward specific airfoil families, such as Goettingen and NACA, due to their predominant representation. Hence, airfoil geometry parameterization methods can be explored for geometry generation. The chosen parameterization approach must facilitate the generation of airfoils across multiple styles while maintaining a particular

focus on thinner profiles, as rotorcraft blades typically feature thin and minimally cambered airfoils.

4.2 Envelope Definition

The range for the problem domain's design variables can be inferred from Section 4.1. The ranges for independent variables for the rotorcraft blade-section full state-space are defined. Furthermore, the relationship between key independent and dependent variables is described, permitting their subsequent definition.

4.2.1 Flow Conditions Envelope

The ranges of the independent design variables defining the problem domain's flow conditions are provided in Table 4.1. The full range of α is included, as C81 lookup tables typically span values from -180° to 180°. While the Mach number is not directly dependent on the angle of attack, α imposes a constraint on the maximum attainable M for a given α. Additionally, M is restricted to the range of 0.2 to 0.9, as outlined in Section 4.1. Lastly, a conservatively large range for h is selected to encompass the altitudes relevant to urban flight operations.

Table 4.1 Envelope range for independent flow condition design variables

Variable	Range
Angle of attack	$\alpha \in$ [-180, 180] °
Mach number	$M \in$ [0.2, 0.9], as a function of α
Altitude	$h \in$ [0, 5000] m

Given the independent variables in Table 4.1, and the assumptions and models outlined in Section 4.1.1, the range for the dependent variable Re can be determined, as described in Table 4.2. The independent variables are used to determine physical parameters used in the computation of the Reynolds number.

Table 4.2 Envelope range for the dependent flow condition variable

Variable	Range
Reynolds number	$Re \in [2.5, 21.0] \times 10^6$, as a function of M & h

4.2.2 Geometry Envelope

Several airfoil parameterization techniques have been considered, but Non-Uniform Rational B-Splines (NURBS) [35] were selected for representing the geometry envelope. Unlike traditional parameterization methods such as PARSEC [14], NURBS offer greater flexibility in generating and representing highly varied and irregular airfoil geometries. However, while PARSEC provides intuitive parameters, NURBS control points can be less straightforward to manipulate, requiring heuristic approaches to ensure realistic airfoil shapes. Nevertheless, NURBS serve as a powerful airfoil parameterization tool. It permits representations that encompass the entire envelope of possible rotor blade-section geometries, something that would be impossible with the constraints imposed by classical airfoil parameterization techniques. Furthermore, higher-order NURBS representation allows for C^2 curvature continuity.

The geometry envelope is defined by specifying ranges for the Cartesian coordinates of the NURBS control points. By definition, each control point will have a different weight. However, it was found heuristically that fixing the weights across all airfoils simplifies shape control and that simply varying the control points' positions was sufficient. Hence, the airfoils are generated by describing the Cartesian coordinates of 9 control points P_i, as illustrated in Figure 4.4.

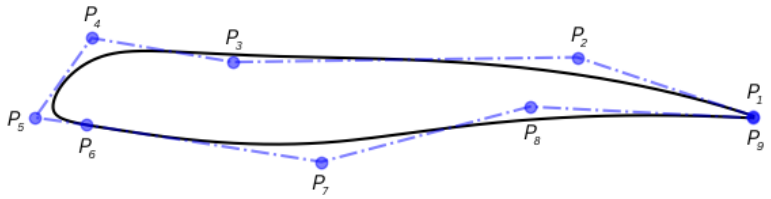


Figure 4.4 Example of an airfoil generated with NURBS control points

The weights and the envelope ranges for the Cartesian coordinates of the NURBS control points P_{1-4} and P_5 are respectively defined in Table 4.3 and Table 4.4.

Table 4.3 Envelope range for control points P_{1-4} (Cartesian coordinates normalized to chord)

	P_1	P_2	P_3	P_4
x	[1.0, 1.0]	[0.6, 0.8]	[0.2, 0.4]	[0.025, 0.075]
y	[0.0005, 0.005]	[-0.10, 0.15]	[0.05, 0.15]	[0.05, 0.15]
w	1	1	2.5	10

The control points P_{6-9} are defined differently from the preceding ones. The envelope range of their Cartesian y-coordinate is not directly defined, but rather measured with respect to the

Table 4.4 Envelope range for the control point P_5 (Cartesian coordinates normalized to chord)

	P_5
x	[-0.05, 0]
y	0
w	100

y-coordinates belonging to the P_{1-4} control points. This approach eliminates unreasonable shapes and prevents the possibility of overlapping upper and lower surfaces. Furthermore, ensuring that control points P_1 and P_9 have distinct y-coordinates guarantees a nonzero trailing edge thickness, allowing for greater representation fidelity. The weights and the envelope ranges for the Cartesian coordinates of the NURBS control points P_{6-9} are defined in Table 4.5.

Table 4.5 Envelope range for control points P_{6-9} (Cartesian coordinates normalized to chord)

	P_6	P_7	P_8	P_9
x	[0.025, 0.075]	[0.2, 0.4]	[0.6, 0.8]	[1.0, 1.0]
Δy	[-0.10, -0.15]	[-0.05, -0.15]	[-0.05, -0.10]	[-0.0005, -0.005]
y	$y_4 + \Delta y_6$	$y_3 + \Delta y_7$	$y_2 + \Delta y_8$	$y_1 + \Delta y_9$
w	10	2.5	1	1

4.3 Sampling Method

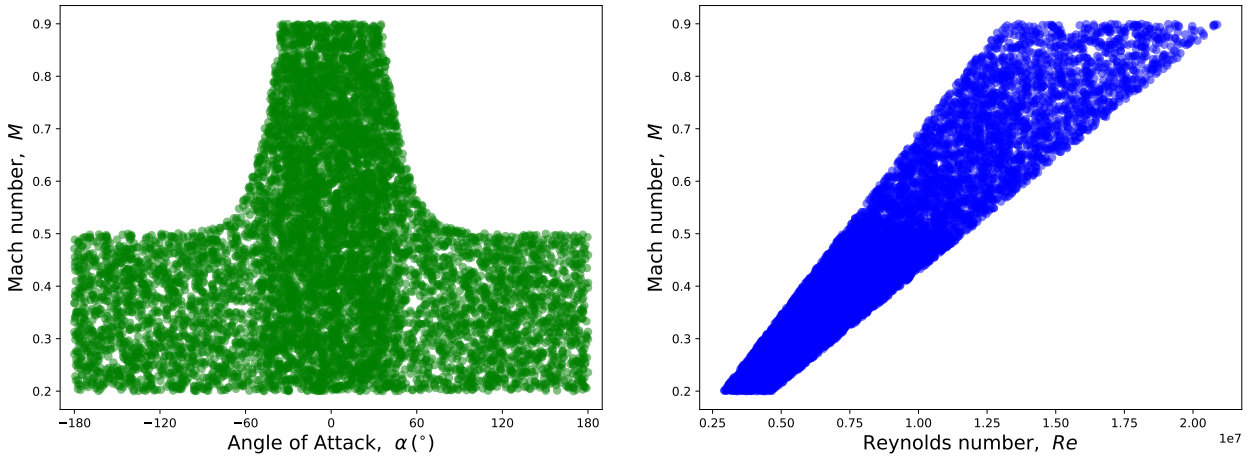
Latin Hypercube Sampling (LHS) [36] is the chosen method for constructing the Design of Experiments (DoE) that encapsulates the flow condition and geometry envelope ranges described in Section 4.2. LHS is a widely used sampling technique in DoE. This method is particularly well-suited for machine learning applications, specifically those involving CFD-generated databases. LHS has smaller variance compared to random sampling, ensuring a greater global uniformity across design variables. In methods such as Sobol [37], adding new samples may disrupt the existing structure, whereas LHS maintains global uniformity through locally stratified intervals that do not disrupt the distribution characteristics. This characteristic of LHS is particularly useful when iteratively refining a dataset, as new samples can be integrated without reconstructing the entire DoE. For example, out of the 16541 total samples used in this study's AI-surrogate modeling, roughly 8000 *a posteriori* additions were made after the definition of the initial DoE.

LHS was the method utilized in sampling the ranges for M, h, and NURBS control point coordinates. However, the sampling of α followed a custom probability distribution defined in Equation 4.14, normalized using Equation 4.15 to satisfy the fundamental requirement that probability density functions integrate to 1. The M – α relation described in Equations 4.5–4.8 made it that large values of M are only found near small values of $\alpha = 0$. Hence, custom sampling was used in favor of LHS as having a greater sample concentration near $\alpha = 0$ offered a greater uniformity with respect to the entire range of M. This custom probability distribution was found heuristically with the aim of roughly representing a distribution function following the shape of the red lines represented in Figure 4.3. Furthermore, the ϵ found in the custom distribution function was slightly tweaked in every iterative dataset refinement to ensure that no same α value was sampled twice.

$$f(\alpha) = \begin{cases} \tanh\left(\frac{\alpha+30}{20} + \epsilon\right) + 1, & \text{if } \alpha < 0 \\ \tanh\left(\frac{30-\alpha}{20} + \epsilon\right) + 1, & \text{if } \alpha \geq 0 \end{cases} \tag{4.14}$$

$$C = \frac{1}{\int_{-\infty}^{\infty} f(\alpha)\, d\alpha} \tag{4.15}$$

Utilizing the aforementioned techniques resulted in the sampled M – α envelope shown in Figure 4.5a. Dependent variables, such as Re, were subsequently computed based on the sampled independent variables, M and h, resulting in the sampled M – Re envelope shown in Figure 4.5b.



(a) (b)

Figure 4.5 Sampled (a) M – α and (b) M – Re envelopes

Furthermore, LHS was used to sample NURBS control point coordinates, resulting in 16541

unique airfoil geometries each associated with a combination of unique flow conditions M, Re, and α. Three examples of NURBS generated airfoils are illustrated in Figure 4.6.

Figure 4.6 Examples of airfoils generated with sampled NURBS control points

4.4 CFD Configuration

The generation of an ML training database using a CFD solver necessitates a thorough examination of solver parameters and meshing strategies. The trade-off between solution uncertainty, fidelity, and computational cost must be carefully balanced, as these factors directly influence the downstream development of the ML surrogate model. Insufficient solution fidelity can lead to a dataset that fails to capture the complex, nonlinear flow physics inherent to the problem domain, ultimately resulting in a surrogate model that inadequately represents real-world behavior. Furthermore, high solution uncertainty can also degrade the quality of the developed surrogate model as there may be greater difficulty in discerning the dataset's underlying patterns through deep learning algorithms. Dataset density is another critical factor, as an inadequate number of data points can impair the effectiveness of ML training. However, there exists an inherent trade-off between dataset density, solution fidelity, and uncertainty. For a given computational budget, prioritizing dataset density may require compromising on solution fidelity and uncertainty, as lower-fidelity simulations can be computed more rapidly, enabling a larger dataset to be generated. Inversely, prioritizing solution fidelity and uncertainty will lead to the generation of a smaller dataset given the same computational budget.

The selection of an optimal set of solver parameters and mesh generation properties that properly proportions database fidelity, uncertainty, and density for a given problem application can be estimated *a posteriori*, after database generation and model development. Identifying this balance *a priori*, however, is typically guided by heuristics, as achieving an optimal set of parameters beforehand is an inherently complex and challenging task.

4.4.1 Meshing Strategy

The first step in developing an appropriate spatial discretization strategy was to perform a mesh refinement study. Rather than assessing the software's spatial order of convergence, something typically conducted for software verification, this study aimed to determine the level of refinement required to achieve the necessary accuracy for the given problem application. The refinement study was conducted on a simple test case featuring a NACA 0012 airfoil in the freestream conditions of $\alpha = 15°$, $M = 0.15$, and $Re = 6 \times 10^6$.

The Cadence Fidelity *Pointwise* meshing software was used to generate this study's meshes. The NACA 0012 geometry was first imported and fitted with a cubic spline to ensure that the prescribed number of i-indexed (indexed along wall surface) nodes are comprised in the geometry's actual shape, ensuring C^2 continuity. Therefore, nodes surrounding the geometry can be distributed along the cubic spline using hyperbolic tangent functions for appropriate spacing, concentrating points near the geometry's leading- and trailing-edges. Cells were then generated in the j-index (indexed normal to wall surface) direction by extruding the distributed spline points. The curvilinear mesh is generated with a hyperbolic extrusion algorithm, following a growth rate of 1.1, extending outward until the far-field boundary reaches a distance of roughly 30 chord lengths. Figure 4.7 illustrates the effect of mesh refinement on solution accuracy, where the y^+ value described in legend is the minimum dimensionless wall distance. The dimensionless wall distance is defined in Equation 4.16, where u_τ is the friction velocity and ν is the kinematic viscosity.

$$y^+ = \frac{u_\tau y_{wall}}{\nu} \tag{4.16}$$

The dimensional wall distance (Δs) required to achieve the target y^+ values is estimated using empirical formulas based on flat-plate boundary layer theory [38]. Specifically, Δs is calculated as a function of the Reynolds number (Re_x) evaluated at the chordwise location $x = c$, along with the freestream velocity, fluid density, and dynamic viscosity, as described in Equations 4.17. In these expressions, C_f denotes the skin friction coefficient, τ_{wall} the wall shear stress, and u_τ the friction velocity.

$$\begin{cases} C_f = \frac{0.026}{Re_x^{1/7}}, \\ \tau_{wall} = \frac{C_f \rho_\infty U_\infty^2}{2}, \\ u_\tau = \sqrt{\frac{\tau_{wall}}{\rho_\infty}}, \\ \Delta s = \frac{y^+ \mu_\infty}{u_\tau \rho_\infty}. \end{cases} \tag{4.17}$$

In this case, due to the turbulence model selection described in the next section, the mesh must be sufficiently refined to ensure a first cell-center dimensionless wall distance of $y^+ < 1$. This requirement arises from the choice of turbulence model as well as the absence of a wall model, necessitating $y^+ < 1$ to accurately resolve the viscous sublayer. In the mesh refinement study conducted at $Re = 6 \times 10^6$, a minimum normal wall-spacing of $\Delta s_{\min} = 4.45 \times 10^{-6}$ chord was required to meet this criterion.

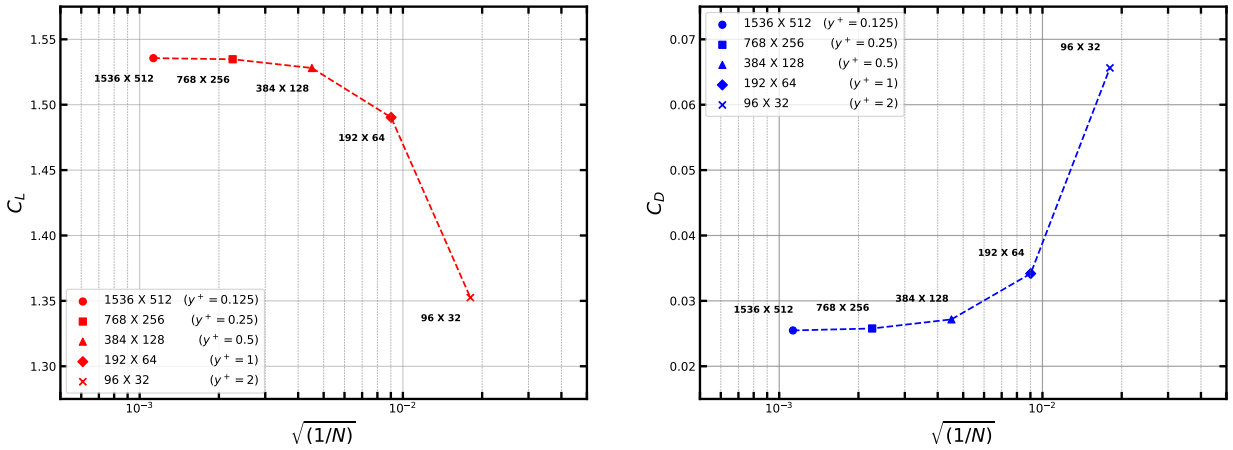
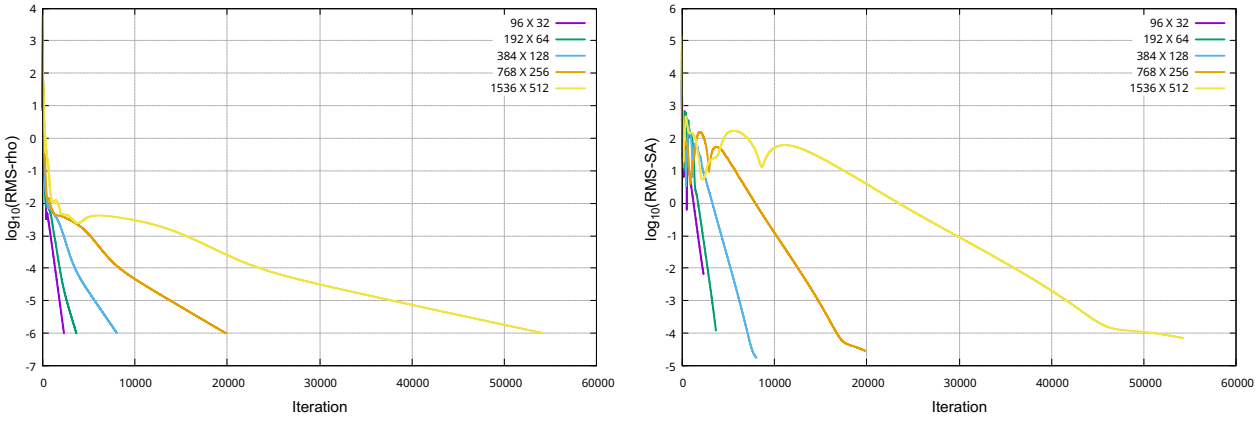


Figure 4.7 Mesh refinement impact on (a) C_L and (b) C_D

Furthermore, the impact of grid resolution on simulation time was assessed in this mesh refinement study. The number of iterations required to achieve convergence is shown in Figure 4.8. The scaled ρ-residual, a figure indicating proximity in solving the non-linear flow system, is taken as the convergence criterion when below 10^{-6}. Although the corresponding wall-clock times were erroneously not recorded at the time, the computational cost for each refinement level can be inferred based on the number of iterations needed for convergence.

The mesh refinement study provides insight into the tradeoff between grid resolution and computational cost, where an appropriate resolution was identified between 192×64 and 384×128 for the problem application. However, additional heuristic optimization was applied to refine the meshing strategy further. Heuristics were used to improve quality in key mesh quality metrics, including cell orthogonality, skewness, and aspect ratio, which affect the accuracy and time required for convergence.

The heuristically optimized mesh builds upon the initial meshing strategy employed in the refinement study. This strategy continues to utilize 2D curvilinear and quadrilateral cells. However, unlike the refinement study, which relied on meshes generated in the *C-mesh* style,



(a) (b)

Figure 4.8 Mesh refinement convergence history for (a) ρ and (b) ν_t residuals

the optimized approach adopts an *O-mesh* configuration. The *O-mesh* offers more uniform grid resolution, reducing directional bias in wake refinement. For instance, a *C-mesh*, which enhances resolution primarily in the trailing-edge wake, remains suitable only for flows with low angles of attack (α). Given the range of α spanning $-180°$ to $180°$, as described in Section 4.2, and the requirement for a meshing strategy applicable to all simulation cases, the *O-mesh* was selected for its ability to provide consistent wake resolution across the entire range of possible angles of attack.

The minimum normal wall spacing was adjusted to $\Delta s_{\min} = 1.0 \times 10^{-6}$ chord to ensure a first cell-center dimensionless wall distance of $y^+ < 1$ across the entire simulation domain. However, this reduction in wall spacing led to an increase in cell aspect ratio throughout the generated grid, specifically in the near-wall region. Hence, node refinement in the i-direction was increased to mitigate this effect, reducing the maximal measured global aspect ratio to tolerable values (guided by laboratory expertise, 10^4 was chosen as the limit). Furthermore, i-direction node distribution was optimized based on mesh quality metrics. Notably, for wall-adjacent cells connecting the trailing edge to the upper and lower surfaces, the i-direction length was limited to 1.0×10^{-5} chord to maintain cell skewness below 0.5 across the entire mesh domain.

Additionally, a more sophisticated extrusion strategy is employed to ensure near-wall cell orthogonality. An algebraic extrusion algorithm with a growth rate factor of 1.1 is applied to generate the first 50 cells in the j-direction. Beyond this region, the hyperbolic extrusion algorithm, with a growth rate factor of 1.15, is employed to extend the grid outward until the far-field boundary reaches approximately 50 chord lengths. The hyperbolic extrusion algorithm offers greater grid resolution uniformity at the cost of worsened orthogonality. While

the hyperbolic extrusion enhances grid resolution uniformity, it compromises orthogonality. This trade-off is intentional, as orthogonality is crucial in the near-wall region, whereas a uniform grid with lower density is preferable in the far field.

The resulting meshing strategy consistently produces meshes similar to the one shown in Figure 4.9, typically comprising 240×120 cells. Moreover, this approach is robust across the entire problem domain's geometric envelope, enabling automation within the AI-CFD surrogate modeling workflow.

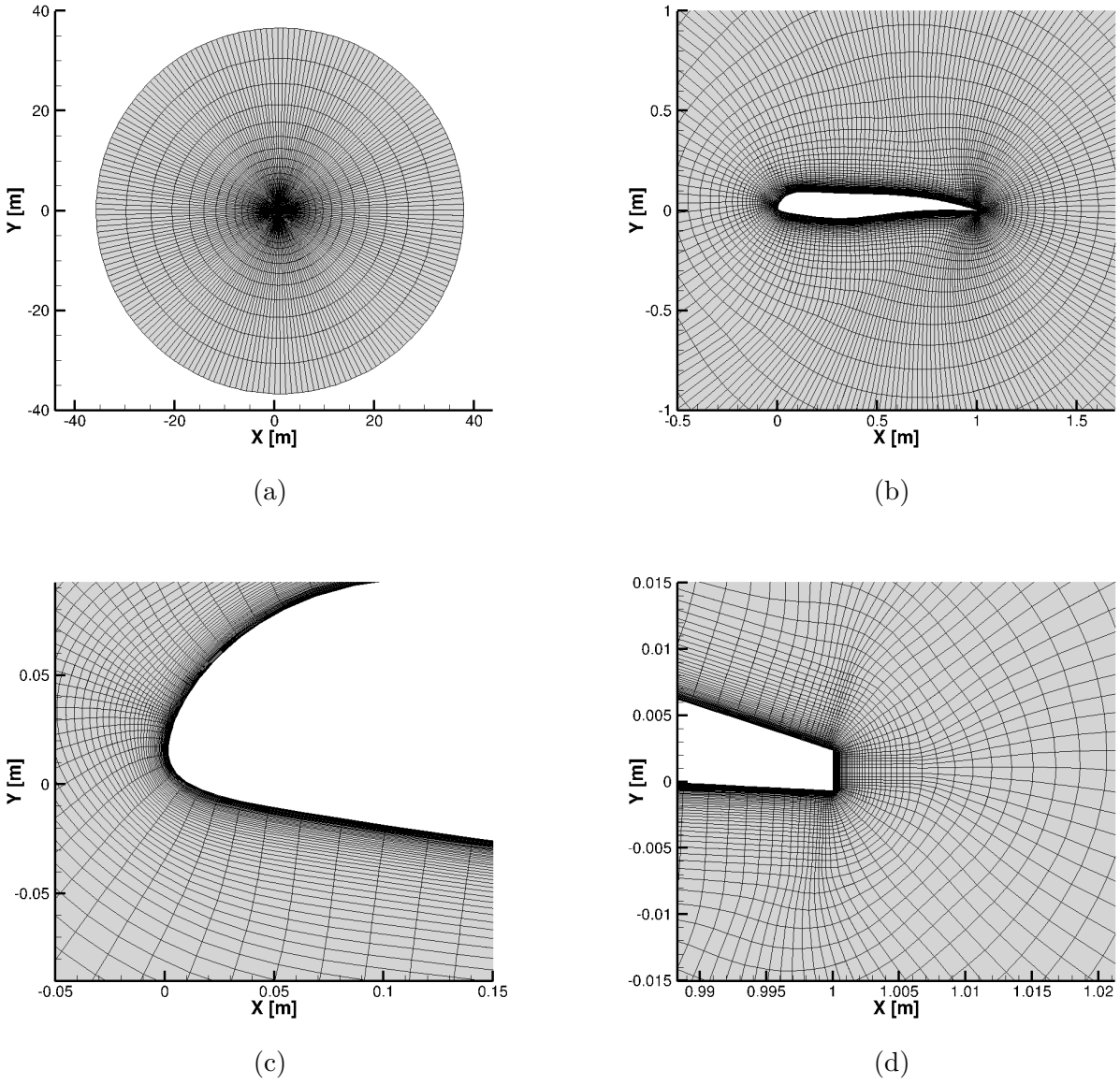


Figure 4.9 Global view of (a) far- and (b) near-field mesh areas, and local view of (c) leading- and (d) trailing-edge mesh areas

4.4.2 Flow Solver Parameter Configuration

The CHAMPS [39] (CHApel Multi-Physics Simulation) software is used to generate the CFD flow solutions required in the ML training database. CHAMPS is a cell-centered, density-based, FVM solver capable of simulating complex transonic flows on 3D unstructured grids. This software is developed in the new Chapel programming language for aerodynamic simulations and notably features compressible RANS-based algorithms. The Chapel language's use in this CFD solver presents an alternative to traditional scientific computing paradigms, such as C++ combined with MPI libraries for supercomputing parallelism.

The flow solver and meshing strategy described in the previous section are configured to operate in a shared-memory parallelism framework. This choice leverages the low-latency communication between processors on a single node, ensuring computational efficiency. While distributed-memory configurations can reduce the wall-clock time per individual calculation by leveraging a larger number of processors, the total computational time required to generate the entire database would ultimately increase. Furthermore, the relatively coarse meshes tailored for RANS-based algorithms further diminish the efficiency of distributed-memory computations. In CFD, the computational burden of solving the flow field decreases with coarser meshes due to the reduced number of volume cells. However, the communication overhead remains large with distributed-memory. As a result, the trade-off between computational cost and communication overhead renders distributed-memory computations unnecessary for this specific application. While CHAMPS was conceived for distributed-memory usage, the software offers a high level of sophistication, supporting multiple compilation methods that respectively optimize performance for both shared- and distributed-memory configurations. In this instance, CHAMPS' shared memory parallelism permits an efficient CFD database generation with limited communication overhead.

In this study, the parameters used in the CHAMPS CFD software are configured heuristically. A *bounding-box* of test cases is designed to encompass the most extreme simulation conditions within the problem domain. Simulations are conducted in the *bounding-box* in order to favor convergence and stability across the entire problem domain. The cases prescribed in this *bounding-box* are listed in Table 4.6. Furthermore, each case is tested on several geometries. The NACA 4205, NACA 4220, NACA 4805, and NACA 4820 are evaluated in the *bounding-box* as they represent geometries that would incur extreme flow conditions due to their respective cambers, max camber positions, and small or large thicknesses. In the heuristic parameter tuning process, parameters that primarily influenced convergence were adjusted liberally, whereas those influencing solution accuracy were tuned conservatively to minimize numerical degradation. While not infallible, this approach maximizes the likeli-

hood of convergence across a significant portion of the simulation database with minimal intervention.

Table 4.6 *Bounding-box* for heuristic parameter search

	Angle of attack (α)	Mach number (M)	Reynolds number (Re)
Case 1	$-180°/180°$	0.50	10.6×10^6
Case 2	$-90°$	0.55	11.7×10^6
Case 3	$0°$	0.90	19.1×10^6
Case 4	$90°$	0.55	11.7×10^6

The Roe flux differencing scheme [40] is selected as it excels at capturing discontinuities, such as shockwaves that will inherently be present in the simulated problem domain. Additionally, Harten's correction [41] [42] is used to ensure the satisfaction of the entropy condition. While only slightly sacrificing solution accuracy, an entropy correction coefficient $\delta = 0.30$ was found to substantially increase stability with respect to typically smaller nominal values $\delta = 0.10$ or $\delta = 0.05$. Furthermore, the Monotonic Upstream-centered Scheme for Conservation Laws (MUSCL) [43] is paired with the Roe scheme and serves as a reconstruction technique to allow for higher-order spatial accuracy. This combination allows for 2nd-order accurate flux calculations while maintaining the upwind nature of the Roe scheme, permitting increased stability with respect to available centered 2nd-order accurate schemes. However, the inclusion of the MUSCL reconstruction requires the addition of slope-limiters for stability, as the presence of sharp gradients calculated across cell interfaces may cause spurious oscillations without them. Venkatakrishnan's limiter [44] [45] is used to satisfy this requirement, where the limiter coefficient $K = 5$ was found to be sufficient. Additionally, the gradients used in MUSCL reconstruction are computed with Weighted Least Squares (WLS) for stable approximations.

Furthermore, the inclusion of a turbulence model is essential for accurately capturing the complex nonlinear phenomena inherent in the flow fields surrounding rotorcraft blades. The one-equation Spalart-Allmaras (SA) [46] turbulence model is employed, selected over algebraic (zero-equation) and two-equation models. The SA model provides sufficient fidelity for the problem domain while incurring lower computational costs compared to two-equation models. Notably, it was shown to effectively capture complex flow phenomena, such as deep stall, with adequate fidelity and accuracy.

The CHAMPS solver is configured implicitly to ensure stability and accelerate convergence for the presented steady-state problem. Hence, this implicit approach allows the use of Courant–Friedrichs–Lewy (CFL) numbers greater than 1. Heuristic tuning within the previ-

ously defined *bounding-box* tests determined a maximally stable CFL = 50. In simulations, this number is initialized to CFL = 1, and then ramped to CFL = 50 between the 100^{th} and 300^{th} iterations. Given the implicit configuration, the resulting linear system is solved using the Lower-Upper Symmetric Gauss-Seidel (LU-SGS) algorithm. Additionally, this heuristic tuning found that a relaxation factor of $\omega = 0.7$ between iterations was necessary for stability.

4.5 CFD Generation of Database

Once the optimization of CFD solver parameters and the automated meshing strategy, as described in Section 4.4, are complete, the CFD database for ML training can be generated. This database spans the simulation envelope of geometries and flow conditions, capturing the full state-space of rotorcraft blade-section configurations as defined in Section 4.2. Given the substantial computational demands of generating a sufficiently large dataset for ML training, High-Performance Computing (HPC) is essential. As previously mentioned, simulations are executed on shared-memory systems to enhance efficiency by minimizing communication overhead, a critical factor in large-scale database generation.

For improved computational efficiency and database completeness, a strategy was developed for simulation cases that failed to meet the prescribed convergence criteria outlined in Table 4.7. The aforementioned *bounding-box* study identified that a Cauchy windowing convergence criterion was the most suitable approach to ensure appropriate aerodynamic coefficient convergence while limiting the number of necessary simulation iterations. Specifically, for the C_x coefficient, a tolerance of 10^{-8} was applied over a Cauchy window spanning 300 iterations. If this criterion was not met, an alternative convergence condition based on the root mean square (RMS) ρ-residual, with a tolerance of 10^{-2}, was used. This combination effectively reduced computational cost while ensuring acceptable engineering accuracy. If neither criterion was satisfied, a bifurcation strategy, described in Appendix A, was employed to ensure database completeness, thereby reducing the need for excessive DoE infill.

Table 4.7 Simulation convergence conditions

	Condition	Outcome
C_1	Cauchy window criterion is met	Converged
C_2	NaN is found	Bifurcation
C_3	ρ-residual $> 10^0$ (Outer Limit Cycle)	Bifurcation
C_4	$10^0 > \rho$-residual $> 10^{-1}$	Pruned
C_5	$10^{-1} > \rho$-residual $> 10^{-2}$	Pruned
C_6	$10^{-2} > \rho$-residual	Converged

4.5.1 Processed Database

An overview of the bifurcation strategy is presented in Figure 4.10, illustrating the division of database subsets labeled **(a)** to **(d)**. The process begins with the initial simulation of cases in subset **(a)**, followed by the evaluation of their convergence conditions. Cases satisfying the C_3 condition proceed to subset **(b)**, where Selective Frequency Damping (SFD), detailed in Appendix A, is applied, while cases with the C_2 condition move to subset **(c)** for solver parameter calibration aimed at improving stability. After re-simulating cases in **(c)** with calibrated parameters, those found with the C_3 condition are subsequently moved to subset **(d)** for additional SFD application. Upon completion of the bifurcation strategy, cases with unconverged conditions (C_2, C_3, C_4, or C_5) are discarded. Cases achieving convergence under conditions C_1 or C_6 are retained for ML training. The subset **(e)** (not shown in Figure 4.10) aggregates all converged and discarded simulation data points generated during this process.

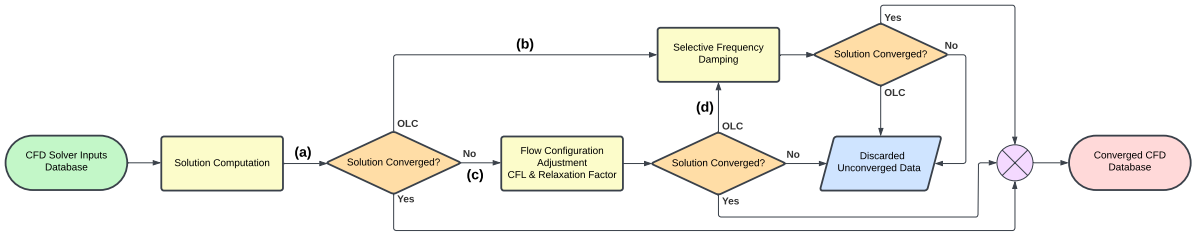


Figure 4.10 Bifurcation strategy overview

Table 4.8 illustrates the effectiveness of the bifurcation strategy alongside the CFD configuration described in Section 4.4 across 16,541 simulation data points. Without incorporating SFD or calibrating solver parameters where necessary, the convergence rate stands at 82.15%. Introducing the bifurcation strategy increases the convergence rate to 95.56%, representing a 13.41% improvement, yielding an additional 2218 successfully converged data points.

Table 4.8 Database generation synopsis

Subset \ Criteria	C_1	C_2	C_3	C_4	C_5	C_6	CONV%
(a)	12618	1068	1838	26	20	971	82.15
(b)	1451	14	291	1	1	80	83.30
(c)	565	52	365	7	31	48	57.40
(d)	66	0	279	8	4	8	20.27
(e)	14700	66	570	42	56	1107	95.56

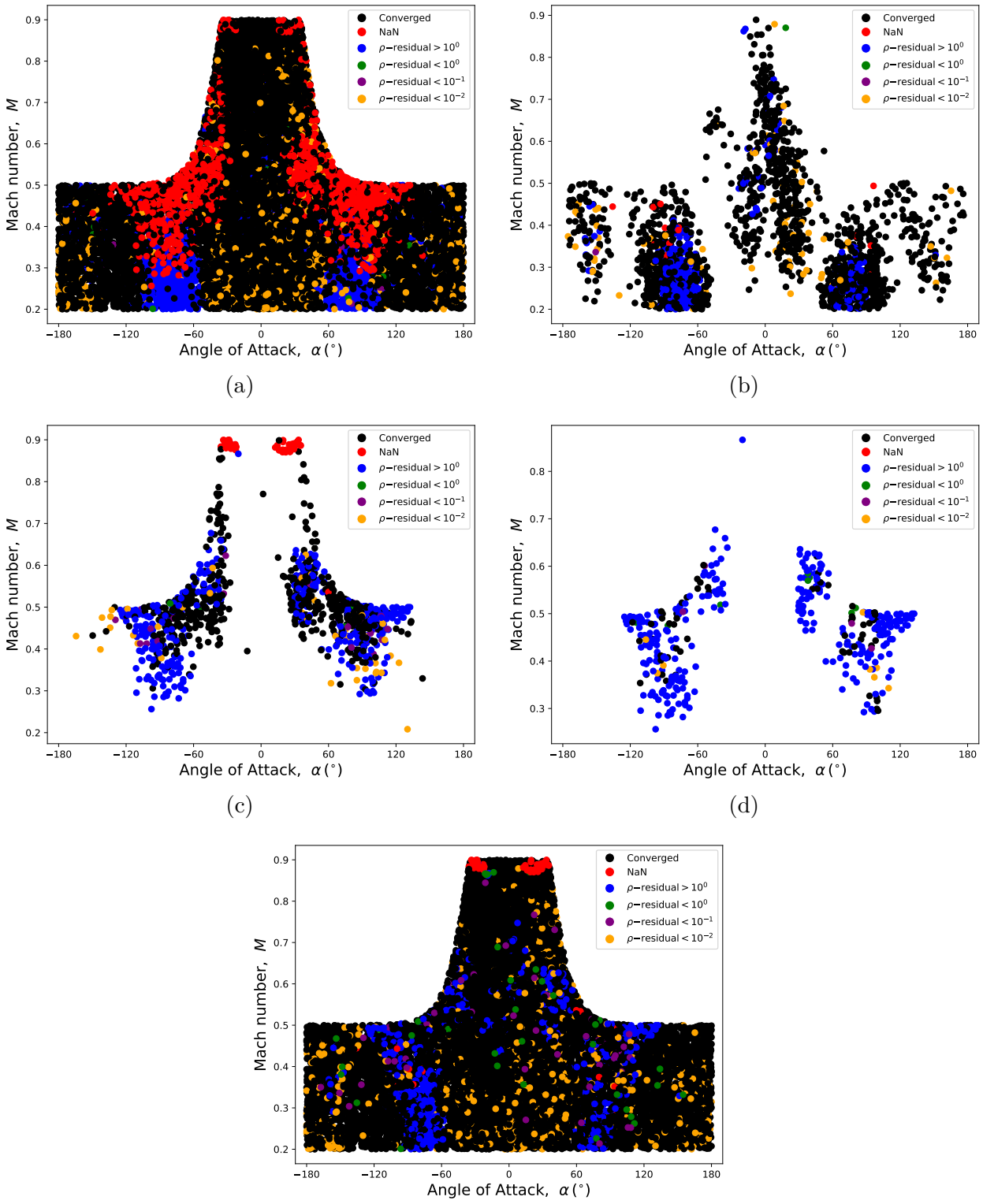


Figure 4.11 Bifurcation strategy applied to the originally simulated dataset in (a). When OLCs are detected in (a), SFD is applied, yielding (b). When NaNs are present in (a), solver parameters are adjusted, resulting in (c). If OLCs are subsequently detected in (c), SFD is then applied, producing (d). The aggregate is presented in (e), where the final converged dataset is comprised of simulation cases C_1 and C_6.

The database used for developing the AI-CFD surrogate model required approximately 16.83 core-years to generate a total of 16,541 simulation data points. Of these, 15,807 data points converged successfully and were subsequently useable in ML training, validation, and testing. This indicates that the adopted simulation strategy required approximately 1.06×10^{-3} core-years per converged data point. As is standard practice in ML workflows, the dataset is randomly split into subsets for training (80%), validation (10%), and testing (10%). The validation set is used for hyperparameter tuning and optimization, while the testing set is used for unbiased model evaluation.

Furthermore, as detailed in Appendix B, an additional dataset is generated exclusively for surrogate model evaluation. This dataset consists of 4-digit NACA airfoils evaluated under full-factorial combinations of freestream flow conditions spanning the entire problem domain. Including full-factorial samples across angle-of-attack values enables the generation of aerodynamic coefficient – α curves, which are commonly used to illustrate geometric performance. The NACA dataset comprises 1,098 samples, of which 1,027 simulations converged, requiring approximately 1.1 core-years to generate. This dataset is used to evaluate model generalization towards an airfoil geometry type not seen during training.

4.6 Data Processing

Data processing is essential in supervised ML training, as raw data is often ill-suited for effective learning. For each data point, the unprocessed input vector $\mathbf{x}$, defined in Equation 4.18, includes the freestream flow conditions, angle of attack α, Mach number M, and Reynolds number Re, along with the airfoil geometry represented by n Cartesian coordinates in the Selig format. In this format, the airfoil's x- and y-coordinates begin at the upper surface trailing edge, wrap around the leading edge, and end at the lower surface trailing edge. However, this unprocessed input vector poses several challenges for ML training. First, representing airfoil geometries through n Cartesian coordinates introduces a high-dimensional input space, which can hinder model training. Additionally, the number n and distribution of coordinates will vary greatly across unseen geometries, reducing the model's ability to generalize.

$$\mathbf{x} = [\alpha,\, M,\, Re,\, x_1,\, y_1,\, x_2,\, y_2,\, \ldots,\, x_n,\, y_n]^T \tag{4.18}$$

Furthermore, the unprocessed output vector $\mathbf{y}$, defined in Equation 4.19, contains the aerodynamic coefficients corresponding to its input vector $\mathbf{x}$. Unlike $\mathbf{x}$, the output vector $\mathbf{y}$ does not present generalization issues. However, significant scale differences exist among the aerodynamic coefficients, necessitating an appropriate normalization strategy to ensure effective

ML training. Consequently, the output vector $\mathbf{y}$ must also be inverse-scaled after model inference to ensure properly scaled aerodynamic coefficient predictions.

$$\mathbf{y} = \begin{bmatrix} C_L \\ C_D \\ C_M \end{bmatrix} \tag{4.19}$$

4.6.1 Geometry

As previously mentioned, geometries generated using methods different from those in the database workflow can exhibit significant variations in both the distribution and number of Cartesian coordinates. To ensure generalization across diverse geometry generation techniques, a robust pre-processing strategy must be applied to the training data's geometries. This strategy must also be reproducible when handling unseen geometries.

The adopted pre-processing strategy begins by fitting a cubic spline through the original geometry, preserving C^2 continuity throughout subsequent operations. The geometry is then resampled to a fixed set of 513 coordinate points, ensuring a consistent input vector size, which simplifies the architecture of the surrogate model. Following resampling, the geometry is split into upper and lower surfaces, where hyperbolic tangent smoothing [47] is applied to each to achieve a consistent point distribution. This smoothing respectively employs initial and end spacings of 2×10^{-3} chord at the leading and trailing edges.

4.6.2 Dimensionality Reduction

The technique employed to address the high dimensionality of the input dataset $\mathbf{X}$ is Principal Component Analysis (PCA) [48]. This process is crucial, as input vectors, following geometry smoothing, have an immense dimension size of 1029×1. PCA reduces dimensionality by projecting the dataset onto a lower-dimensional space that captures its most significant structures. It identifies directions (principal components) along which the data exhibits the highest variance, highlighting the most informative features. Furthermore, PCA aims to reduce dataset feature correlation by applying orthogonal transformations to the dataset. An illustration of the orthogonal axis transformations used in PCA is presented in Figure 4.12.

To apply PCA, the covariance matrix of the dataset must first be computed. The eigenvectors and eigenvalues of this matrix are then determined, where the eigenvectors define the directions of the feature space, and the eigenvalues quantify the variance along each corresponding direction. The top k eigenvalues are selected, and their associated eigenvectors are

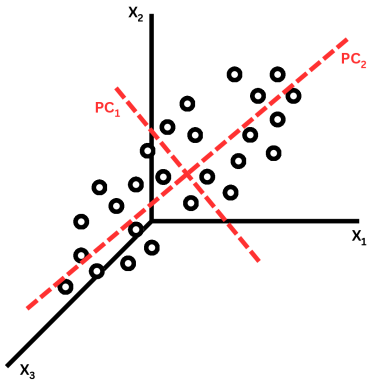


Figure 4.12 Illustration of principal component analysis (PCA) principle

assembled into the matrix $\mathbf{W}_k$. The transformed dataset, $\hat{\mathbf{X}}_k$, is obtained by projecting the original dataset onto the selected eigenvectors, as shown in Equation 4.20.

$$\hat{\mathbf{X}}_k = \mathbf{X}\mathbf{W}_k \tag{4.20}$$

The result of applying PCA to the dataset $\mathbf{X}$ is shown in Figure 4.13, illustrating how the explained variance increases with the number of principal components. As depicted, the majority of the dataset's total variance is captured by the first few principal components, indicating that only a small number of dimensions are needed to effectively represent the data. The PCA applied in this instance offers a reduction in dimensionality as well as feature correlation, thereby facilitating more efficient and effective machine learning training.

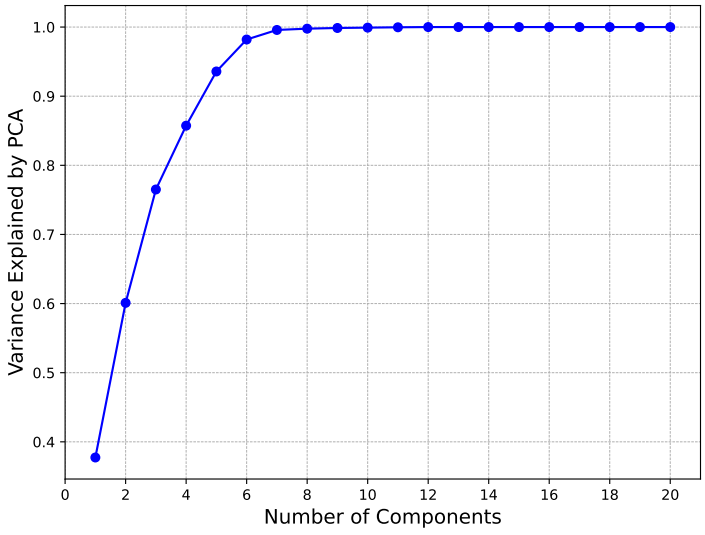


Figure 4.13 Effectiveness of PCA for an increasing number of components

4.6.3 Normalization

Following geometry smoothing and input data dimensionality reduction, the input and output datasets, $\mathbf{X}$ and $\mathbf{Y}$, need to be normalized as the scales of their features are found to vary greatly. The computed components from PCA exhibit considerable differences in magnitude, where input variable scales range from values near zero to those exceeding 10^7, leading to imbalances that can negatively impact the effectiveness of gradient-based methods used in ML training. Common choices for normalization techniques are applied to the input and output datasets. The input dataset is normalized using min-max scaling, where each feature is rescaled based on its minimum and maximum values, as described in Equation 4.21, while the output dataset undergoes z-score standardization, with each output variable normalized using its mean and standard deviation, as detailed in Equation 4.21.

$$x' = \frac{x - x_{\min}}{x_{\max} - x_{\min}} \tag{4.21}$$

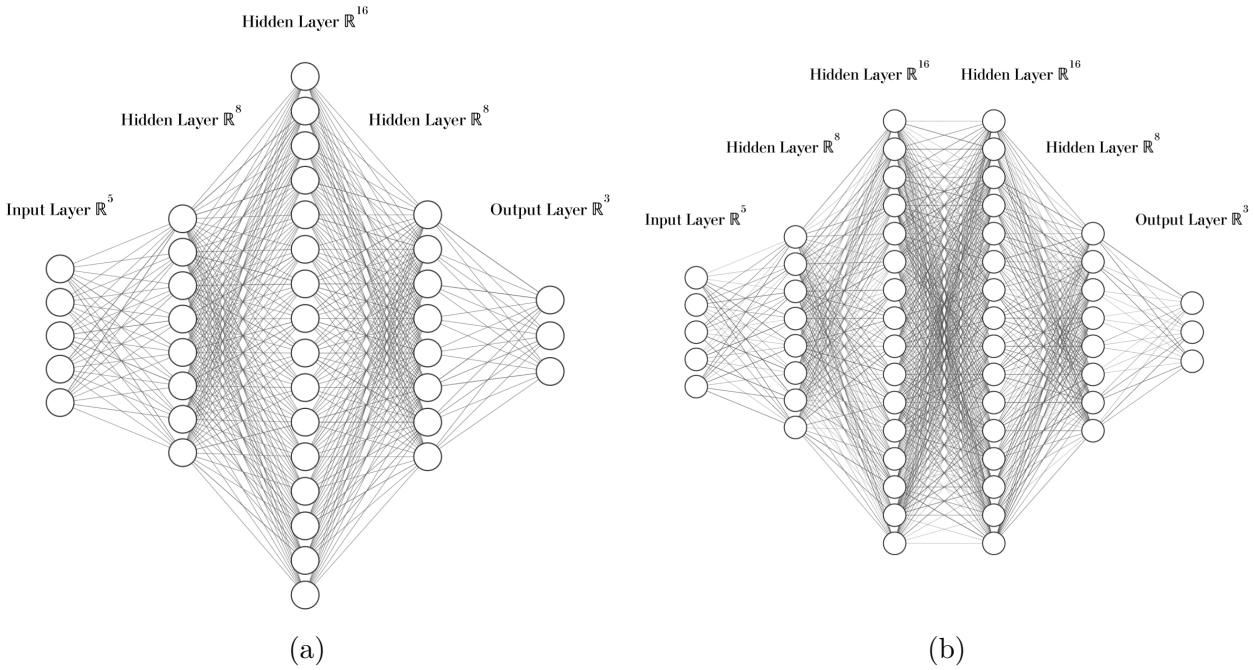
$$y' = \frac{y - \mu}{\sigma} \tag{4.22}$$

4.7 Machine Learning

4.7.1 Architecture

As highlighted by the literature review section, the MLP architecture is the most appropriate for this surrogate modeling task. However, this selection leaves several key decisions, including the number of input PCA components, the number of hidden layers, and the number of neurons or basic units in each layer. While the specific values for these parameters will be determined through optimization in a later section, the overall architecture is first established. The hidden layer sizing follows a structured approach, referred to herein as the *diffuser-nozzle* sizing scheme. This sizing scheme, illustrated in Figure 4.14, features a greater number of neurons in the middle hidden layers of the MLP model. The *diffuser-nozzle* scheme aligns with prior work on architectures such as U-Net [49], which utilize middle layers with increased complexity compared to the preceding and succeeding layers. This design serves multiple purposes. In the diffuser portion (early layers), the model captures low-level patterns, while the *nozzle* portion (later layers) serves as a bottleneck, forcing an appropriate gradual transformation of features before reaching the output layer. Wider middle hidden layers permit a richer feature extraction, capturing more abstract and higher-level features and allowing for greater complexity in model representation. For simplicity in subsequent optimization tasks, the *diffuser-nozzle* sizing scheme is streamlined: the number of neurons

per hidden layer doubles until reaching the middle layer(s), after which it is progressively halved until the output layer.



(a) (b)

Figure 4.14 MLP architecture with *diffuser-nozzle* sizing scheme for (a) odd and (b) even number of hidden layers

The developed model is designed to simultaneously predict C_L, C_D, and C_M. Previous studies have explored both approaches: models that make separate predictions [24] and those that generate unified predictions of aerodynamic coefficients [27]. Heuristic analyses conducted in this study found no significant difference between these approaches when trained on the generated AI-CFD database. Therefore, a simultaneous prediction architecture was chosen, prioritizing simplicity in development.

Furthermore, the selection of activation functions is a critical step in configuring the MLP architecture. Non-linear activation functions are essential components of MLPs, enabling them to model complex non-linear relationships between inputs and outputs. In this study, the tested non-linear activation functions were limited to those commonly used in the machine learning community, including the hyperbolic tangent, sigmoid, Rectified Linear Unit (ReLU), and Exponential Linear Unit (ELU) functions [50]. These functions were evaluated heuristically in preliminary studies, where ReLU and ELU yielded the best training dynamics. ReLU demonstrated efficient training performance, while ELU provided a good balance between training efficiency and representational capacity, particularly in architectures with a relatively larger number of hidden layers.

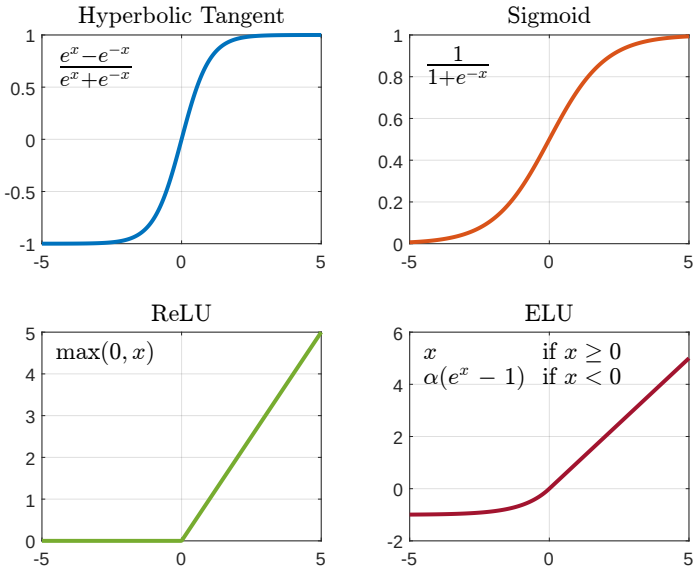


Figure 4.15 Heuristically evaluated activation functions

4.7.2 Training

In this study, machine learning development is implemented using the PyTorch library [51]. The model architectures and training optimization processes leverage the functions and packages available within this framework. The primary objective of training optimization is to refine the model parameters, specifically the weights and biases, in order to converge to a trained model with satisfactory inference performance. Training is conducted using the backpropagation algorithm [52], the *de facto* standard in the machine learning community. Compared to alternative approaches, such as the Levenberg-Marquardt algorithm [8], which has been employed in earlier works [53], backpropagation is highly effective for deep learning applications due to its scalability to networks with multiple hidden layers.

The efficiency of backpropagation stems from its use of the chain rule of differentiation. Specifically, the gradient of the loss function with respect to the model parameters provides the necessary updates during training. In this study, the Mean Squared Error (MSE) loss function, as defined in Equation 4.23, is a common choice for regression tasks. It is used to measure the squared L^2 norm between ground truth values and model predictions.

$$\mathcal{L} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 \tag{4.23}$$

For a randomly sampled batch of CFD data points, the stochastic gradient, given by Equation 4.24, can be computed efficiently for all model parameters. This efficiency arises from

the differentiability of neural network components, which consist of weighted sums, activation functions, and subsequent transformations leading to model outputs. The gradient of the loss function with respect to a given parameter follows naturally from its composition within these operations via the chain rule.

$$\boldsymbol{g} = \nabla_\theta \mathcal{L}_\theta(x) \tag{4.24}$$

After initializing model weights using the Kaiming He initialization scheme [54], the AdEMAMix optimizer [55] described in Equation 4.25 is employed for batch-wise parameter updates. This momentum-based optimizer extends AdamW [56], which itself is a correction to the widely used Adam optimizer [16]. Momentum-based optimizers leverage an Exponential Moving Average (EMA) of gradients, where older gradients have an exponentially decaying contribution on present parameter updates. AdEMAMix introduces an additional EMA moment, $\boldsymbol{m}_2$, alongside the first and second moments of AdamW, respectively $\boldsymbol{m}_1$ and $\boldsymbol{\nu}$. This additional EMA in AdEMAMix allows for simultaneous importance to be placed on recent and older gradients, something EMA in Adam or AdamW struggle with, given their single EMA. While this approach incurs higher memory overhead, it remains a negligible concern given the relatively small model sizes in this training application.

$$\begin{cases} \boldsymbol{m}_1^{(t)} = \beta_1 \boldsymbol{m}_1^{(t-1)} + (1-\beta_1)\boldsymbol{g}^{(t)}, \quad \hat{\boldsymbol{m}}_1^{(t)} = \frac{\boldsymbol{m}_1^{(t)}}{1-\beta_1^t}, \\ \boldsymbol{m}_2^{(t)} = \beta_3 \boldsymbol{m}_2^{(t-1)} + (1-\beta_3)\boldsymbol{g}^{(t)}, \\ \boldsymbol{\nu}^{(t)} = \beta_2 \boldsymbol{\nu}^{(t-1)} + (1-\beta_2){\boldsymbol{g}^{(t)}}^2, \quad \hat{\boldsymbol{\nu}}^{(t)} = \frac{\boldsymbol{\nu}^{(t)}}{1-\beta_2^t} \\ \boldsymbol{\theta}^{(t)} = \boldsymbol{\theta}^{(t-1)} - \eta \left(\frac{\hat{\boldsymbol{m}}_1^{(t)} + \alpha \boldsymbol{m}_2^{(t)}}{\sqrt{\hat{\boldsymbol{\nu}}^{(t)}} + \epsilon} + \lambda \boldsymbol{\theta}^{(t-1)} \right). \end{cases} \tag{4.25}$$

Nominal values are used for the decay coefficients, set as $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\beta_3 = 0.9999$, as well as for the $\boldsymbol{m}_2$ EMA moment factor, given by $\alpha = 8$. The parameter ϵ is also assigned a nominally small value of 10^{-8}. The learning rate η and weight decay λ are selected as a function of training dynamics evaluated heuristically.

An illustration of underfitting and overfitting regimes in the training dynamics of machine learning model development is shown in Figure 4.16. Model training performance is assessed based on loss measured over training epochs, where each epoch represents a complete pass through the training dataset using stochastic batch sampling. The figure presents example training and validation loss curves. Training loss is computed on the dataset used for model parameter updates, while validation loss is measured on an independent dataset to evaluate generalization to unseen data. The objective in machine learning development is to minimize

validation loss. Comparing training and validation loss curves helps diagnose whether the validation loss has been effectively minimized and detect signs of underfitting or overfitting.

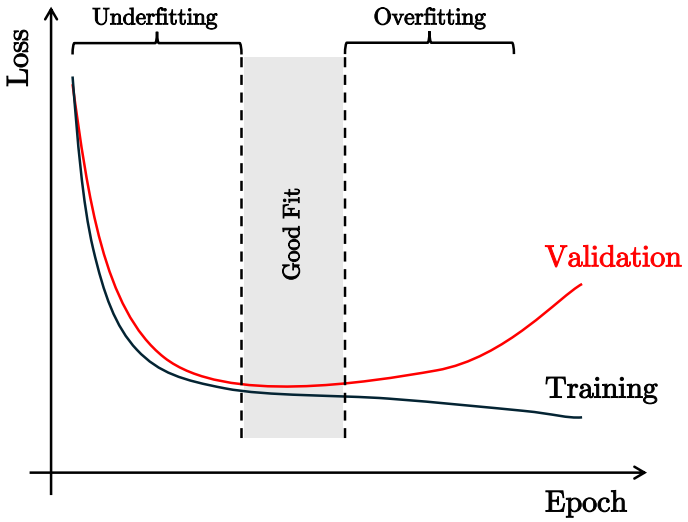


Figure 4.16 Example of machine learning training dynamics

Overfitting is detected when training performance significantly surpasses validation performance, indicating that the model is not only learning the underlying patterns in the training data but also memorizing noise and random fluctuations. Furthermore, overfitting is characterized by a decline in validation performance as training continues on new data points. Regularization techniques can be applied to mitigate overfitting. One approach is decoupled weight decay, parameterized by λ in the AdEMAMix optimizer, which penalizes large weight values by explicitly shrinking them after gradient-based updates. Heuristic analysis of training dynamics suggests that $\lambda = 10^{-5}$ effectively reduced overfitting. Another effective regularization technique is neuron dropout [57], where a fraction of neuron activations is randomly set to zero during training. By randomly setting a fraction of neuron activations to zero during training, noise is introduced, forcing the neural network to develop redundant representations, thereby reducing the influence of noise and fluctuations found in the dataset. A dropout fraction of 25% has been found to be effective in preventing overfitting. Additionally, early stopping is commonly used to limit overfitting. Training is halted prematurely if validation loss does not improve for $N = 2500$ epochs.

Underfitting is detected when both training and validation performance are poor and improve at a similar rate with additional training data. This indicates that the model has not fully captured the underlying patterns in the training dataset. Furthermore, underfitting often occurs when there is existing potential for performance improvement, suggesting that further training on additional data points may be beneficial. Several hyperparameters influence

underfitting. Insufficiently complex architectures can limit a model's capacity to learn, while excessively high regularization may suppress the learning of relevant patterns. In terms of training dynamics, the learning rate and batch size are selected as a function of training behavior. An initial $\eta = 10^{-3}$ paired with a stochastic batch size of 512 has been found to effectively prevent early-stage underfitting. To mitigate late-stage underfitting, a learning rate scheduler with step decay is employed. Large learning rates late in training can prevent convergence to a global minimum. A 5% learning rate decay every 250 epochs has been found effective in addressing underfitting when loss convergence plateaus.

Table 4.9 Summary of training hyperparameters

Hyperparameter	Value
Batch size samples	512
Learning rate	$\eta = 10^{-3}$
Learning rate step decay	$\eta_i \cdot 5\%$
Learning rate decay step size epochs	250
Decay coefficient for $\boldsymbol{m_1}$	$\beta_1 = 0.9$
Decay coefficient for $\boldsymbol{m_2}$	$\beta_2 = 0.999$
Decay coefficient for $\boldsymbol{\nu}$	$\beta_3 = 0.9999$
Scaling factor for $\boldsymbol{m_2}$	$\alpha = 8$
Weight decay factor	$\lambda = 10^{-5}$
Neuron dropout fraction	25%
Early stopping window epochs	2500
Max epochs	10000
Validation dataset random split	10%

4.7.3 Hyperparameter Optimization

In machine learning, training involves optimizing model parameters, while hyperparameter optimization focuses on tuning model architecture and parameters that govern the optimization process itself. The previous section discussed the heuristic selection of hyperparameters relating to training dynamics. This section shifts focus to the optimization of hyperparameters that define the model's architecture. The objective of this study is to develop a surrogate model for aerodynamic predictions that significantly accelerates computations compared to conventional CFD. To achieve this, it is essential to optimize models that balance low complexity with high predictive accuracy. Due to the discrete nature of hyperparameters and the absence of gradients, a black-box approach to hyperparameter optimization is an essential choice. The Oríon toolbox [58], developed at the Montreal Institute for Learning Algorithms (Mila), is specifically designed for the optimization of machine learning hyperparameters.

This toolbox supports various optimization algorithms, including grid search, evolutionary algorithms, Bayesian optimization, etc.

Using the Oríon toolbox, this study employs the included Tree-structured Parzen Estimator (TPE) algorithm [59] [60]. Bayesian optimization methods like TPE are generally efficient in guiding the search for optimal hyperparameter sets. This algorithm is specifically conceived for machine learning hyperparameter optimization and effectively handles categorical spaces. However, given the small number of hyperparameters in this study's optimization problem, as shown in Table 4.10, using TPE may be somewhat excessive. Nevertheless, it is included in the framework since future work will involve optimizing a higher-dimensional space with more categorical variables. For instance, while this study fixes the ReLU activation function across all hidden layers, future studies may allow different activation functions for individual hidden layers, introducing additional categorical variables.

Table 4.10 Hyperparameters decision variables for optimization

Hyperparameter	Range
Number of hidden layers	$\{N_l \in \mathbb{Z} \mid 1 \leq x \leq 4\}$
Number of input components	$\{N_i \in \mathbb{Z} \mid 3 \leq x \leq 20\}$
Number of neurons in first hidden layer	$\{N_n \in \mathbb{Z} \mid 32 \leq x \leq 512\}$

The TPE algorithm enables efficient sampling of hyperparameter configurations across the solution space of the objective function, where TPE employs an acquisition function that balances exploration and exploitation. This balance is crucial for identifying the Pareto front in multi-objective optimization problems, such as the presented application, where the goal is to optimize model configurations that achieve both low complexity and high predictive accuracy. Hence, TPE is used effectively in this study's hyperparameter optimization procedure, where its resulting Pareto front and models derived from it are presented and evaluated in the following results section.

CHAPTER 5 RESULTS AND DISCUSSION

This results section, summarized in Table 5.1, begins with the presentation of the Pareto front obtained from the hyperparameter optimization procedure, followed by the selection and fine-tuning of the best-performing model. This model is then evaluated on two distinct testing datasets. Additionally, the hyperparameter optimization procedure's top 20 models are evaluated on the same datasets to analyze variability in model performance. Finally, the computational cost and prediction speed of various surrogate model architectures are compared against CFD simulations to quantify the achieved prediction acceleration.

Table 5.1 Summary of results section

Section	Model(s)	Result
5.1	N/A	The Pareto front resulting from the hyperparameter optimization procedure is presented and analyzed.
5.2	N/A	Fine-tuning loss curves and the architecture of the best performing model are presented.
5.3.1	*Top*-1	Evaluation of the best fine-tuned model is performed on a testing dataset obtained from a random 80–10–10% split of the original training dataset. This assessment measures the model's ability to generalize to previously unseen geometries and freestream flow conditions.
5.3.2	*Top*-1	Evaluation of the best fine-tuned model is performed on a testing dataset composed of 4-digit NACA airfoils. Since the training dataset exclusively contains NURBS-generated airfoils, this evaluation assesses the model's ability to generalize to a distinct class of airfoil geometries.
5.4	*Top*-20	The top 20 models obtained from the hyperparameter optimization procedure are evaluated on both the random 10% test split and the NACA testing datasets to assess performance variability resulting from training.
5.5	*Top*-1	The computational cost of CFD simulations is compared to that of the best ML surrogate model's predictions.

5.1 Pareto Front

The multi-objective function used in this study's hyperparameter optimization procedure is defined in Equation 5.1, where the objectives are equally weighted ($w_1 = w_2$). The two objectives, $J_l(x)$ and $J_c(x)$, represent prediction performance and model complexity, respectively. The loss objective, $J_l(x)$, described in Equation 5.2, is a scaled measure of the best recorded validation loss, $\mathcal{L}_{\text{val}}$, across a surrogate model's training epochs for a given hyperparameter configuration. The complexity objective, $J_c(x)$, described in Equation 5.3, is a scaled representation of the total number of neural connections, N_{con}, resulting from the same hyperparameter configuration.

$$f(x) = w_1 \cdot J_l(x) + w_2 \cdot J_c(x) \tag{5.1}$$

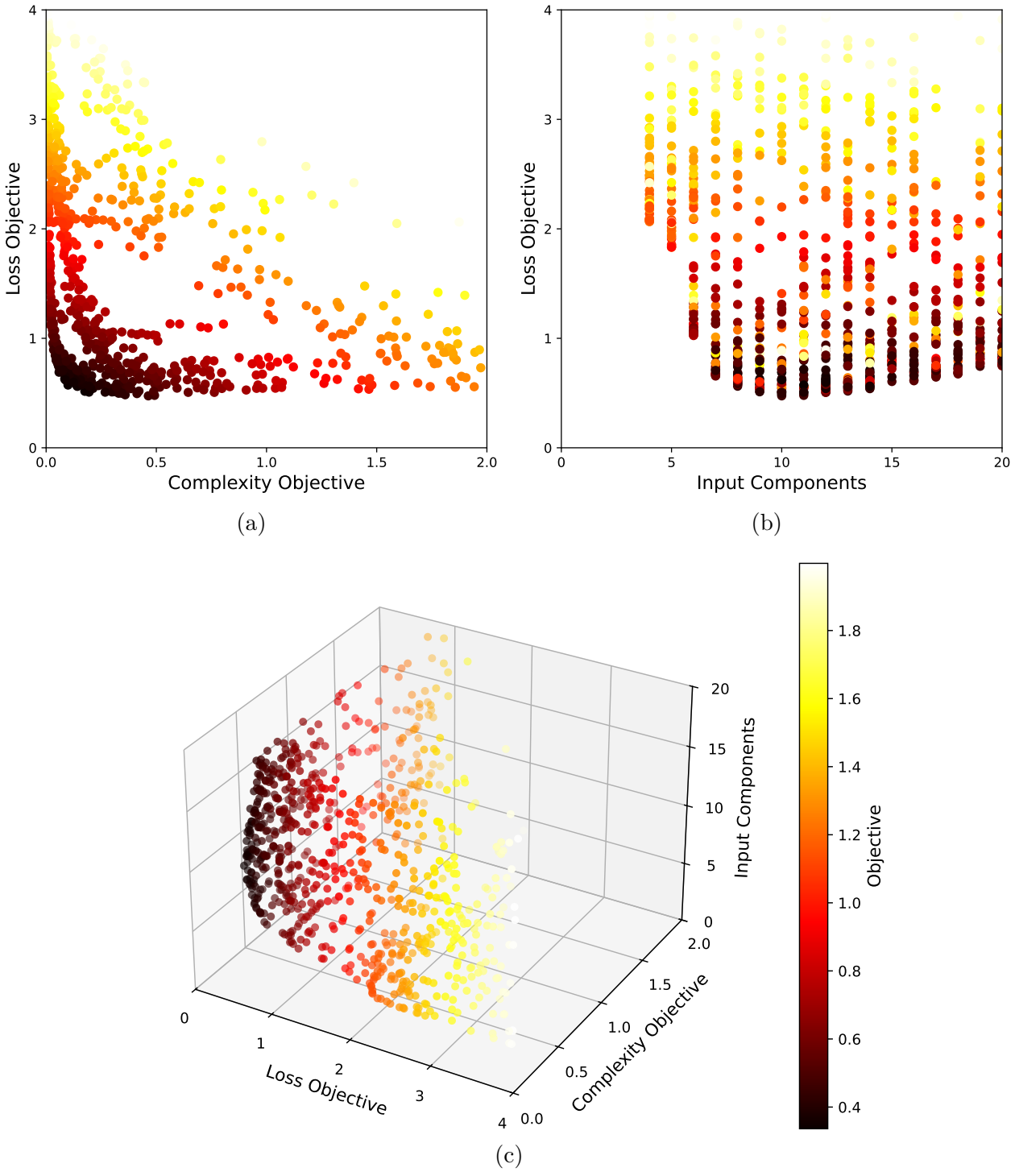
$$J_l(x) = \frac{\mathcal{L}_{\text{val}}}{5 \times 10^{-4}} \tag{5.2}$$

$$J_c(x) = \frac{N_{\text{con}}}{5 \times 10^{5}} \tag{5.3}$$

In this experiment, 50 randomly selected initial hyperparameter configurations were evaluated to construct the TPE algorithm's initial kernel density estimators, followed by 950 TPE sampled configurations. The resulting multi-objective optimization experiment produced the Pareto fronts illustrated in Figure 5.1. In these figures, low-performing configurations ($f(x) > 2$) are pruned to focus on optimal solutions. The Pareto front illustrates the trade-off between model complexity and prediction performance, demonstrating that beyond a certain threshold, increasing model complexity does not necessarily improve predictive accuracy for the given CFD dataset. Figure 5.2 further highlights this trade-off by labeling model configurations according to their number of hidden layers. The results indicate that a single hidden layer fails to capture the dataset's underlying patterns, while four hidden layers provide no significant performance gains. However, the early stopping mechanism used to prevent overfitting may have been too aggressive, potentially limiting the performance of larger models that require more training epochs for convergence.

5.2 Fine Tuning

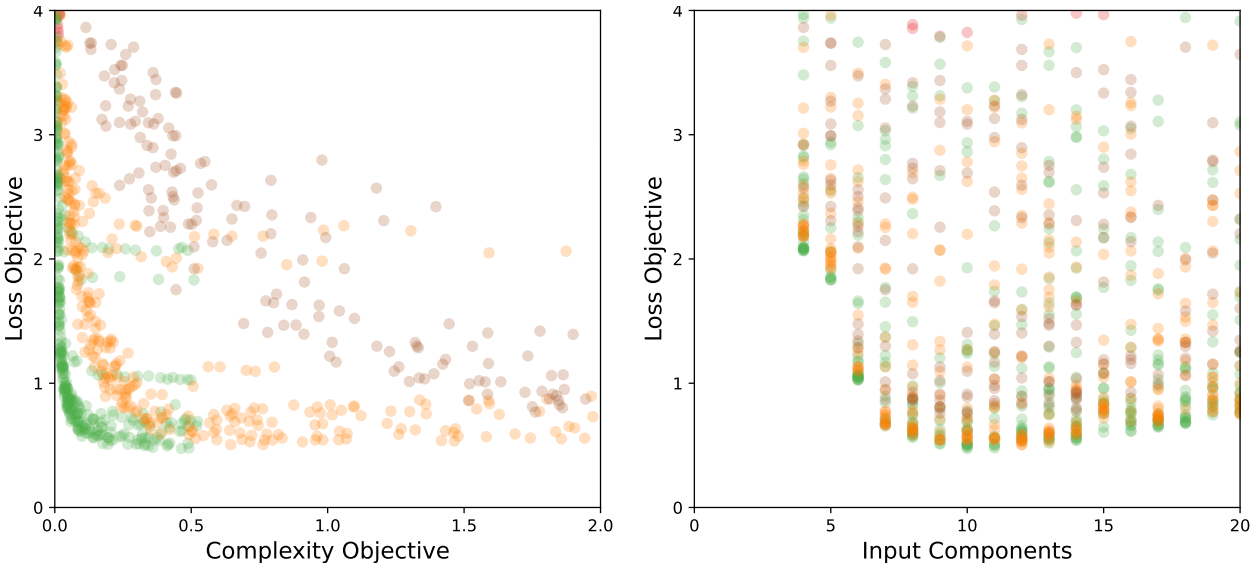
Fine-tuning beyond the hyperparameter optimization process brings forward the model that will be used in the subsequent results section. While hyperparameter optimization identifies highly effective models for a given application, further fine-tuning, guided by trends observed during the optimization process, can still yield slight improvements. The Pareto front de-



(a) (b)

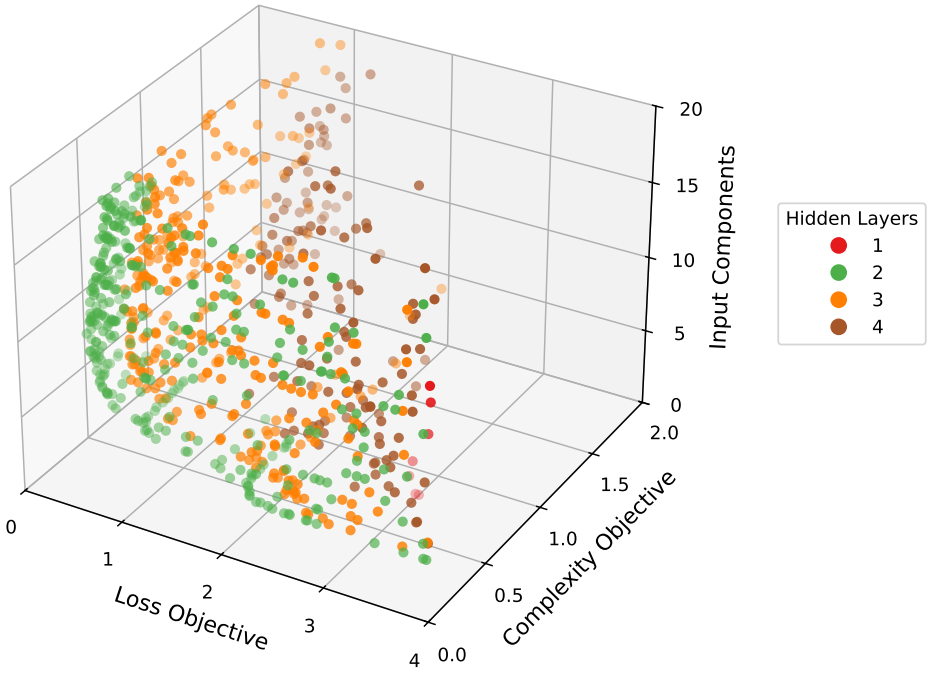
(c)

Figure 5.1 Pareto front for multi-objective



(a)

(b)



(c)

Figure 5.2 Pareto front for multi-objective across number of hidden layers

rived from hyperparameter optimization suggests that using 10 input components resulting from the PCA pre-processing is optimal. Additionally, it indicates no significant predictive performance gains beyond a model complexity objective of $J(x)_c > 0.5$. Furthermore, models with either one or four hidden layers underperform given the generated CFD database.

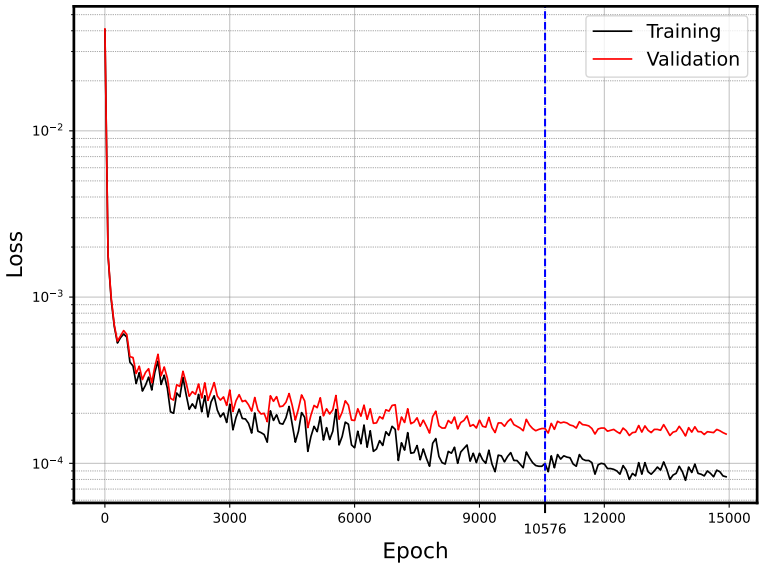


Figure 5.3 Training and validation loss curves of the best-performing model architecture

The best model configuration is fine-tuned and manually assessed to ensure training convergence while mitigating overfitting, as illustrated in Figure 5.3, where the best validation loss and corresponding model parameters are recorded at the $10{,}576^{th}$ epoch. This model has a complexity objective of approximately $J(x)_c \approx 0.5$ and utilizes 10 input components. Furthermore, ELU activation functions are used across all hidden layers. Heuristics in this stage indicate that ELU enhances representational capacity compared to ReLU but at the cost of reduced training dynamics. The resulting three hidden-layer model with 265,472 neural connections, detailed in Table 5.2, and yields the lowest validation loss among all configurations tested. Consequently, this model is selected for predictive performance evaluation in the results section.

5.3 Best Model Evaluation

The best developed model remains to be evaluated on testing datasets. While the validation dataset, used to assess model performance during training and hyperparameter optimization, gives a sense of the model's ability to generalize to unseen data, it cannot be used in the model's evaluation. Since model architecture and training parameters are selected based on

Table 5.2 Fine-tuned architecture used in results section

Layers	Configuration	Activation Function
L_1	10 PCA input components	–
L_2	256 neurons	ELU
L_3	512 neurons	ELU
L_4	256 neurons	ELU
L_5	3 outputs (C_L, C_D, C_M)	–

validation performance, using the validation dataset to evaluate the model's accuracy and generalization ability would introduce bias. Hence, completely unseen testing datasets are generated specifically to evaluate the model.

Two testing datasets are used in the model's evaluation. The first consists of a 10% random split from the initial dataset generated with the workflow described in the methodology section. This dataset serves to evaluate the model's predictive accuracy and its ability to generalize to unseen geometries and freestream flow conditions. The second dataset, described in Appendix B, contains the NACA 0012, NACA 2416, and NACA 4608 airfoil geometries. While it also evaluates predictive accuracy and generalization, it does so for geometries generated using a method different from the one described in the methodology section.

5.3.1 Best Model Performance on Random 10% Split Testing Set

The best model is first evaluated on the 10% random split testing dataset. Of the 15,807 data points generated from the workflow described in the methodology section, 1580 are used in this testing dataset to evaluate model accuracy and generalization ability. Each data point represents a combination of a unique and unseen geometry and freestream flow conditions. Furthermore, given the large random sampling, the envelope of tested geometries and flow conditions is very similar to the initially sampled envelope.

Table 5.3 Summary of model inference error on random 10% split testing set

	MAE	RMSE	|max|	R^2
C_L	0.0302	0.0479	0.4048	0.9962
C_D	0.0115	0.0163	0.1451	0.9987
C_M	0.0128	0.0212	0.1943	0.9958

Table 5.3 presents the predictive performance of the developed surrogate model on the 10% randomly split testing dataset. The evaluated Coefficient of Determination (R^2) for C_L, C_D,

and C_M suggests that a significant proportion of their variance is predictable from the model's inputs. High R^2 values on the testing dataset indicate that the model has effectively captured underlying patterns in the training data without substantial overfitting, demonstrating model generalization toward unseen data. However, the Mean Absolute Error (MAE) for C_L, C_D, and C_M remains substantial. These error magnitudes exceed the tolerances typically required in aerodynamic design, where errors are measured in lift, drag, or moment *counts* (1×10^{-4}). In the context of flight simulators, these error levels are most likely acceptable, although their suitability would ultimately be at the discretion of the flight simulator integration engineer.

Furthermore, Table 5.3 reports the absolute maximum prediction errors (|max|) for C_L, C_D, and C_M, highlighting that the model has the potential to produce significant errors for certain

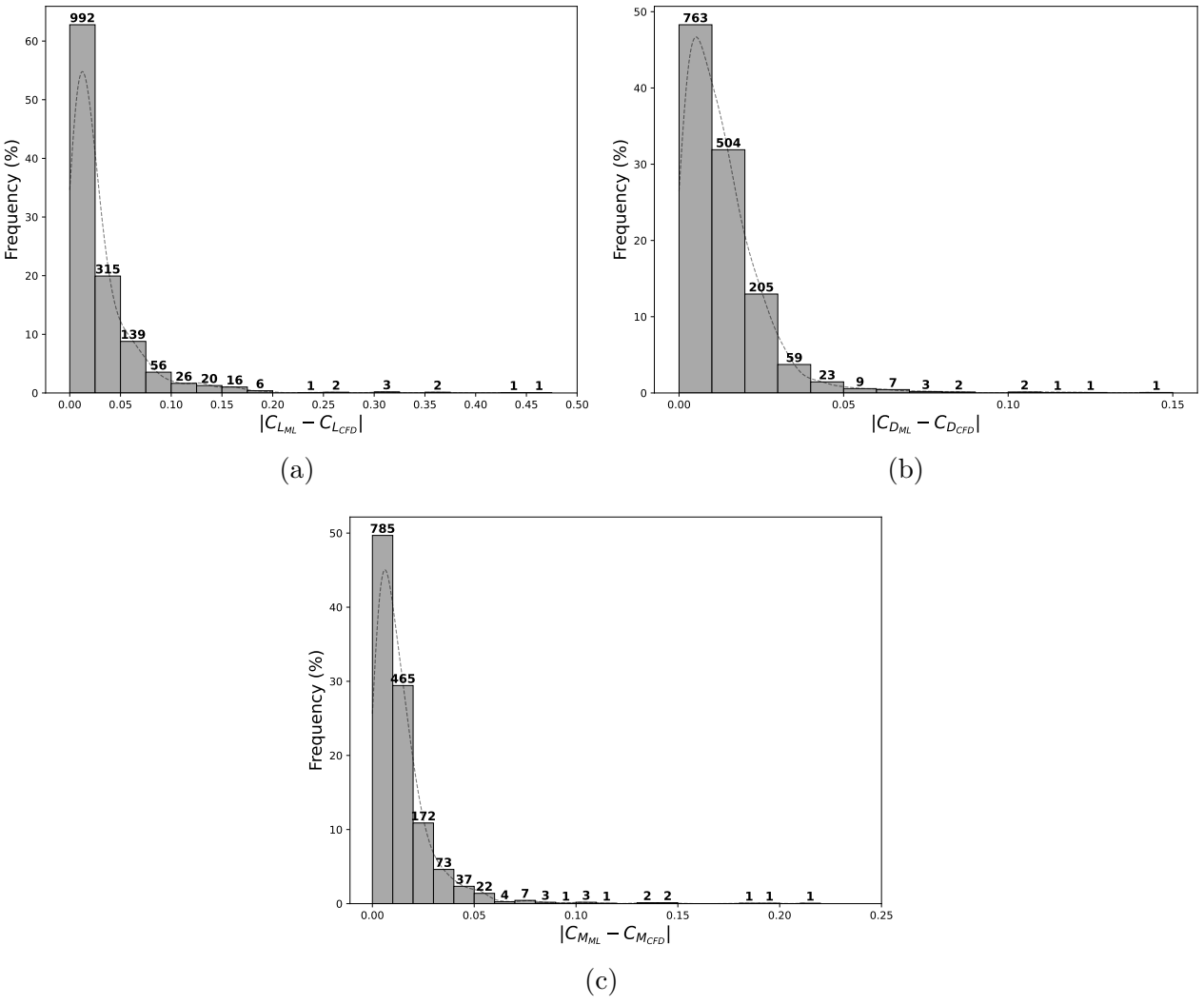


Figure 5.4 Histograms for model prediction error for (a) C_L, (b) C_D, and (c) C_M with respect to 10% random split testing dataset

input configurations. Figure 5.4 depicts the frequency of large errors, showing that while such errors do occur, they are relatively rare across the entire testing dataset. Again, while most likely acceptable for flight simulators, the suitability of such maximum errors would be at the discretion of the flight simulation integration engineer.

Large errors in model predictions follow trends across the domain envelope of input configurations. Figure 5.5 presents parity plots comparing model predictions with CFD values for C_L, C_D, and C_M. As seen in the parity plots for C_L and C_M, predictions of larger magnitudes tend to correspond to higher errors, as indicated by their greater deviation from the ideal parity line ($y = \hat{y}$). Additionally, these high-magnitude predictions occur in sparsely populated regions of the parity plots, suggesting that the training dataset underrepresents the underlying patterns in these regions, contributing to higher errors.

In contrast, the C_D parity plot shows a more uniform error distribution across magnitudes, with no clear correlation between prediction magnitude and error. This indicates a more consistent error spread across cases and relatively small errors, even for large-magnitude predictions. Such behavior suggests underlying patterns across the magnitude spectrum of C_D are effectively represented in the training dataset.

Furthermore, Figure 5.5 presents colormaps of absolute errors across the M–α envelope, showing consistently large errors in high Mach number regions for C_L, C_D, and C_M. As discussed in the methodology section, a significant portion of data points in this region were pruned due to divergent simulations, resulting in a notable sparsity compared to other areas of the CFD database. This sparsity likely contributes to the increased error, suggesting that the training dataset underrepresents the underlying patterns in these regions. Moreover, high Mach regimes are often characterized by highly nonlinear and discontinuous flow phenomena, further complicating the accurate modeling of trends.

Similarly, significant errors in C_L and C_M are observed in the $\alpha \in [-30, 30]^\circ$ and $\alpha \in [-180, -150] \cup [150, 180]^\circ$ regions of the M–α envelope. Unlike the high Mach region discussed earlier, which suffered from sparse data due to pruning, these angle of attack ranges are well represented in the training dataset. Nevertheless, these are also the regions where the onset of stall typically occurs in pitching airfoils, a phenomenon marked by its complexity and discontinuous behavior. Although only a few data points were pruned here, the discontinuous nature of stall makes it challenging to capture the underlying patterns during training. Moreover, neural networks like those developed in this study cannot explicitly model discontinuous functions, as the included activation functions must be differentiable for gradient-based optimization. Consequently, deeper architectures, which approximate such discontinuities through compositions of smooth functions, may introduce significant errors.

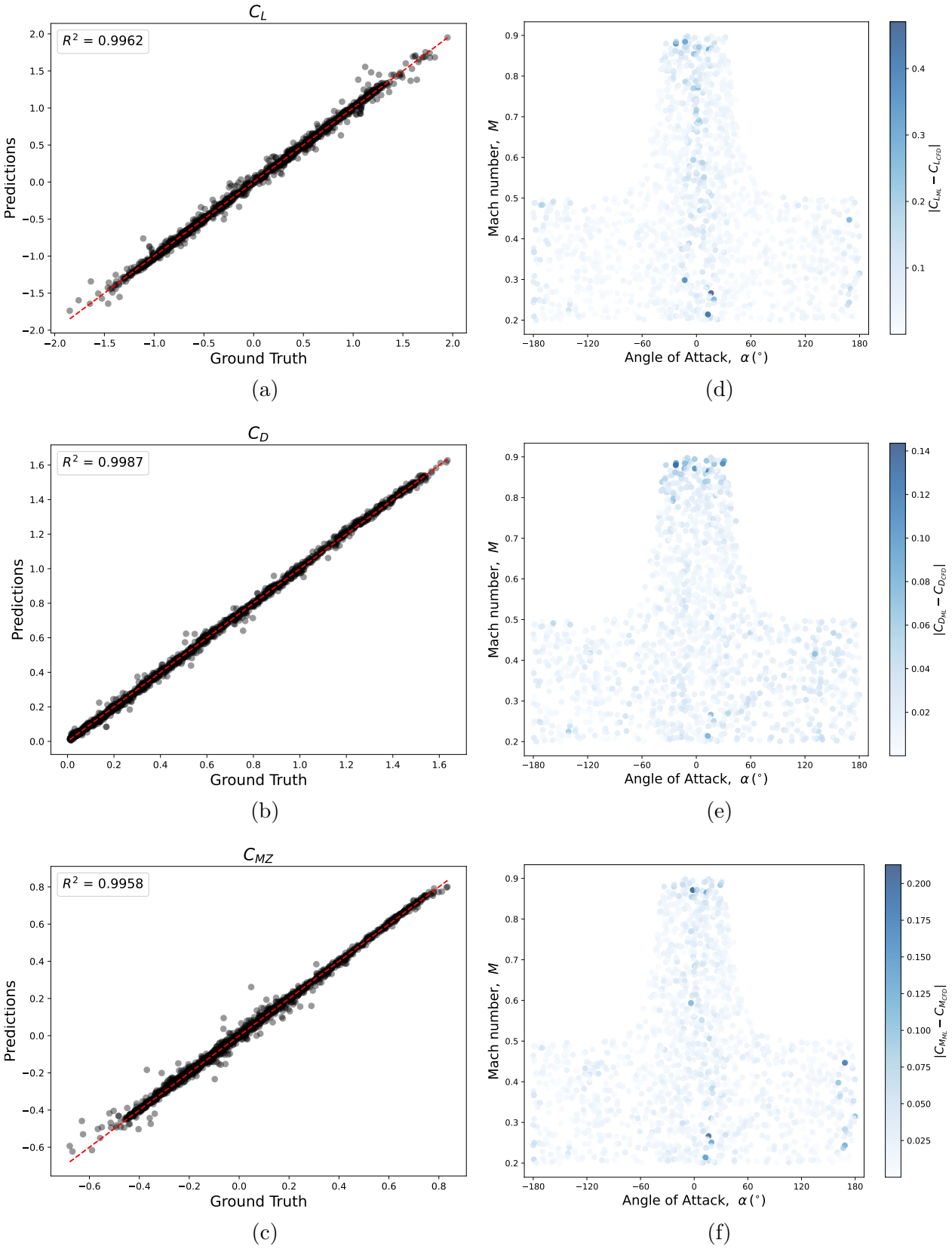


Figure 5.5 Parity plots for (a) C_L, (b) C_D, and (c) C_M model predictions with dashed red line representing ideal parity line ($y = \hat{y}$) and model absolute error for (d) C_L, (e) C_D, and (f) C_M across $M - \alpha$ envelope for 10% random split testing dataset

5.3.2 Best Model Performance on NACA Testing Set

The best developed model is also evaluated on a separate testing dataset containing the NACA 0012, NACA 2416, and NACA 4608 airfoil geometries. This additional testing dataset covers the envelope of freestream flow conditions contained within the datasets generated through the workflow described in the methodology section. However, this NACA dataset serves the additional purpose of evaluating the developed model's generalization ability toward geometries generated with a method unlike the one used in the training workflow. This dataset, specifically included for testing generalization, contains 1027 data points.

Table 5.4 Summary of model inference error on NACA testing set

	MAE	RMSE	|max|	R^2
C_L	0.0405	0.0612	0.4443	0.9928
C_D	0.0156	0.0203	0.0967	0.9984
C_M	0.0228	0.0312	0.1731	0.9906

Table 5.4 summarizes the model's predictive performance on the NACA testing dataset. The high R^2 values for C_L, C_D, and C_M demonstrate that the model has the capacity to generalize to unseen geometries generated with a different approach than that used for training. However, the higher R^2 values obtained from the 10% random split dataset indicate that the model's performance is diminished on the NACA dataset, suggesting that it is overfitted to the NURBS airfoils present in the training data.

Consequently, the MAE for C_L, C_D, and C_M on the NACA test dataset is significantly higher than that on the 10% random split dataset. This increased error suggests that the model's generalization to new geometric types can be further improved. One potential improvement is expanding the training dataset to include geometries generated using multiple methods. Another avenue is adopting a more advanced geometric parameterization approach. In the current framework, cubic spline smoothing and PCA-based transformations are used for parameterization. However, despite the cubic spline smoothing, the method remains sensitive to variations in Cartesian coordinate distributions. Future research should explore alternative approaches that aim to achieve a similar level of dimensionality reduction as PCA while being less sensitive to coordinate distribution along the geometry.

Furthermore, Table 5.4 also presents the absolute maximum prediction errors for C_L, C_D, and C_M in the NACA testing dataset, demonstrating that, similar to the 10% random split dataset, certain input configurations can lead to very large errors. Figure 5.6 illustrates the frequency of absolute errors, showing that while large errors do occur, they remain relatively

rare. However, compared to the 10% random split dataset, smaller errors are more infrequent, and moderate errors are more frequent, as reflected in the increased MAE and variance, which shift the absolute error distribution towards larger values.

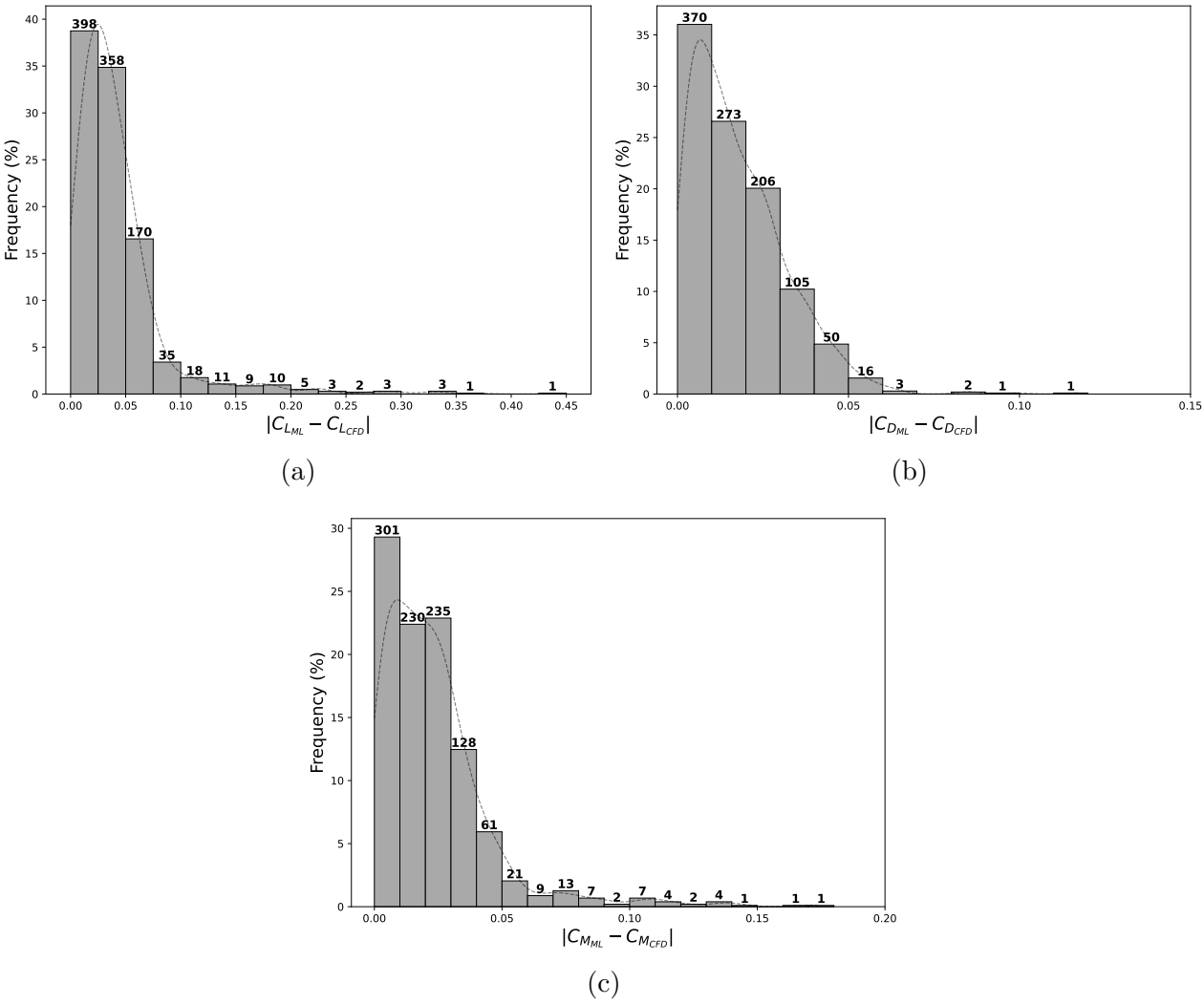


Figure 5.6 Histograms for model prediction error for (a) C_L, (b) C_D, and (c) C_M with respect to NACA testing dataset

Similar trends in model prediction error are observed across the NACA testing dataset, consistent with those seen in the 10% random split dataset. Figure 5.7 displays parity plots and absolute error colormaps, comparing model predictions with CFD values for C_L, C_D, and C_M in the NACA testing dataset.

As seen in the parity plots for C_L and C_M in the NACA dataset, the model struggles to generalize to unseen airfoil geometries. Similar to the trend observed in the 10% random split dataset, but to a greater extent, predictions with larger magnitudes tend to exhibit

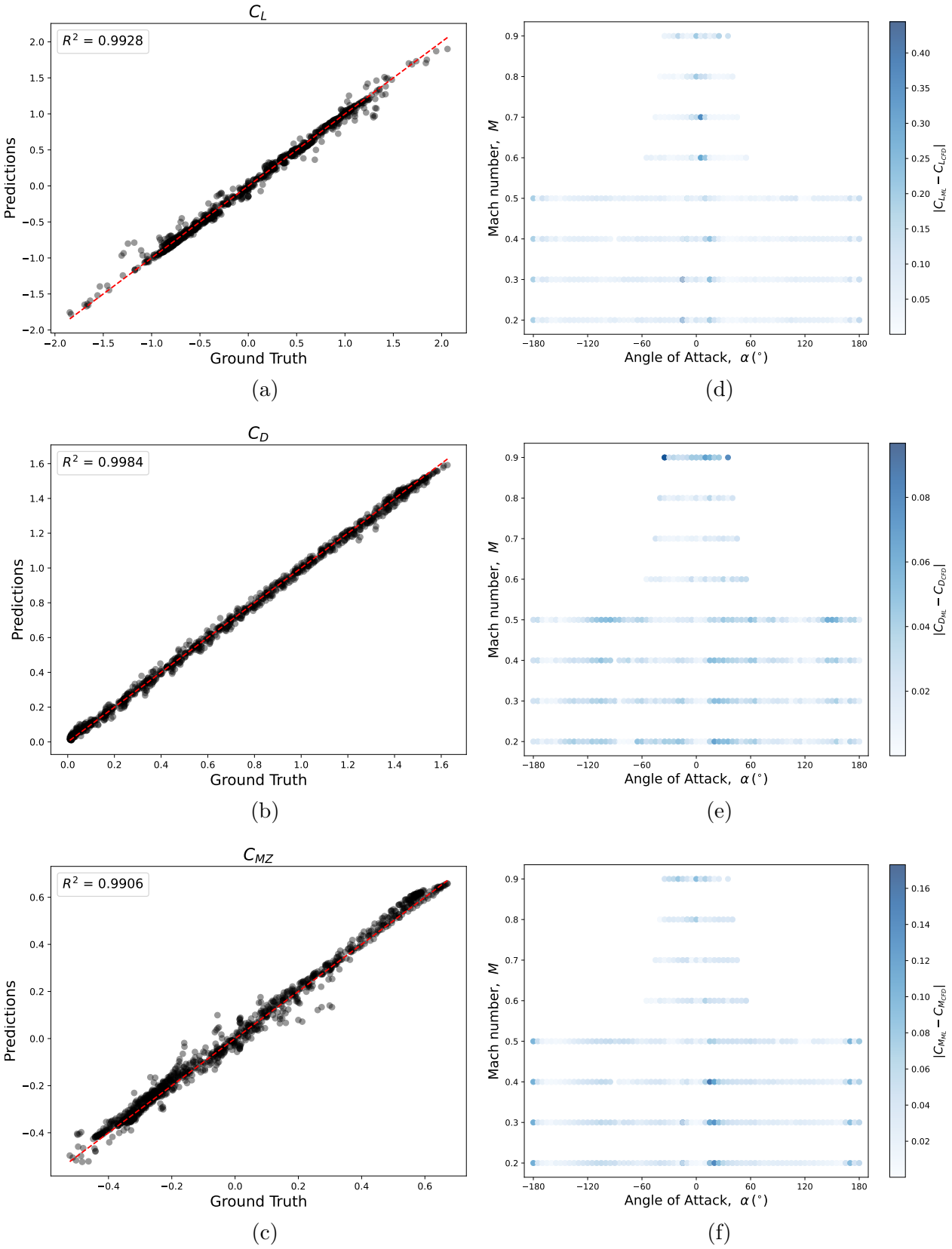


Figure 5.7 Parity plots for (a) C_L, (b) C_D, and (c) C_M model predictions with dashed red line representing ideal parity line ($y = \hat{y}$) and model absolute error for (d) C_L, (e) C_D, and (f) C_M across $M - \alpha$ envelope for NACA testing dataset

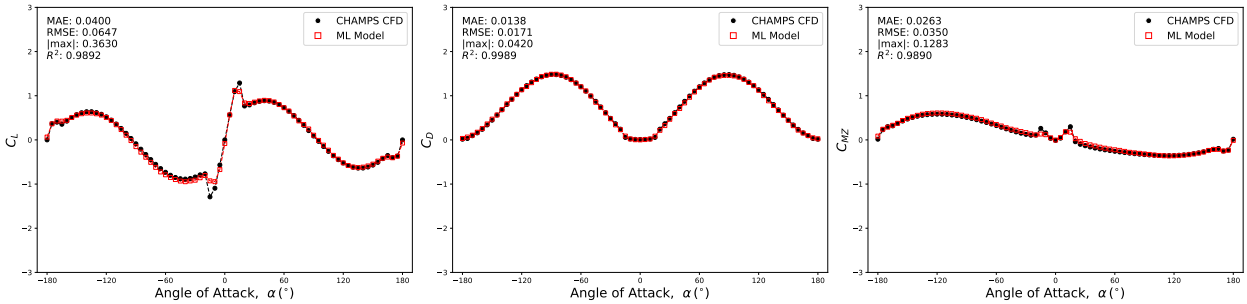
higher errors, as indicated by their greater deviation from the ideal parity line ($y = \hat{y}$). However, significant errors are present across the entire range of C_L and C_M, suggesting that the model's difficulty lies also in generalizing to new airfoil shapes rather than specifically in handling large-magnitude predictions. In contrast, the C_D parity plot closely follows the trend seen in the 10% random split dataset, indicating that the model generalizes well in terms of C_D.

Furthermore, Figure 5.7 presents colormaps of absolute errors across the M–α envelope of the NACA testing dataset. Similar to the 10% random split dataset, significant errors occur in the high Mach number regions for C_L, C_D, and C_M. Additionally, large errors in C_L and C_M are observed within the $\alpha \in [-30, 30]^\circ$ and $\alpha \in [-180, -150] \cup [150, 180]^\circ$ ranges, further reflecting trends seen in the 10% random split dataset. The consistency of these error-prone regions across both testing datasets supports the hypothesis that the model struggles in areas where the training dataset underrepresents key patterns. Moreover, these findings highlight the model architecture's difficulty in making accurate predictions in high Mach and stall onset regions, where nonlinear and often discontinuous phenomena dominate. This limitation may arise because the model's activation functions are differentiable for effective training, restricting the ability to capture patterns in regions of discontinuous flow phenomena.

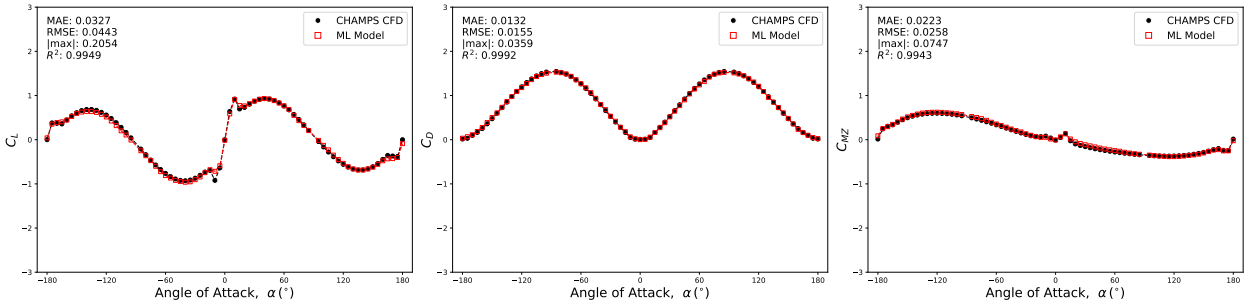
To address the limitations of this model architecture, which relies on the composition of smooth functions to approximate discontinuities, a more targeted approach for training dataset infill is necessary to adequately represent these critical regions. Greater density in these critical areas would reduce the tendency to settle into suboptimal local minima during training, resulting from parameter updates pushed by high training loss due to infrequent sampling instances of discontinuities in the solution space of the domain envelope.

Due to the full-factorial nature of the NACA testing dataset, polar plots spanning large ranges of angles of attack can be generated to illustrate the variation of C_L, C_D, and C_M. Figure 5.8, Figure 5.9, and Figure 5.10 respectively depict these polar plots for the NACA 0012, NACA 2416, and NACA 4608 airfoils, under different freestream flow conditions. These plots also provide a comparative analysis of the CFD predictions and those from the developed ML model. As is evidenced by the absolute error colormaps and reinforced by the polar plots, significant errors in C_L and C_M predominantly occur at stall. Additionally, relatively large errors in C_D are primarily observed in the polar plots corresponding to the highest sampled Mach number ($M = 0.9$) in the NACA testing dataset.

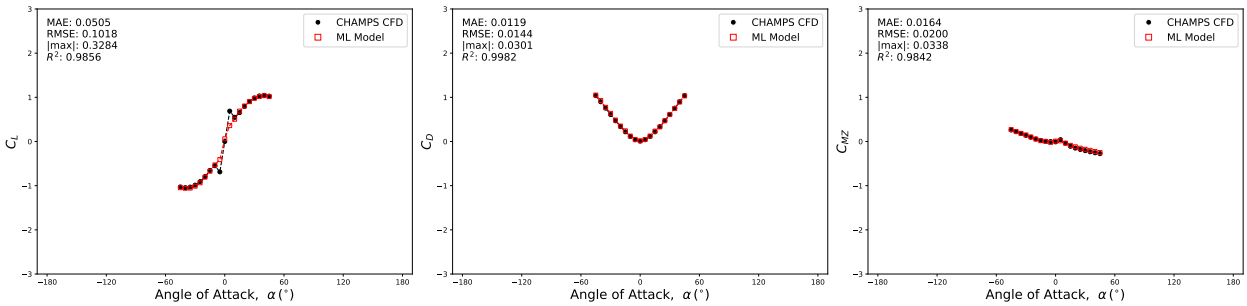
Furthermore, the polar plots reveal an issue in the generated C_M CFD data for the NACA 0012 geometry. Point symmetry around $\alpha = 0^\circ$ is expected but not observed. Since the data points have all converged beyond the coefficient Cauchy window and ρ-residual criteria, the



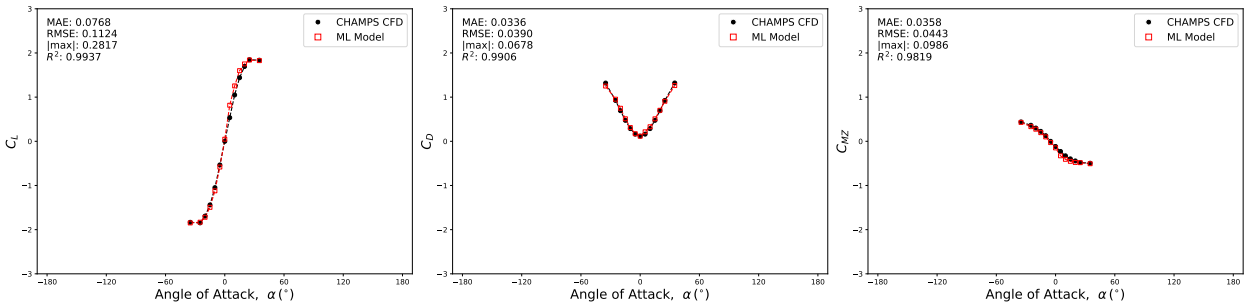
(a) $\alpha \in [-180, 180]°$, $M = 0.3$, and $Re = 6.99 \times 10^6$



(b) $\alpha \in [-180, 180]°$, $M = 0.5$, and $Re = 11.65 \times 10^6$

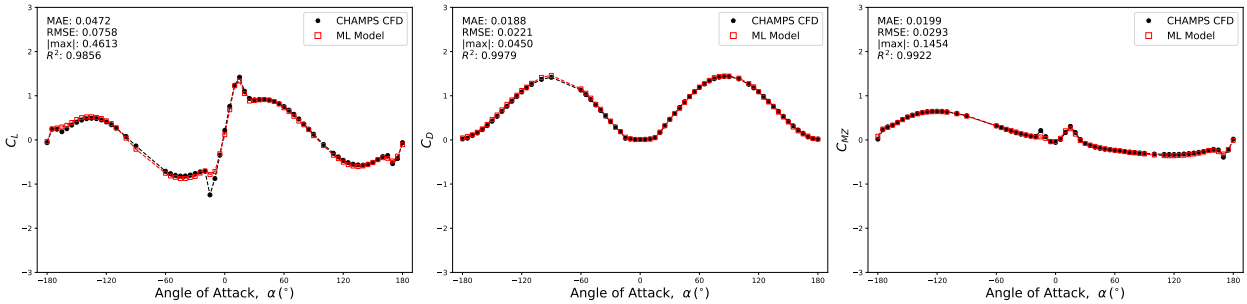


(c) $\alpha \in [-45, 45]°$, $M = 0.7$, and $Re = 16.31 \times 10^6$

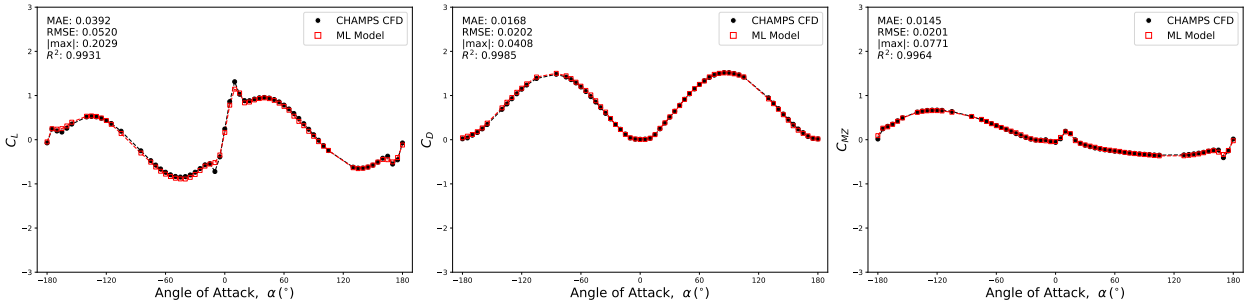


(d) $\alpha \in [-35, 35]°$, $M = 0.9$, and $Re = 20.97 \times 10^6$

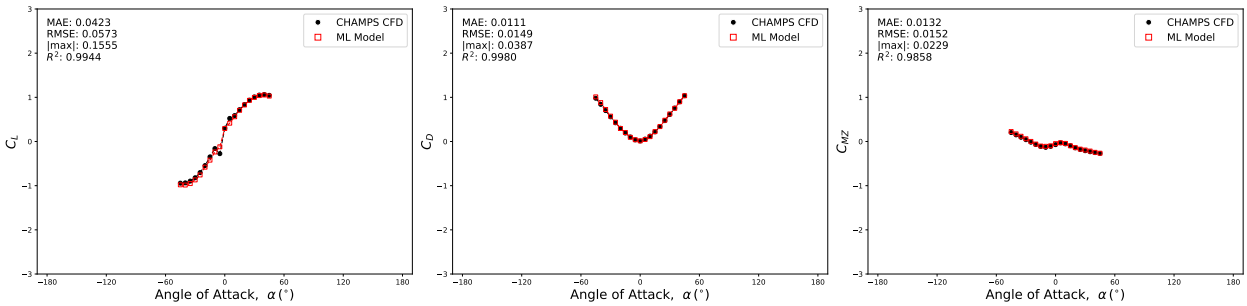
Figure 5.8 Polar plots for NACA 0012 geometry comparing CFD and ML model predictions



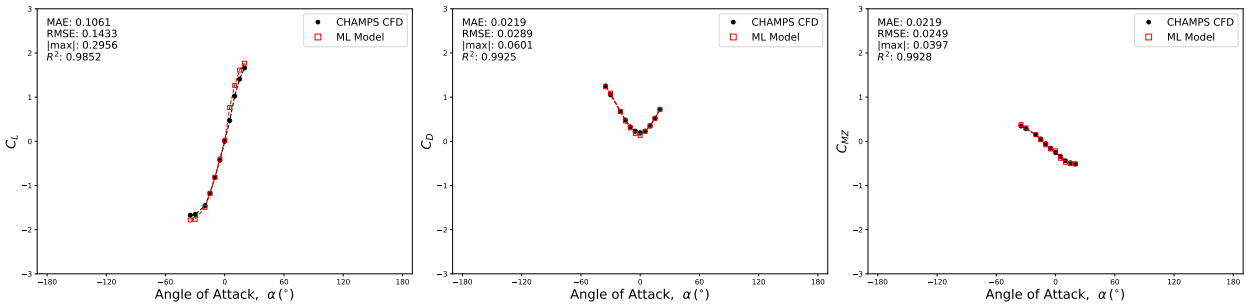
(a) $\alpha \in [-180, 180]°$, $M = 0.3$, and $Re = 6.99 \times 10^6$



(b) $\alpha \in [-180, 180]°$, $M = 0.5$, and $Re = 11.65 \times 10^6$

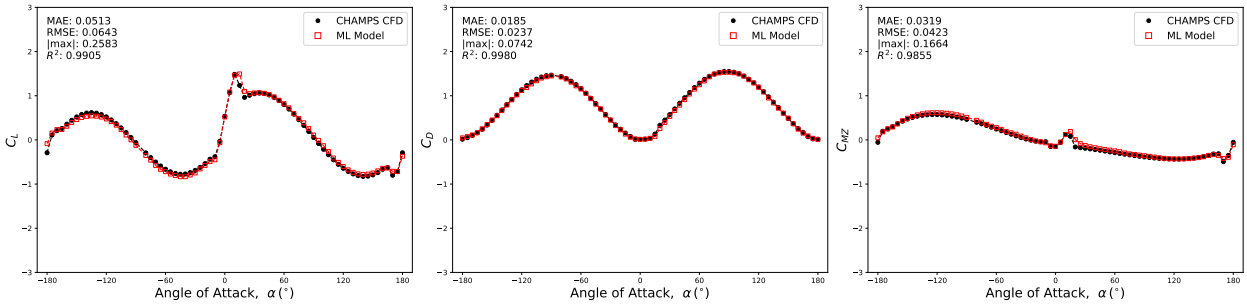


(c) $\alpha \in [-45, 45]°$, $M = 0.7$, and $Re = 16.31 \times 10^6$

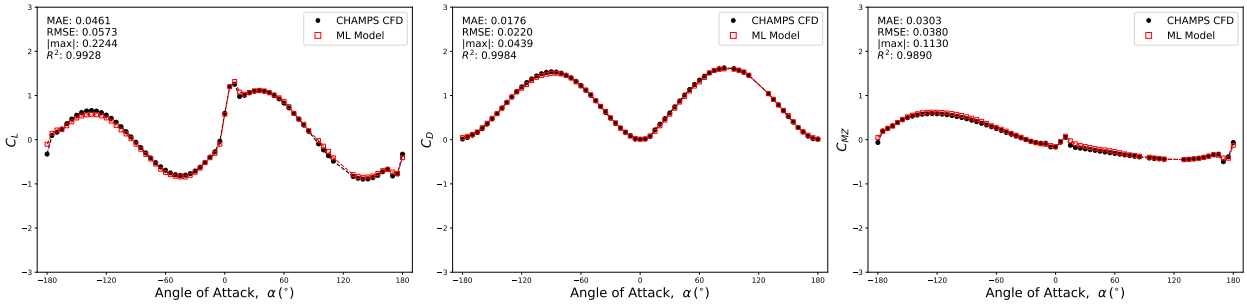


(d) $\alpha \in [-35, 35]°$, $M = 0.9$, and $Re = 20.97 \times 10^6$

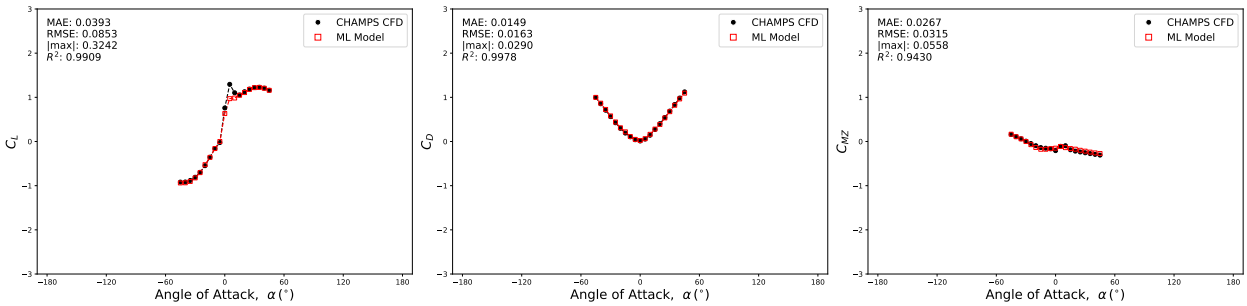
Figure 5.9 Polar plots for NACA 2416 geometry comparing CFD and ML model predictions



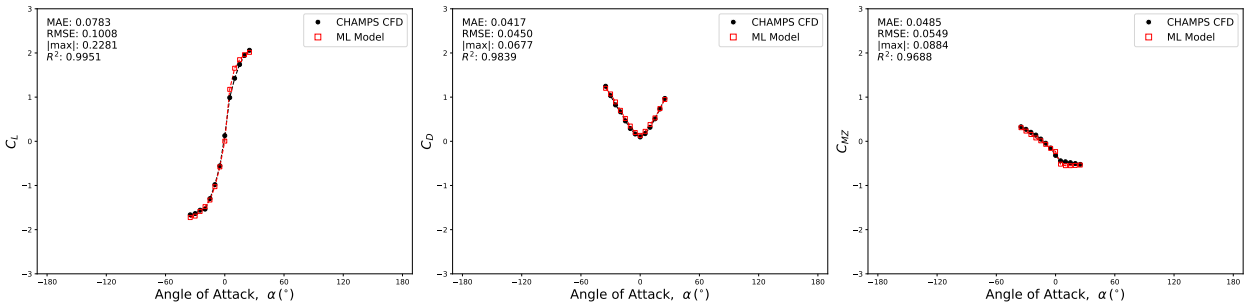
(a) $\alpha \in [-180, 180]°$, $M = 0.3$, and $Re = 6.99 \times 10^6$



(b) $\alpha \in [-180, 180]°$, $M = 0.5$, and $Re = 11.65 \times 10^6$



(c) $\alpha \in [-45, 45]°$, $M = 0.7$, and $Re = 16.31 \times 10^6$



(d) $\alpha \in [-35, 35]°$, $M = 0.9$, and $Re = 20.97 \times 10^6$

Figure 5.10 Polar plots for NACA 4608 geometry comparing CFD and ML model predictions

current hypothesis attributes this asymmetry to misuse of the Cadence Fidelity *Pointwise* meshing software, causing grid generation imprecision. Despite setting grid node precision tolerances to machine precision, the generated meshes exhibit slight asymmetries. Geometry nodes exhibit inaccuracies greater than 10^{-5} chord, as illustrated in Figure 5.11, resulting in larger errors in grid node location propagating through the grid as cells are extruded in the far-field direction. Although this geometric representation error may seem minor, it is significant relative to the minimum wall spacing in the meshes, $\Delta s_{min} = 1.0 \times 10^{-6}$ chord.

While this error in point symmetry for C_M can only be seen for a symmetrical airfoil like the NACA 0012, it indicates that errors can be expected to occur across the entire CFD training dataset in the measure of C_M. Hence, the developed surrogate model will be biased towards this inaccuracy influenced by geometric representation error. Interestingly, comparatively minute differences in point symmetry for C_L and plane symmetry for C_D around $\alpha = 0°$ are found. Differences are smaller than 0.5%, which is expected given the convergence criteria used to generate the dataset.

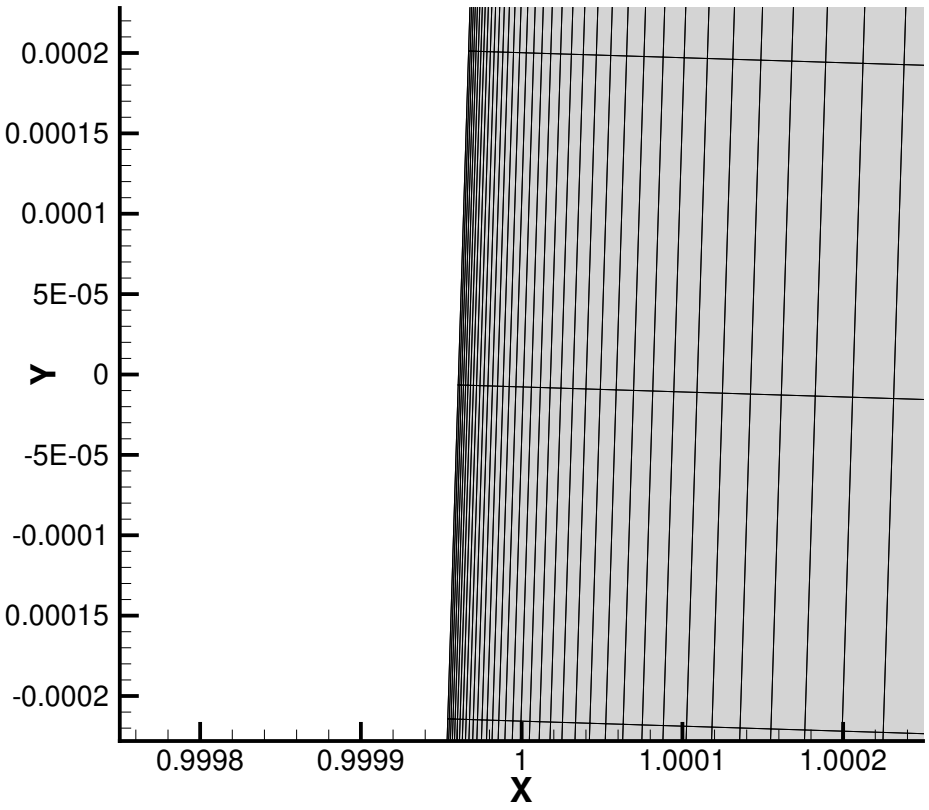


Figure 5.11 View of geometric representation error found in NACA 0012 trailing-edge mesh area. Without this apparent misuse of the Cadence Fidelity *Pointwise* meshing software, the trailing-edge should be perpendicular to the chord line.

5.4 Performance Variability Across Top-20 Models

The inference performance across the models developed from the aforementioned workflow exhibits significant variability. This variability stems from multiple factors. While different model architectures inherently possess distinct characteristics, the training optimization process can be just as influential. In particular, stochastic batch sampling produces models with different parameter sets, even when their architectures are identical. These differences are further amplified by the utilized early stopping procedure, which prevents overfitting. Consequently, multiple models may achieve near identical validation losses yet demonstrate varying testing performance, as distinct local minima were found during training.

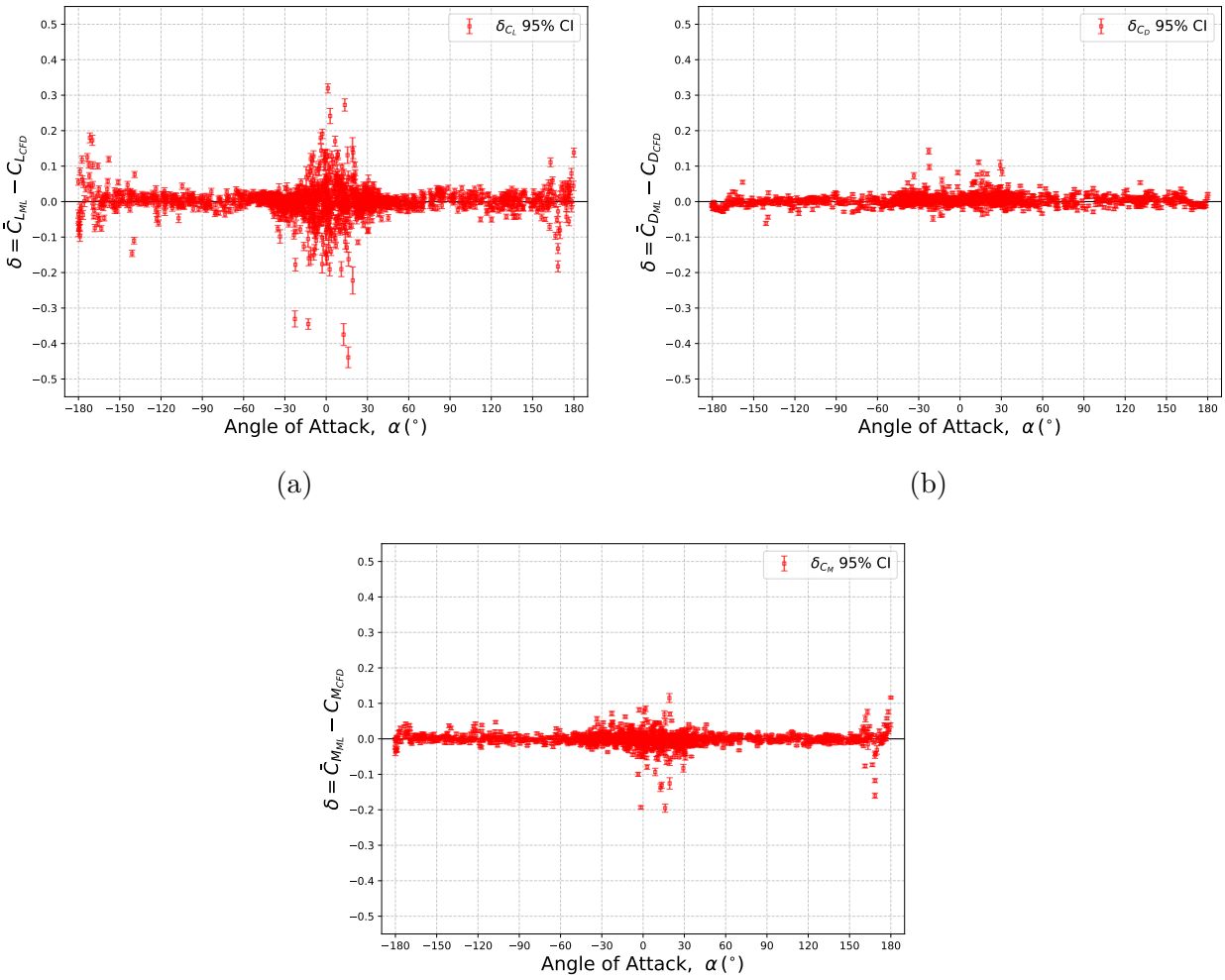


Figure 5.12 Mean prediction error of *Top*-20 models with 95% confidence interval on the measure of error for 10% random split testing dataset for (a) C_L, (b) C_D, and (c) C_M

Figure 5.12 and Figure 5.13 respectively show the mean prediction error and its 95% confidence interval across the angle of attack envelope for both the 10% random split and NACA testing datasets. These values are derived from the predictions of the 20 best models selected from the Pareto front generated through the hyperparameter optimization procedure described in the methodology section. These architectures were the best-performing in terms of validation loss and were each subsequently fine-tuned to achieve even lower validation loss.

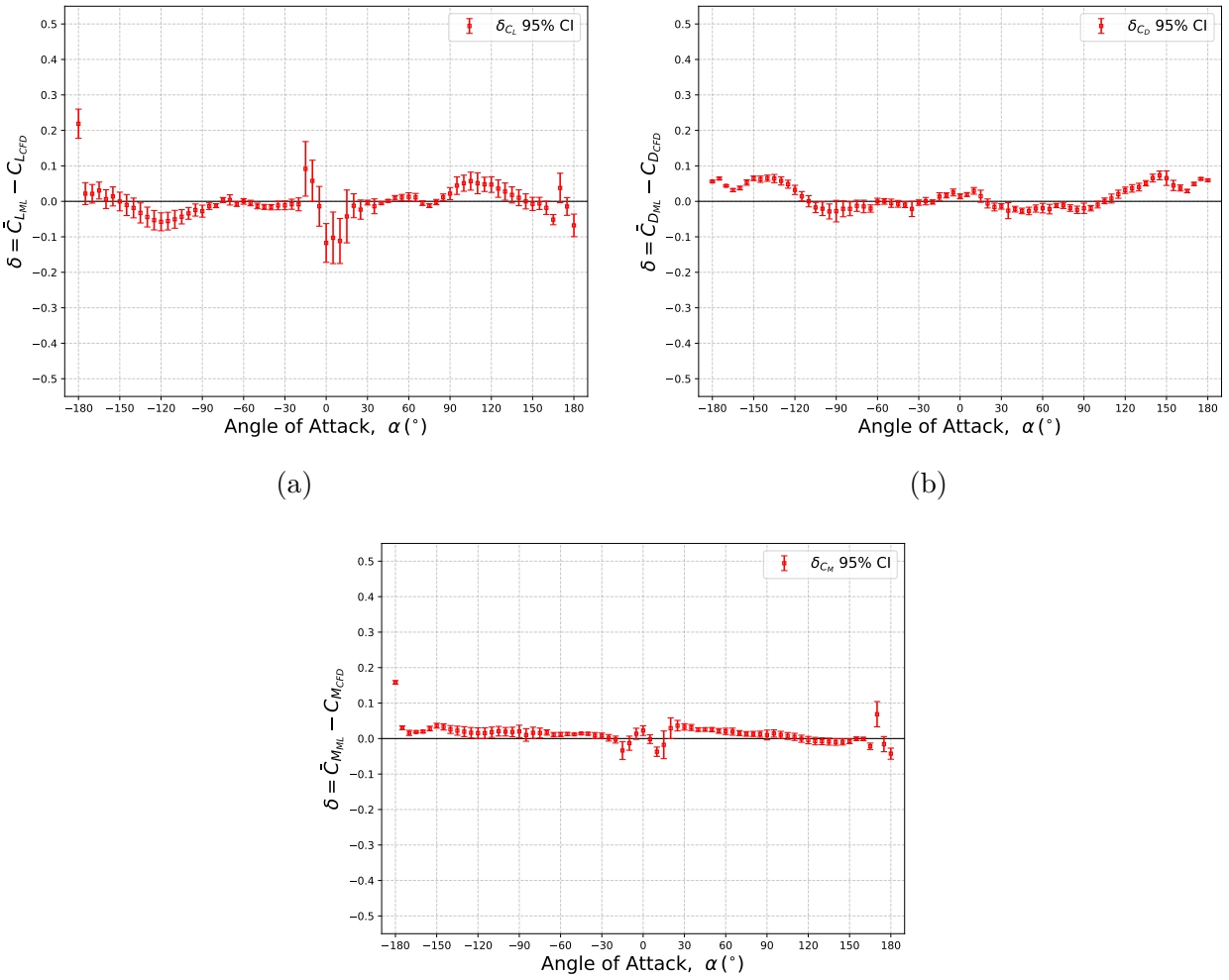


Figure 5.13 Mean prediction error of *Top*-20 models with 95% confidence interval on the measure of error for NACA testing dataset for (a) C_L, (b) C_D, and (c) C_M

These figures illustrate that certain ranges of the angle of attack, specifically $\alpha \in [-30, 30]°$ and $\alpha \in [-180, -150] \cup [150, 180]°$, exhibit significantly larger confidence intervals in error measurement, meaning that predictions in these regions have greater variance compared to

others. However, these large confidence intervals often straddle zero error ($\delta = 0$), implying that training convergence across models may oscillate between local minima associated with capturing the underlying patterns of the nonlinear and often discontinuous phenomena in these regions. These findings further reinforce the need for a targeted approach to training dataset infill to better represent these critical regions.

5.5 Prediction Acceleration

Another essential aspect of this research is the acceleration of predictions compared to conventional CFD methods. To enable seamless coupling with medium-fidelity rotorcraft simulation methods in flight simulators, the developed surrogate model must deliver near real-time predictions, ensuring the simulator's responsiveness is not degraded. While more complex analyses are possible, a direct comparison of computational costs between the best developed ML surrogate model and CFD simulations offers valuable insight, as illustrated in Table 5.5. The CFD database, generated using the workflow detailed in the methodology section, required 16.83 core-years to compute 16,541 simulation data points. In contrast, the *Top*-1 surrogate model described in Table 5.2 requires just 2.3×10^{-8} core-years for predictions made with the database's corresponding 16,541 input vectors, achieving a reduction in computational cost by a factor of 0.54×10^{9}.

Table 5.5 Computational cost for 16,541 CFD and ML surrogate model predictions

	Processor	Nodes	Time	Cost
CHAMPS CFD	2 × AMD EPYC 7532 (2 × 32 cores)	8 —	11.97 (days)	16.83 (core-years)
Top-1 ML Model	Intel i9-10850K (1 core)	1 —	0.98 (s)	3.1×10^{-8} (core-years)
Factor estimate				0.54×10^{9}

For context, the 16.83 core-years used for CFD simulations were computed on CPU nodes with dual AMD EPYC 7532 (2×32 cores) parallelized processors, whereas the 3.1×10^{-8} core-years for the aforementioned surrogate model's predictions were computed on a single CPU core of an Intel i9-10850K processor. Given the differences in hardware and parallelization strategies between the CFD and surrogate model computations, the reported speedup should be considered a coarse estimate. Nonetheless, the surrogate model's inference speed is likely sufficient for flight simulator integration.

CHAPTER 6 CONCLUSION

The overarching objective of this work was to develop an AI-based surrogate model for rapid and accurate prediction of aerodynamic coefficients for VTOL rotorcraft blades across all flight conditions. By generalizing across geometries and flow conditions, the developed model aims to improve the performance of flight simulators that rely on medium-fidelity rotorcraft simulations by eliminating the interpolation error associated with traditional lookup tables. Additionally, the work successfully fulfills its originally stated specific objectives:

- The model development framework exploits the full interpolative capacity of deep learning by training on a sparsely sampled dataset spanning the problem domain. Efficient sparse sampling strategies are employed: angles of attack are drawn from a custom probability distribution, while Latin Hypercube Sampling is used for other relevant variables. Furthermore, developmental efficiency is enhanced by implementing a robust simulation pipeline, maximizing yield across complex cases that contain highly nonlinear and frequently discontinuous flows. To maximize yield, solver calibration is employed in cases of divergence, while Selective Frequency Damping is applied in instances of residual outer limit cycles. Of the 16,541 simulations conducted, over 95% achieved convergence, demonstrating the robustness and efficiency of the strategy in simulating complex, nonlinear, and often discontinuous aerodynamics.

- This work extends prior efforts in the literature by covering a broader range of freestream flow conditions and generalizing across the full state space experienced by rotorcraft blades. When evaluated on an unseen test dataset spanning angles of attack from -180° to 180°, Mach numbers from 0.2 to 0.9, and Reynolds numbers from 2.5×10^6 to 20.0×10^6, the best-performing model demonstrates high predictive accuracy. It achieves an $R^2 = 0.9962$ and MAE $= 0.0302$ for C_L, $R^2 = 0.9987$ and MAE $= 0.0115$ for C_D, and $R^2 = 0.9958$ with MAE $= 0.0128$ for C_M. These results confirm the model's ability to generalize to unseen geometries and flow conditions representative of the operational envelope of VTOL rotorcraft.

- Furthermore, a critical requirement for integration into flight simulators is fast inference to enable near real-time predictions. This work addresses that need by employing meta-optimization techniques and leveraging the Tree-structured Parzen Estimator algorithm to efficiently explore the architectural design space. As a result, the framework produces models that balance predictive accuracy with computational efficiency. The

best-performing model achieves an estimated speedup of approximately 0.54×10^9 in computational resource usage compared to conventional CFD predictions, enabling substantial reductions in simulation cost while preserving the accuracy and fidelity required to capture complex aerodynamic behavior.

6.1 Limitations

Limitations identified throughout this work are listed below:

- Relative to the entire domain of tested cases, larger errors and wider confidence intervals in the measure of error were observed in model predictions near stall. This region, characterized by highly nonlinear and often discontinuous flow behavior, is also classically associated with high uncertainty in CFD simulations. Nevertheless, this indicates that the training dataset in these regions appears too sparse to capture these underlying patterns accurately, leading to suboptimal training. Furthermore, these confidence intervals frequently straddle zero error, suggesting that the training process oscillates between suboptimal local minima when attempting to learn these complex patterns.

- The chosen architecture, which employs differentiable activation functions to enable effective training, is inherently ill-suited for representing discontinuous functions. Since it relies on the composition of smooth activation functions, it can only approximate discontinuities, often inaccurately, particularly when the training dataset lacks sufficient representation of those discontinuities necessary in forming these approximations.

- The developed surrogate model exhibited a small bias toward NURBS airfoils. This limitation was observable during evaluation on the NACA testing dataset, where the model's inference performance declined ever so slightly. Despite the diversity within the training dataset, all geometries were derived from NURBS representations. However, this gap in performance may be attributable to the geometry parameterization approach, where cubic spline smoothing followed by PCA transformation, can be sensitive to variations in the Cartesian coordinate distribution of the input geometries.

- This study did not account for rotorcraft mechanisms that induce oscillatory rotor motion. Various designs produce coupled pitching, heaving, and lead-lag rotor motion dynamics. The unsteady motion arising from these complex mechanisms, along with the associated aerodynamic hysteresis observed once the surrounding flow reaches its outer limit cycle was excluded from this work. These effects were omitted due to their high computational cost and frequent issues with numerical divergence.

6.2 Future Research

Subsequent investigations should work to overcome the limitations identified and build upon the avenues of inquiry this work has initiated:

- Geometric parameterization techniques should be compared in the context of surrogate modeling for aerodynamic coefficients. The studied parameterization techniques should be agnostic to different airfoil geometry generation techniques and sources. Furthermore, they should aim to reduce the high dimensionality commonly associated with Cartesian coordinate or pixel-based representations while preserving the essential information required for accurate surrogate model predictions.

- Future studies should investigate more targeted approaches to dataset infill. As shown in Appendix C, the globally uniform sampling approach used in this study results in plateauing model error plateaus despite increasing data density. This study's results indicate that relatively large errors predominantly occur in regions where complex patterns remain underrepresented. It is possible that an adaptive infill strategy, such as one guided by active learning and focused on areas of high model inference error, would be necessary to circumvent this deficiency.

- Future work should explore the potential of neural networks to model aerodynamic coefficients associated with unsteady blade-section motions such as pitching, heaving, or lead-lag dynamics. Appendices D–F presents preliminary results for surrogate modeling for the aerodynamic coefficient hysteresis loops of pitching airfoils. This preliminary research spans different airfoil pitching dynamics with varying frequencies and pitching amplitudes but is limited to symmetrical airfoils and fixed freestream flow conditions. Future studies should expand this unsteady model to more complex geometries, flow conditions, and unsteady rotor dynamics.

- Future research into medium-fidelity rotorcraft simulation methods that incorporate non-linear coupling should include comparative analyses of interpolation errors inherent to lookup tables versus the low to moderate prediction errors associated with surrogate models. Additionally, studies should systematically increase lookup table density to identify the inflection point at which interpolation error becomes comparable to or lower than surrogate model prediction error. As table density increases, interpolation error is expected to decrease, allowing for a clearer understanding of the trade-off between these two sources of approximation error.

REFERENCES

[1] S. Fiset and D. Vidal, *Unpublished Simulation Results*. École Polytechnique de Montréal, 2024.

[2] C. Ferlisi, "Rotor wake modelling using the vortex-lattice method," Master's thesis, École Polytechnique de Montréal, April 2018.

[3] M. Parenteau, K. Sermeus, and E. Laurendeau, "VLM Coupled with 2.5D RANS Sectional Data for High-Lift Design," in *2018 AIAA Aerospace Sciences Meeting*. Kissimmee, Florida: American Institute of Aeronautics and Astronautics, Jan. 2018.

[4] V. Proulx-Cabana *et al.*, "A Hybrid Non-Linear Unsteady Vortex Lattice-Vortex Particle Method for Rotor Blades Aerodynamic Simulations," *Fluids*, vol. 7, no. 2, p. 81, Feb. 2022.

[5] A. Cocco *et al.*, "A non-linear unsteady vortex-lattice method for rotorcraft applications," *The Aeronautical Journal*, vol. 128, no. 1328, pp. 2308–2330, Oct. 2024.

[6] V. Proulx-Cabana, G. Michon, and E. Laurendeau, "Parametrization Effects of the Non-Linear Unsteady Vortex Method with Vortex Particle Method for Small Rotor Aerodynamics," *Fluids*, vol. 9, no. 1, p. 24, Jan. 2024.

[7] R. M. Greenman and K. R. Roth, "High-Lift Optimization Design Using Neural Networks on a Multi-Element Airfoil," *Journal of Fluids Engineering*, vol. 121, no. 2, pp. 434–440, Jun. 1999.

[8] H. P. Gavin, "The levenberg-marquardt method for nonlinear least squares curve-fitting problems," 2013.

[9] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, May 2015.

[10] N. Papila *et al.*, "Assessment of neural net and polynomial-based techniques for aerodynamic applications," in *17th Applied Aerodynamics Conference*. Norfolk, VA, U.S.A.: American Institute of Aeronautics and Astronautics, Jun. 1999.

[11] T. Rajkumar and J. E. Bardina, "Training data requirement for a neural network to predict aerodynamic coefficients," in *SPIE Defense + Commercial Sensing*, 2003.

[12] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, ser. Proceedings of Machine Learning Research, G. Gordon, D. Dunson, and M. Dudík, Eds., vol. 15. Fort Lauderdale, FL, USA: PMLR, 11–13 Apr 2011, pp. 315–323.

[13] M. Khurana, H. Winarto, and A. Sinha, "Application of swarm approach and artificial neural networks for airfoil shape optimization," in *12th AIAA/ISSMO multidisciplinary analysis and optimization conference*, 2008, p. 5954.

[14] H. Sobieczky, "Parametric airfoils and wings," 1999.

[15] D. A. Masters *et al.*, "Geometric comparison of aerofoil shape parameterization methods," *AIAA Journal*, vol. 55, no. 5, pp. 1575–1589, 2017.

[16] D. P. Kingma, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[17] G. E. Moore, "Cramming more components onto integrated circuits," *Electronics*, vol. 38, no. 8, pp. 114–117, 1965.

[18] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016.

[19] L. He *et al.*, "Multi-Fidelity Aerodynamic Data Fusion with a Deep Neural Network Modeling Method," *Entropy*, vol. 22, no. 9, p. 1022, Sep. 2020.

[20] D. G. Krige, "A statistical approach to some basic mine valuation problems on the witwatersrand," *Journal of the Southern African Institute of Mining and Metallurgy*, vol. 52, no. 6, pp. 119–139, 1951.

[21] H. Chen *et al.*, "Multiple Aerodynamic Coefficient Prediction of Airfoils Using a Convolutional Neural Network," *Symmetry*, vol. 12, no. 4, p. 544, Apr. 2020.

[22] K. Balla *et al.*, "An application of neural networks to the prediction of aerodynamic coefficients of aerofoils and wings," *Applied Mathematical Modelling*, vol. 96, pp. 456–479, Aug. 2021.

[23] 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*, vol. 96, pp. 23–61, Jan. 2018.

[24] X. Du, P. He, and J. R. Martins, "Rapid airfoil design optimization via neural networks-based parameterization and surrogate modeling," *Aerospace Science and Technology*, vol. 113, p. 106701, Jun. 2021.

[25] R. B. Nelsen, *An introduction to copulas*. Springer, 2006.

[26] A. Sridharan and J. Sinsay, "Accelerating aerodynamic design of rotors using a multi-fidelity approach in torc: Tool for optimization of rotorcraft concepts," 06 2023.

[27] J. Cornelius and S. Schmitz, "Dragonfly rotor optimization using machine learning applied to an overflow generated airfoil database," 05 2024, pp. 1–20.

[28] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *The Bulletin of Mathematical Biophysics*, vol. 5, no. 4, pp. 115–133, Dec. 1943.

[29] F. Rosenblatt, "The perceptron: a probabilistic model for information storage and organization in the brain." *Psychological review*, vol. 65 6, pp. 386–408, 1958.

[30] A. Vaswani *et al.*, "Attention is all you need," 2023.

[31] A. Brocklehurst and G. Barakos, "A review of helicopter rotor blade tip shapes," *Progress in Aerospace Sciences*, vol. 56, pp. 35–74, Jan. 2013.

[32] W. Johnson, H. Yeo, and C. Acree, "Performance of advanced heavy-lift, high-speed rotorcraft configurations," p. 29, 10 2007.

[33] S. Pagluica, "Winds of superhurricane force, and a heated anemometer for their measurement during ice-forming conditions," *Monthly Weather Review*, vol. 62, no. 6, pp. 186–189, Jun. 1934.

[34] M. S. Selig, "UIUC airfoil coordinates database," 1996-2024.

[35] S. Painchaud-Ouellet *et al.*, "Airfoil Shape Optimization Using NURBS Representation Under Thickness Constraint," in *42nd AIAA Aerospace Sciences Meeting and Exhibit*. Reno, Nevada: American Institute of Aeronautics and Astronautics, Jan. 2004.

[36] R. J. B. M. D. McKay and W. J. Conover, "Comparison of three methods for selecting values of input variables in the analysis of output from a computer code," *Technometrics*, vol. 21, no. 2, pp. 239–245, 1979.

[37] I. Sobol', "On the distribution of points in a cube and the approximate evaluation of integrals," *USSR Computational Mathematics and Mathematical Physics*, vol. 7, no. 4, pp. 86–112, 1967.

[38] F. M. White, *Fluid mechanics*, 5th ed., ser. McGraw-Hill series in mechanical engineering. New York London: McGraw-Hill, 2003.

[39] M. Parenteau *et al.*, "Development of parallel cfd applications on distributed memory with chapel," in *2020 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*, 2020, pp. 651–658.

[40] P. Roe, "Approximate riemann solvers, parameter vectors, and difference schemes," *Journal of Computational Physics*, vol. 43, no. 2, pp. 357–372, 1981.

[41] A. Harten, P. Lax, and B. van Leer, "On upstream differencing and godunov-type schemes for hyperbolic conservation laws," *SIAM Rev*, vol. 25, pp. 35–61, 01 1983.

[42] A. Harten and J. Hyman, "Self-adjusting grid methods for one-dimensional hyperbolic conservation laws," *Journal of Computational Physics*, vol. 50, pp. 235–269, 06 1983.

[43] B. van Leer, "Towards the ultimate conservative difference scheme. v. a second-order sequel to godunov's method," *Journal of Computational Physics*, vol. 32, no. 1, pp. 101–136, 1979.

[44] V. Venkatakrishnan, *On the accuracy of limiters and convergence to steady state solutions.*

[45] V. Venkatakrishnan, "Convergence to steady state solutions of the euler equations on unstructured grids with limiters," *Journal of Computational Physics*, vol. 118, no. 1, pp. 120–130, 1995.

[46] P. Spalart and S. Allmaras, "A one-equation turbulence model for aerodynamic flows," *AIAA*, vol. 439, 01 1992.

[47] M. Vinokur, "On one-dimensional stretching functions for finite-difference calculations," *Journal of Computational Physics*, vol. 50, no. 2, pp. 215–234, 1983.

[48] C. M. Bishop, *Pattern recognition and machine learning*, ser. Information science and statistics. New York: Springer, 2006.

[49] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," 2015.

[50] D.-A. Clevert, T. Unterthiner, and S. Hochreiter, "Fast and accurate deep network learning by exponential linear units (elus)," 2016.

[51] N. Ketkar, *Introduction to PyTorch.* Berkeley, CA: Apress, 2017, pp. 195–208.

[52] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, Oct. 1986.

[53] R. M. Greenman, "Two-dimensional high-lift aerodynamic optimization using neural networks," PhD Thesis, Stanford University, Stanford, CA, USA, 1998.

[54] K. He *et al.*, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," 2015.

[55] M. Pagliardini, P. Ablin, and D. Grangier, "The ademamix optimizer: Better, faster, older," 2024.

[56] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," 2019.

[57] N. Srivastava *et al.*, "Dropout: A simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol. 15, no. 56, pp. 1929–1958, 2014.

[58] X. Bouthillier *et al.*, "Epistimio/orion: Asynchronous distributed hyperparameter optimization," 2022.

[59] J. Bergstra *et al.*, "Algorithms for hyper-parameter optimization," in *Advances in Neural Information Processing Systems*, J. Shawe-Taylor *et al.*, Eds., vol. 24. Curran Associates, Inc., 2011.

[60] J. Bergstra, D. Yamins, and D. Cox, "Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures," in *Proceedings of the 30th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, S. Dasgupta and D. McAllester, Eds., vol. 28, no. 1. Atlanta, Georgia, USA: PMLR, 17–19 Jun 2013, pp. 115–123.

[61] E. Åkervik *et al.*, "Steady solutions of the navier-stokes equations by selective frequency damping," *Physics of Fluids*, vol. 18, no. 6, p. 068102, 06 2006.

[62] J. W. Cooley and J. W. Tukey, "An algorithm for the machine calculation of complex fourier series," *Mathematics of computation*, vol. 19, no. 90, pp. 297–301, 1965.

APPENDIX A SIMULATION BIFURCATION STRATEGY

Solutions that meet the convergence criteria C_1 or C_6 are considered converged, as their aerodynamic coefficients reach acceptable engineering accuracy beyond the 4^{th} decimal place and typically beyond the 5^{th}. However, solutions often fail to satisfy these criteria. To prevent excessive DoE infill and maintain database completeness, a bifurcation strategy is applied when solutions measure at the conditions C_2, C_3, C_4, or C_5.

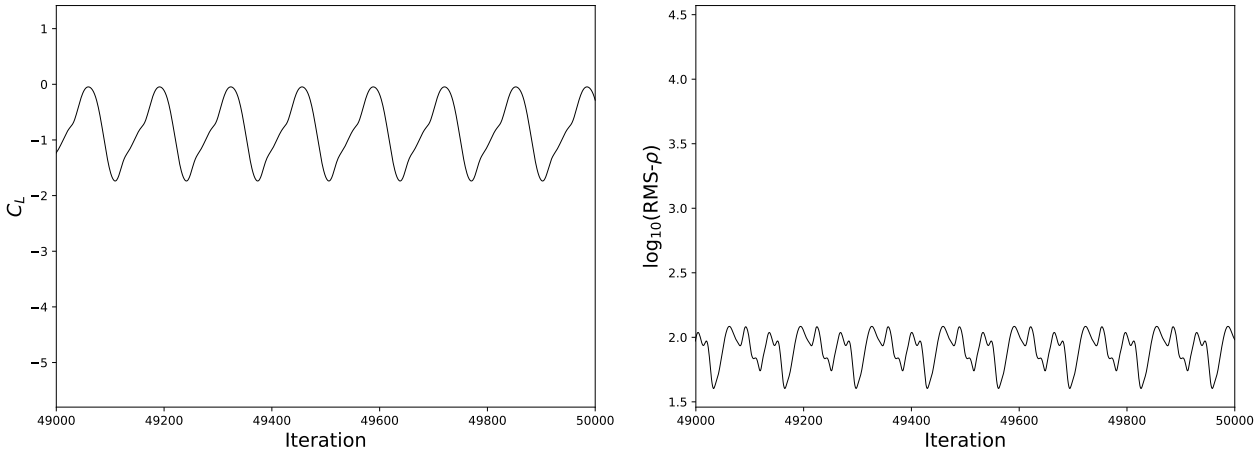
Table A.1 Simulation convergence conditions

	Condition	Outcome
C_1	Cauchy window criterion is met	Converged
C_2	NaN is found	Bifurcation
C_3	ρ-residual $> 10^0$	Bifurcation
C_4	$10^0 > \rho$-residual $> 10^{-1}$	Pruned
C_5	$10^{-1} > \rho$-residual $> 10^{-2}$	Pruned
C_6	$10^{-2} > \rho$-residual	Converged

In the simplest case, when the conditions C_4 or C_5 are measured, the corresponding simulation cases are pruned from the database. This decision was based on the results of several thousand simulations, where less than 1% of cases produced solutions with ρ-residuals between 10^0 and 10^{-2} at the final iteration. Individual analysis of these simulation cases did not reveal any insights into potential solutions that could aid in improving their convergence. Furthermore, the convergence histories of most of these simulations indicated no clear trend toward eventual convergence with the addition of further iterations.

Simulation cases are classified under the C_2 condition when NaN values appear in their solutions. This condition was found in critical portions of the database envelope where highly non-linear flow phenomena are typically found. Cases exhibiting the C_2 condition were predominantly clustered around high M and high Re values, combined with sufficiently large α values to induce deep stall phenomena. Due to the relatively large frequency of the C_2 condition's appearance in the simulation envelope, and the significance of the associated simulations, a strategy was adopted to promote convergence in these cases. Solver parameters were adjusted to improve stability where the CFL number was reduced by a factor of 10, from CFL = 50 to CFL = 5, and the solution update relaxation factor was lowered from $\omega = 0.7$ to $\omega = 0.4$.

The C_3 condition is assigned to simulation cases where the ρ-residual remains greater than 10^0 after the final iteration. An analysis of the convergence histories from several hundred of these cases revealed that nearly all exhibit outer limit cycles (OLCs) in their solutions. OLCs occur in steady simulations when unsteady flow phenomena lead to self-sustaining, oscillatory behavior, preventing convergence. Within this simulation envelope, OLCs are typically observed at α values large enough to induce deep stall, accompanied by flow instabilities such as periodic vortex shedding. In these cases, the periodic fluctuations in both residuals and aerodynamic coefficient histories manifest in non-convergence. An example of the convergence history of a solution exhibiting an OLC is shown in Figure A.1.



(a) (b)

Figure A.1 Outer limit cycle for (a) C_L and (b) ρ-residual convergence histories

Furthermore, cases exhibiting the C_3 condition account for a significant portion of the simulated dataset and correspond to concentrated regions within the simulation envelope, typically associated with sufficiently high α values that trigger deep stall phenomena. To ensure database completeness, an involved strategy is required to deal with bifurcations in cases with the C_3 condition. Selective Frequency Damping (SFD) [61] was employed to address the challenges presented by the C_3 condition, as it has proven effective in stabilizing solutions affected by undesired oscillations. SFD operates under the assumption that a steady-state solution exists for time-dependent, non-linear system of equations. In this instance, RANS equations take on the form of this non-linear system of equations. For an unstable solution like those with the C_3 condition, any perturbation can lead to rapid divergence from steady convergence. SFD introduces a regularization approach based on proportional feedback control, where χ serves as the control coefficient. Since the exact steady-state solution is unknown *a priori*, the error feedback cannot be directly measured against it. Instead, it

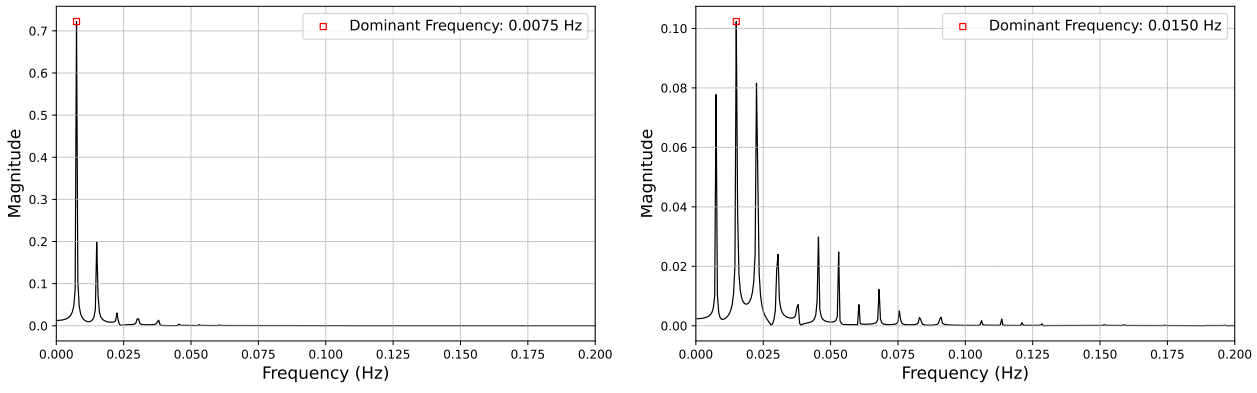
is evaluated relative to an approximate steady solution, achieved by applying a temporal low-pass filter with width Δ.

The key parameters, χ and Δ, used in SFD are determined based on the period T (in iterations) of OLC oscillations observed in the convergence history of the simulations. The filter width is computed according to Equation A.1, where it is set to twice the period. This choice follows the Nyquist-Shannon sampling theorem, which states that the sampling rate must be at least twice the signal's bandwidth to avoid aliasing. The parameter χ is calculated using a commonly adopted proportion based on the rule of thumb outlined in Equation A.2.

$$\Delta = 2 \cdot T \tag{A.1}$$

$$\chi = \frac{1}{\Delta} \tag{A.2}$$

The periods of OLC oscillations were determined using the Fast Fourier Transform (FFT) [62] applied to the convergence histories of C_3 cases. As shown in Figure A.1, the OLC of a typical convergence history consists of multiple fluctuating components. The approach adopted in this bifurcation strategy was to measure the convergence history signal's dominant frequency so that the OLC's period could be calculated as $T = 1 / f$. The choice between using an aerodynamic coefficient or a residual convergence history as the signal presents itself. Figure A.2 presents typical frequency spectra for both the C_L and ρ-residual convergence history signals. Heuristic analysis of initial simulations revealed that the dominant frequencies in C_L convergence histories are generally lower than those in their ρ-residual counterparts. To adopt a conservative approach, the dominant frequencies from the C_L convergence histories were used in this bifurcation strategy, resulting in larger filter widths.



(a) (b)

Figure A.2 Frequency spectrum for (a) C_L and (b) ρ-residual outer limit cycles

APPENDIX B 4–DIGIT NACA TESTING DATASET

To evaluate the model's generalization ability, an additional dataset is generated using a technique different from the one employed in the generation of the AI-surrogate model training dataset. Unlike the training dataset, which consists of geometries generated with NURBS, this testing dataset includes 4-digit NACA airfoils: NACA 0012, NACA 2416, and NACA 4608. These three airfoil geometries are superimposed in Figure B.1.

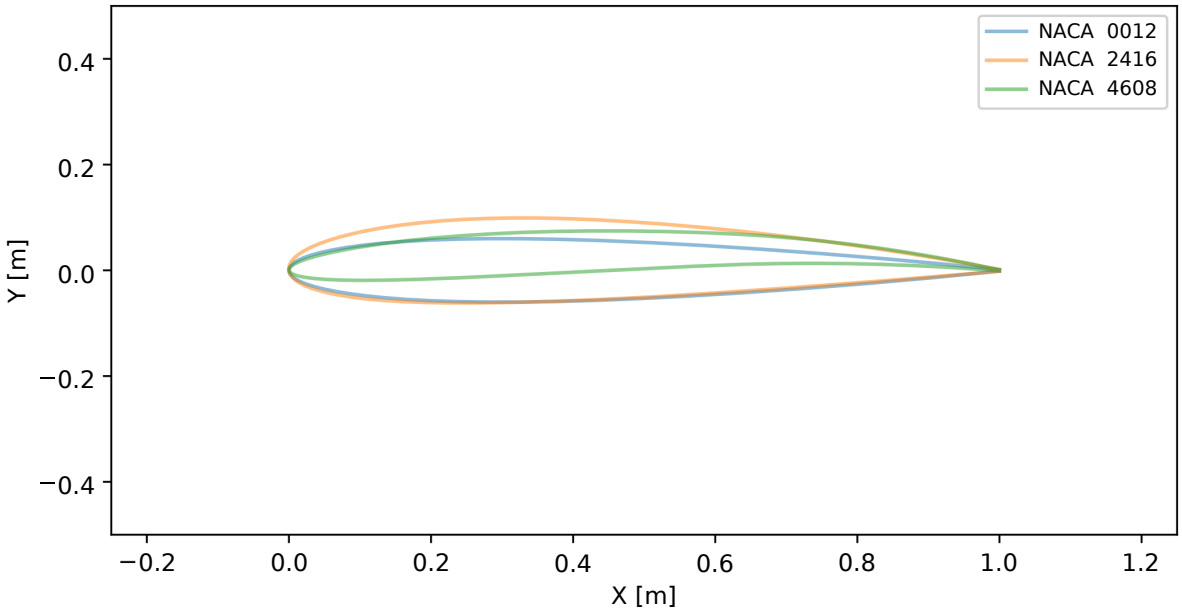


Figure B.1 4-digit NACA airfoils comprising the testing dataset's geometries

Furthermore, complex flow conditions covering the entire applicability domain of the model are simulated in the NACA testing dataset. The domain is sampled using a full-factorial approach, where each of the aforementioned NACA airfoils undergoes CFD simulation at every full-factorial flow condition sample, as detailed in Table B.1 and Table B.2.

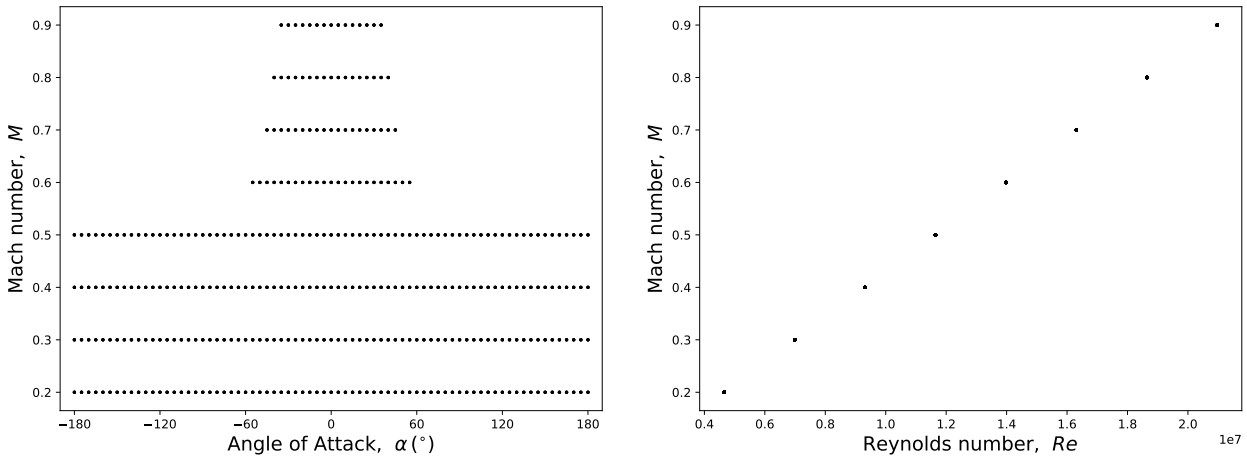
Table B.1 NACA testing dataset envelope range for independent flow condition variables

Variable	Range
Angle of attack	$\alpha \in$ {-180, -175, -170, ... , 180} °
Mach number	$M \in$ {0.2, 0.3, 0.4, ... , 0.9}, as a function of α
Altitude	$h = 0$ m

Table B.2 NACA testing dataset envelope range for the dependent flow condition variable

Variable	Range
Reynolds number	$Re \in [4.7, 21.0] \times 10^6$, as a function of M & h

The envelope of these sampled full-factorial is illustrated in Figure B.2, where the entire range of Mach number and angle of attack values present in the training dataset are represented. Furthermore, sampling at a consistent altitude of $h = 0$ m results in consequently larger Reynolds numbers, given the barometric formula used in the problem domain's definition.



(a) (b)

Figure B.2 Sampled (a) $M - \alpha$ and (b) $M - Re$ full-factorial envelopes for 4-digit NACA testing dataset

Similar to the training dataset, a synopsis of the database generation and bifurcation strategy is conducted, where incorporating SFD and calibrating solver parameters when necessary results in an almost completely converged database.

Table B.3 4-digit NACA testing database generation synopsis

Subset \ Criteria	C_1	C_2	C_3	C_4	C_5	C_6	CONV%
(a)	781	122	166	3	1	25	73.41
(b)	135	8	19	0	0	4	83.73
(c)	77	8	36	0	1	0	63.11
(d)	5	0	30	0	1	0	13.89
(e)	998	16	49	3	3	29	93.53

APPENDIX C TRAINING DATASET DENSITY

The effectiveness of the sampling strategy employed in constructing this study's datasets is evaluated. This strategy relies on a Latin Hypercube Sampling (LHS) DoE, which generates uniformly spaced stratified sampling. Predictive performance for C_L, C_D, and C_M is assessed across models trained on datasets of varying levels of sampling completeness, as illustrated in Figure C.1. Using the 20 top-performing model architectures, selected based on validation loss when trained on the full training dataset (excluding the portions reserved for validation and testing), these models are subsequently tested on the 10% randomly sampled dataset.

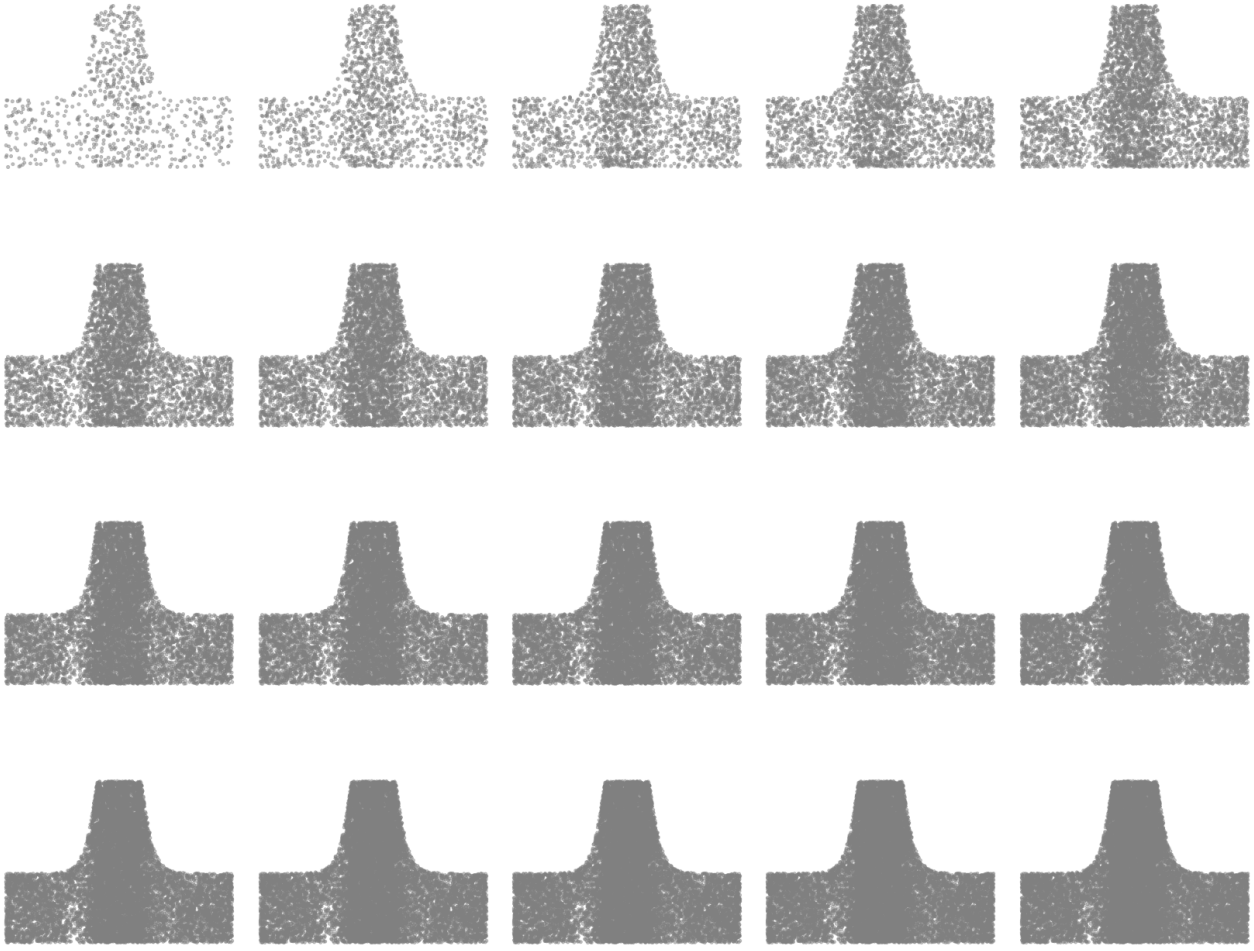


Figure C.1 Training dataset density at increasing levels of sampling dataset (in intervals of 5% from top left (5% of 15,807 samples) to bottom right (100% of 15,807 samples)

Figure C.2 demonstrates that increasing LHS sampling density beyond a certain threshold yields diminishing returns, as model performance improves only marginally with higher density. This analysis is not entirely rigorous, as hyperparameter selection was based on the optimization processes conducted on the complete dataset, whereas optimal model architectures and training dynamics inherently evolve alongside changes in training database density. However, properly applying such optimizations would primarily enhance models trained on low-density databases in this figure, further reinforcing the trend of diminishing returns. Nonetheless, a higher training database density does lead to better-performing models. Future work should explore alternative or more targeted sampling strategies, as well as the impact of uncertainty in data points within the generated CFD database.

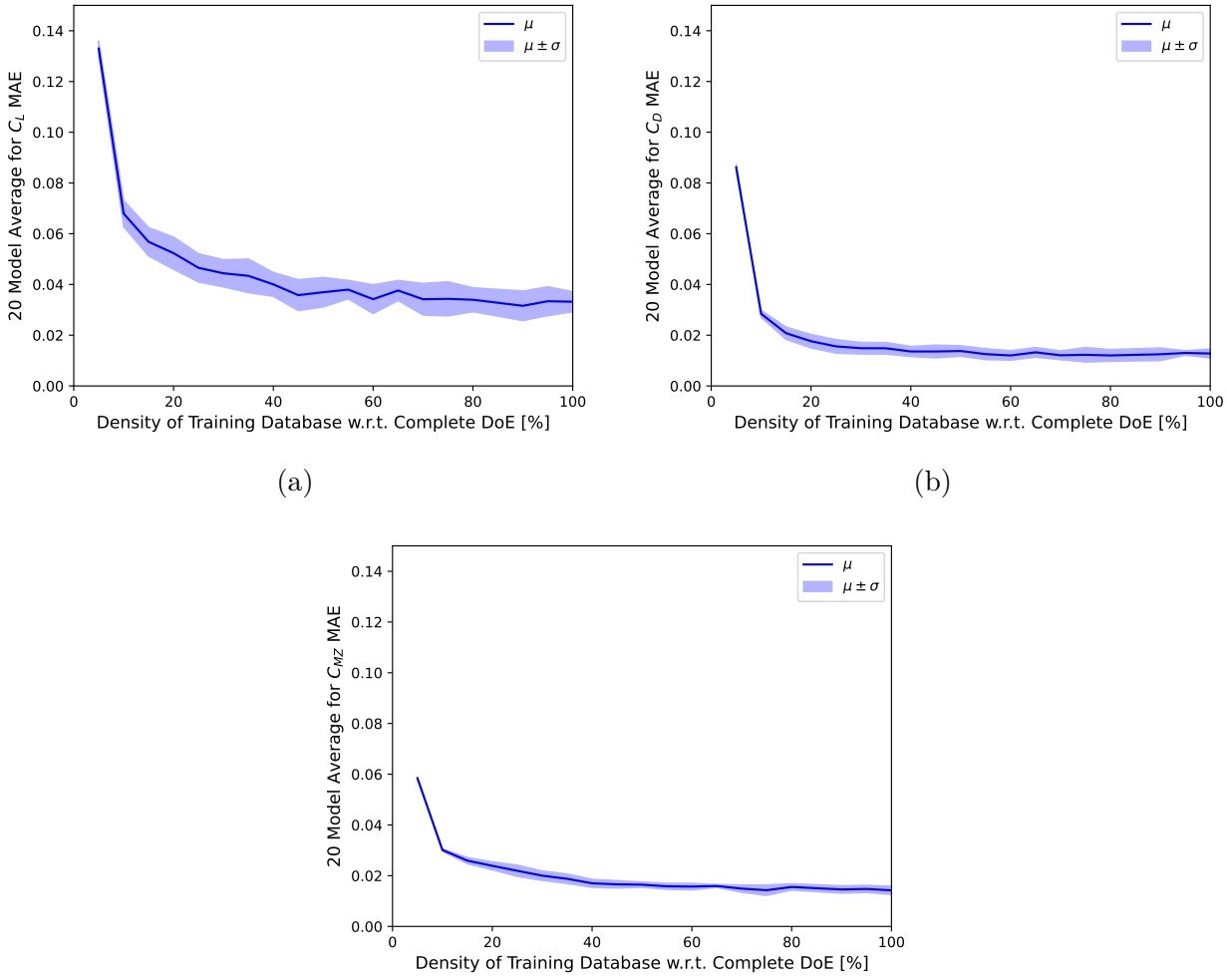


Figure C.2 Mean absolute error for (a) C_L, (b) C_D, and (c) C_M on 10% random split testing dataset for models trained on datasets of increasing sampling densities

APPENDIX D UNSTEADY PITCHING DATASET

The unsteady scenario with hysteresis also necessitates the creation of a CFD database. Although this database is limited in both dimensionality and parameter ranges, its primary purpose is to serve as a demonstration model, showcasing the capability of machine learning surrogate models in predicting vectorial outputs for the hysteresis curves of C_L, C_D, and C_M in the context of pitching rotorcraft blades. The generated dataset adheres to the envelope ranges for pitching dynamic variables, as described in Table D.1, following the equation $\alpha(t) = \alpha_0 + A \cdot sin(\omega t)$. The dataset's geometries conform to the envelope range specified in Table D.3, while the freestream flow conditions remain fixed, as outlined in Table D.2. The dataset achieved convergence in 83.16% of simulated cases, as detailed in Table D.4.

Table D.1 Envelope for pitching dynamic variables

Variable	Range
Initial angle of attack	$\alpha_0 \in$ [-15, 15] °
Pitching amplitude	$A \in$ [2, 10] °
Pitching frequency	$\omega \in [2\pi, 6\pi]$ rad/s

Table D.2 Freestream flow conditions values for unsteady dataset

Variable	Range
Altitude	$h = 0$ m
Mach number	$M = 0.2$
Reynolds number	$Re = 4.66 \times 10^6$, as a function of M & h

Table D.3 Unsteady dataset envelope range for symmetric 4-digit NACA geometry

Variable	Range
Maximum airfoil thickness	$t \in$ [6, 18] %

Table D.4 Unsteady database generation synopsis

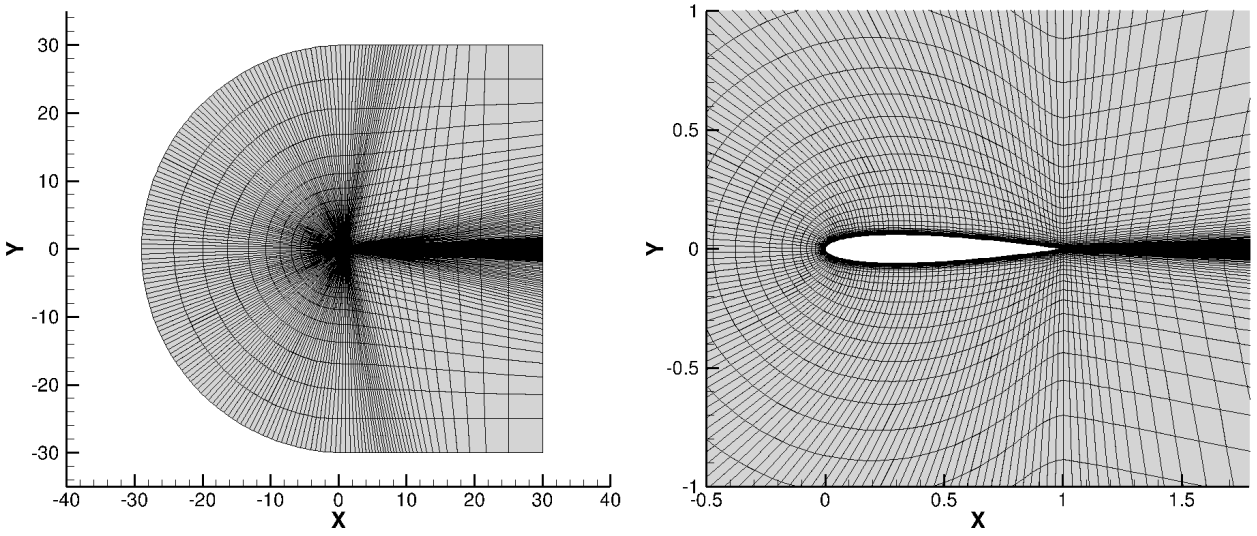
	C_1	C_2	CONV$_{\%}$
Full Dataset	1254	254	83.16

APPENDIX E MESHING STRATEGY FOR UNSTEADY DATASET

The unsteady dataset employs a *C-mesh* configuration, chosen due to the pitching motion being centered around low angles of attack, which leads to localized wake formation. The meshes in this dataset are generated using Cadence Fidelity *Pointwise* meshing software.

Each mesh consists of 192×64 cells in the i- and j-directions, respectively. In the i-direction, 128 cells surround the geometry, while 64 cells are distributed in the wake direction. Additionally, 64 cells are placed along the j-direction, starting with an initial wall-normal spacing of $\Delta s_{min} = 4.4 \times 10^{-6}$ chord. This ensures an initial $y^+ < 1$ for the dataset's fixed Reynolds number of $Re = 4.66 \times 10^6$.

To optimize cell distribution across the computational domain, nodes along the far-field boundaries, extending 30 chord lengths both upstream and parallel to the wake, are positioned using a combination of hyperbolic tangent distributions in the wake region and a uniform distribution across the inflow region.



(a) (b)

Figure E.1 View of (a) far- and (b) near-field areas used in unsteady dataset meshes

APPENDIX F RESULTS FOR UNSTEADY SURROGATE MODEL

Using the development framework described in this research, a surrogate model based on the MLP architecture is developed to predict the unsteady aerodynamic coefficients of pitching airfoils. The surrogate model captures hysteresis loops in the aerodynamic coefficients that occur when flow physics reach their outer limit cycle. Derived from the aforementioned framework, the architecture detailed in Table F.1 is employed, and training is conducted on the dataset described in Appendix D with an 80–10–10% split for training, validation, and testing. The model inputs include the airfoil's maximum thickness (t), initial angle of attack (α_0), pitching amplitude (A), and pitching frequency (ω). The outputs consist of three vectors, each containing 45 values representing the hysteresis loops for C_L, C_D, and C_M.

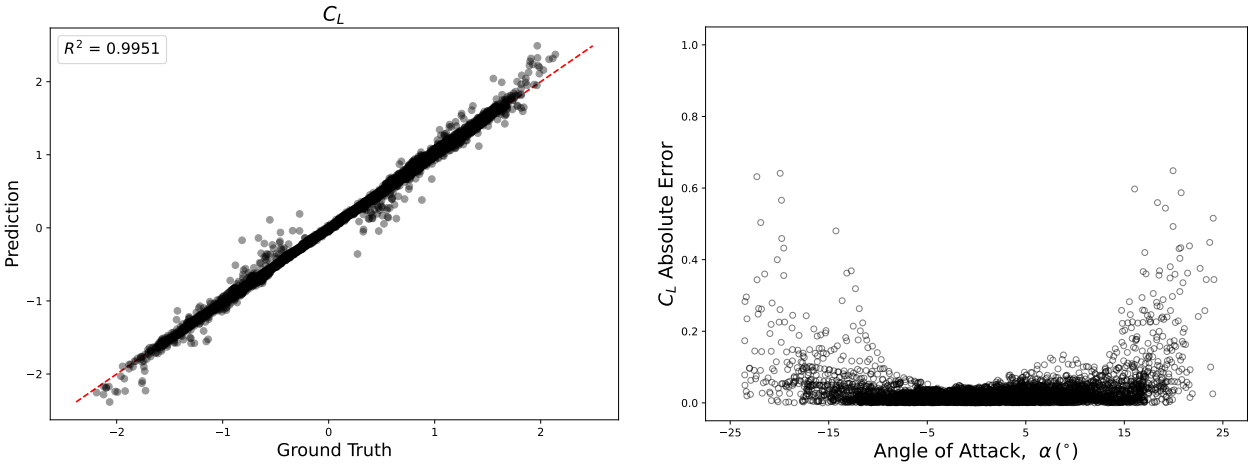
Table F.1 Architecture for unsteady model

Layers	Configuration	Activation Function
L_1	4 inputs (t, α_0, A, ω)	–
L_2	256 neurons	ReLU
L_3	256 neurons	ReLU
L_4	3 output vectors ($C_{L_{1-45}}$, $C_{D_{1-45}}$, $C_{M_{1-45}}$)	–

Using the random 10% testing split, the model's inference performance is evaluated. Errors across this dataset are summarized in Table F.2. Errors are smaller for C_D and C_M than for C_L, but the R^2 value for C_L predictions is higher, indicating that despite larger absolute errors, the model captures more complex trends in C_L input-output relationships more effectively. Furthermore, as illustrated in Figure Figure F.1, and similarly observed in the steady model, large errors are concentrated in underrepresented regions and in areas where stall typically occurs. This suggests that increasing database sampling in these critical regions is essential for performance improvement. These trends are further highlighted by the model's predictions in the hysteresis polar plots shown in Figures F.2–F.4.

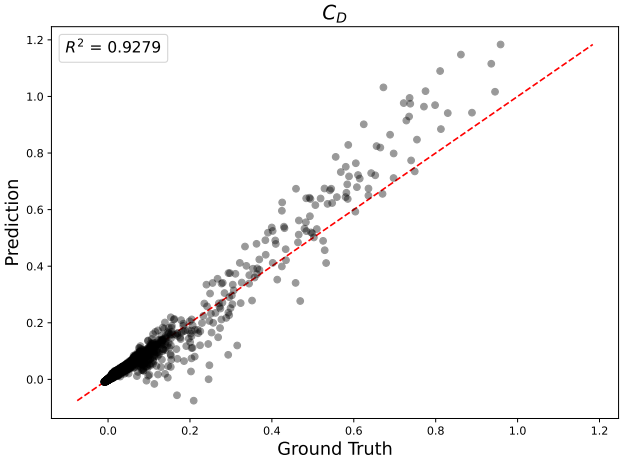
Table F.2 Summary of unsteady model inference error

	MAE	RMSE	|max|	R^2
C_L	0.0292	0.0571	0.6485	0.9951
C_D	0.0054	0.0221	0.3373	0.9279
C_M	0.0092	0.0266	0.4915	0.9433

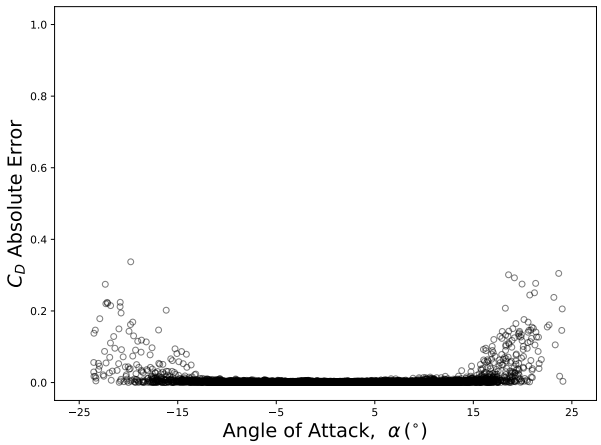


(a)

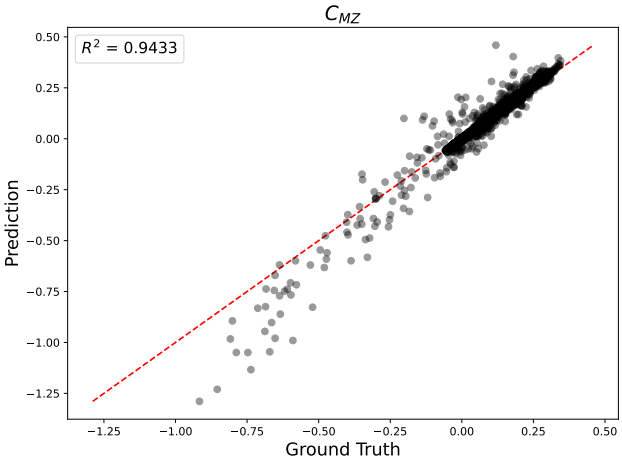
(d)



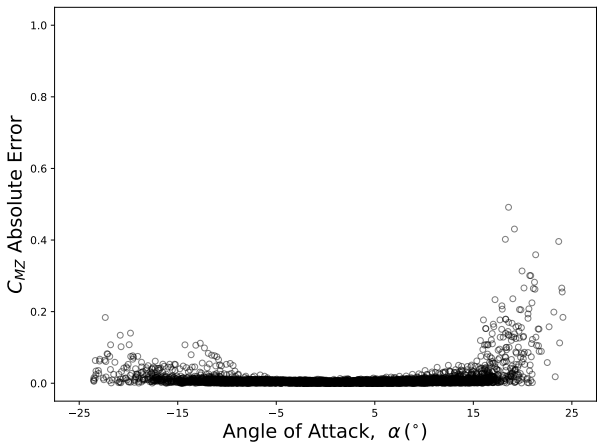
(b)



(e)

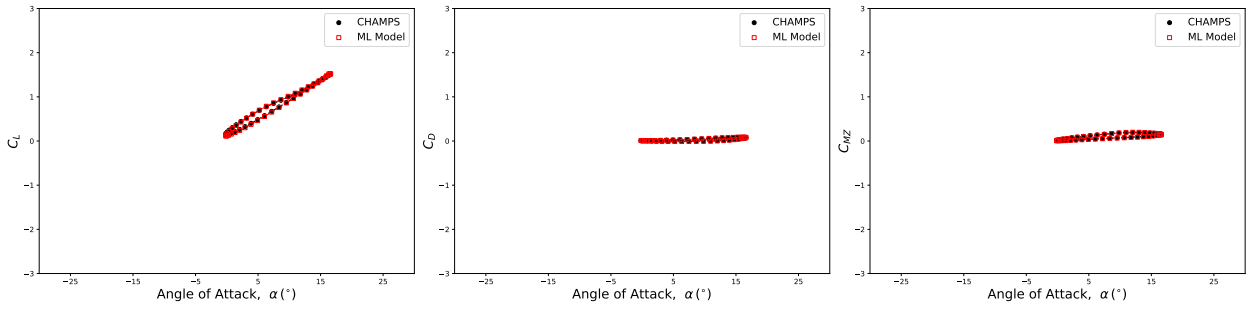


(c)

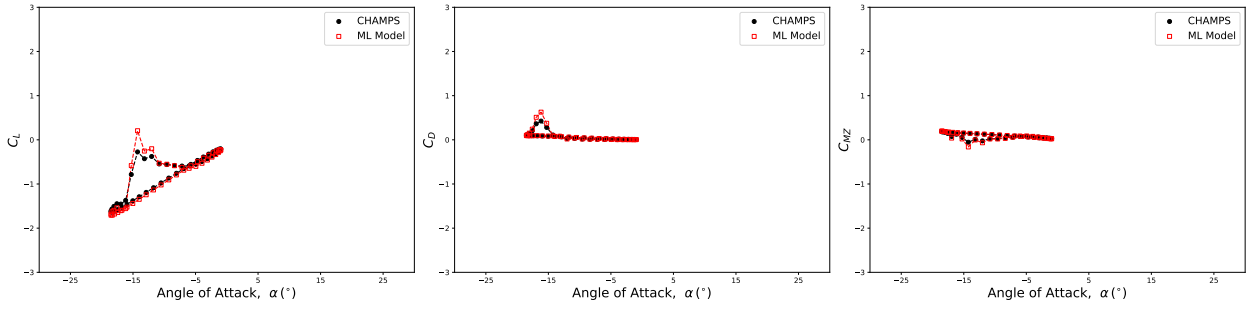


(f)

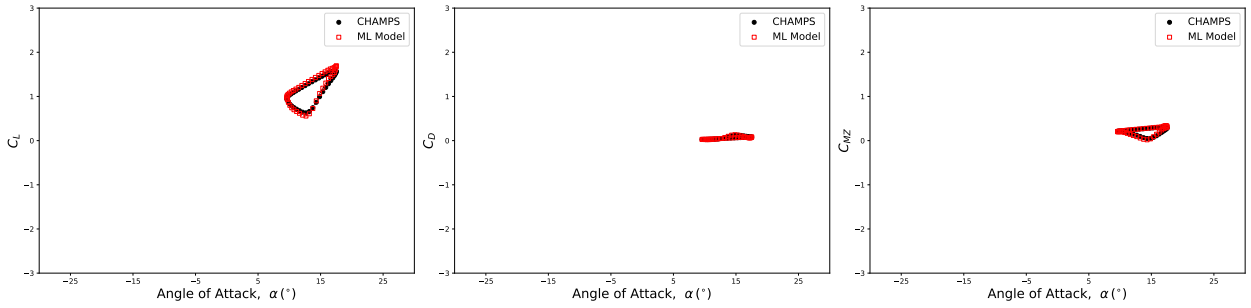
Figure F.1 Parity plots for (a) C_L, (b) C_D, and (c) C_M unsteady model predictions with dashed red line representing ideal parity line ($y = \hat{y}$) and model absolute error for (d) C_L, (e) C_D, and (f) C_M across angles of attack



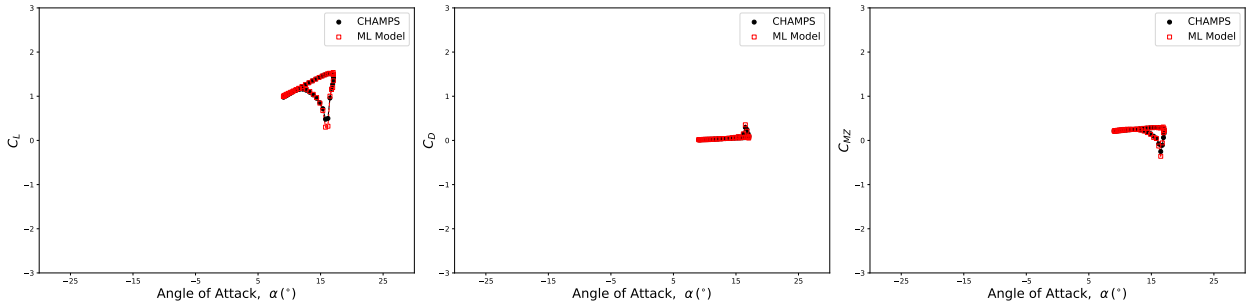
(a) $t = 12.12\%$, $\alpha_0 = 8.21°$, $A = 8.40°$, and $\omega = 16.07$ rad/s



(b) $t = 17.43\%$, $\alpha_0 = -9.74°$, $A = 8.80°$, and $\omega = 11.69$ rad/s

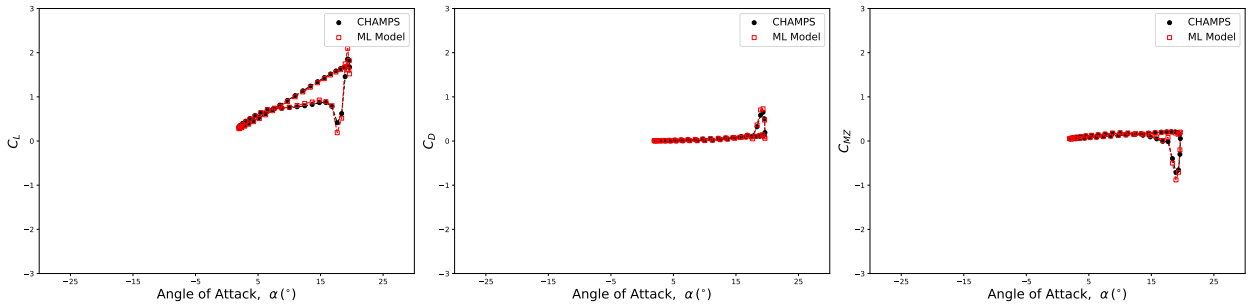


(c) $t = 14.50\%$, $\alpha_0 = 13.55°$, $A = 3.97°$, and $\omega = 15.14$ rad/s

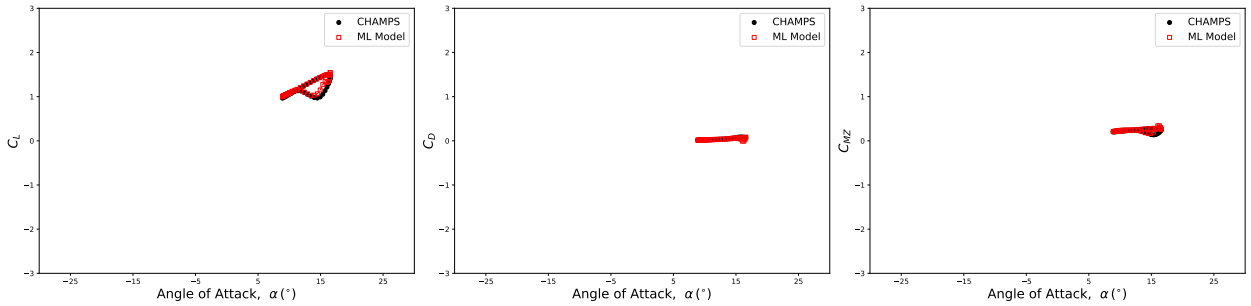


(d) $t = 10.68\%$, $\alpha_0 = 13.06°$, $A = 4.01°$, and $\omega = 7.21$ rad/s

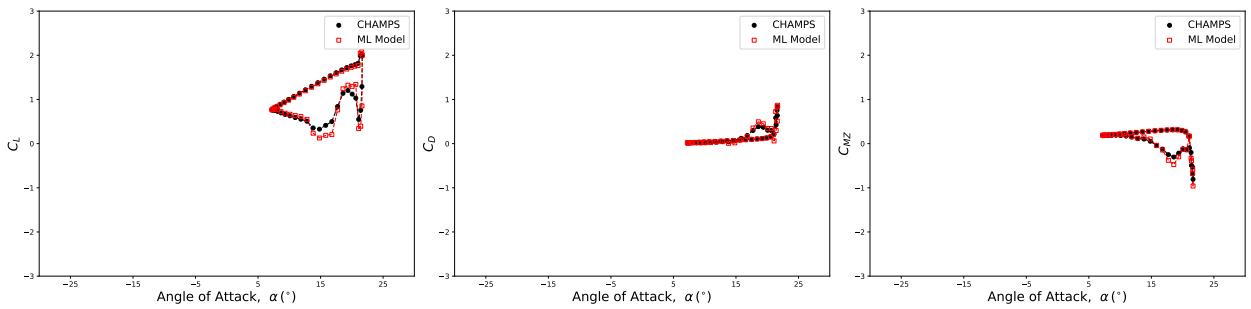
Figure F.2 Unsteady model predictions of hysteresis polar plots (series 1)



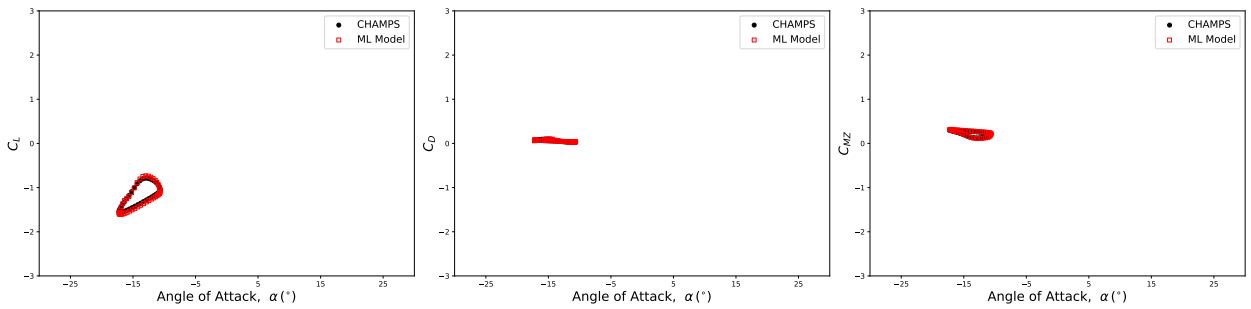
(a) $t = 16.59\%$, $\alpha_0 = 10.79°$, $A = 8.86°$, and $\omega = 11.28$ rad/s



(b) $t = 11.57\%$, $\alpha_0 = 12.74°$, $A = 3.84°$, and $\omega = 7.18$ rad/s

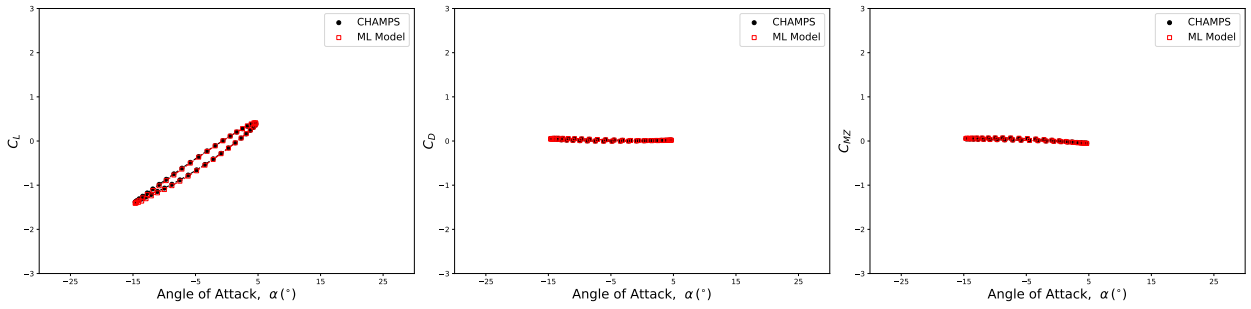


(c) $t = 10.72\%$, $\alpha_0 = 14.44°$, $A = 7.22°$, and $\omega = 13.98$ rad/s

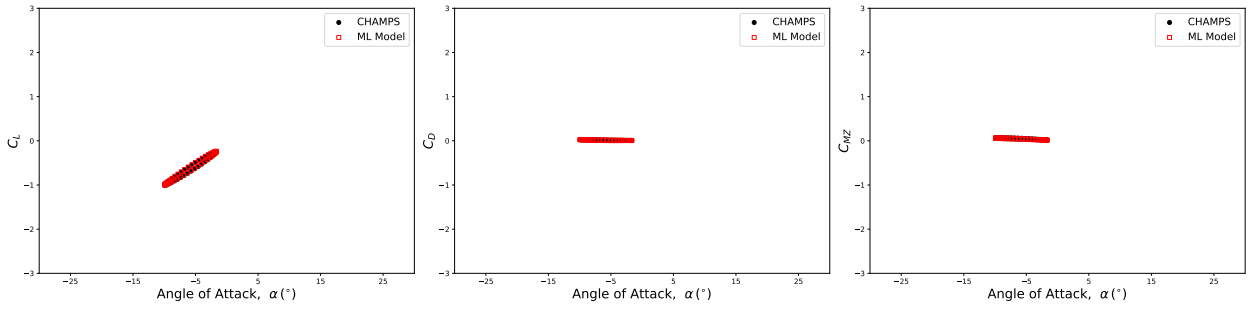


(d) $t = 11.38\%$, $\alpha_0 = -13.94°$, $A = 3.30°$, and $\omega = 14.89$ rad/s

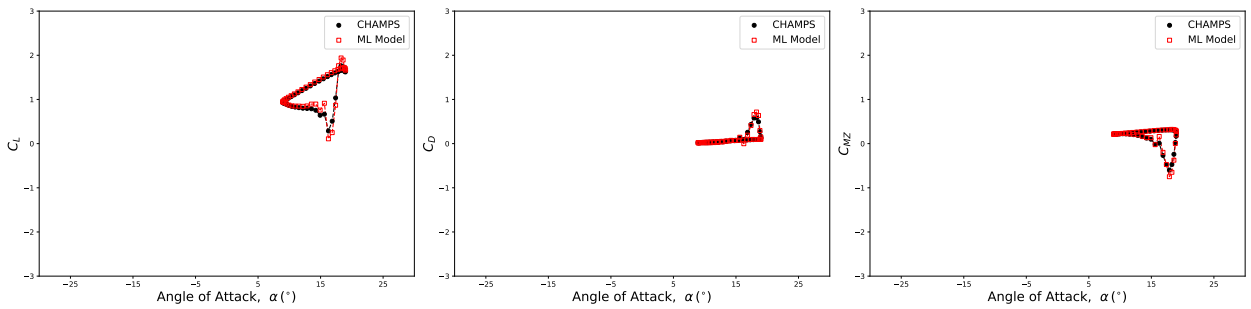
Figure F.3 Unsteady model predictions of hysteresis polar plots (series 2)



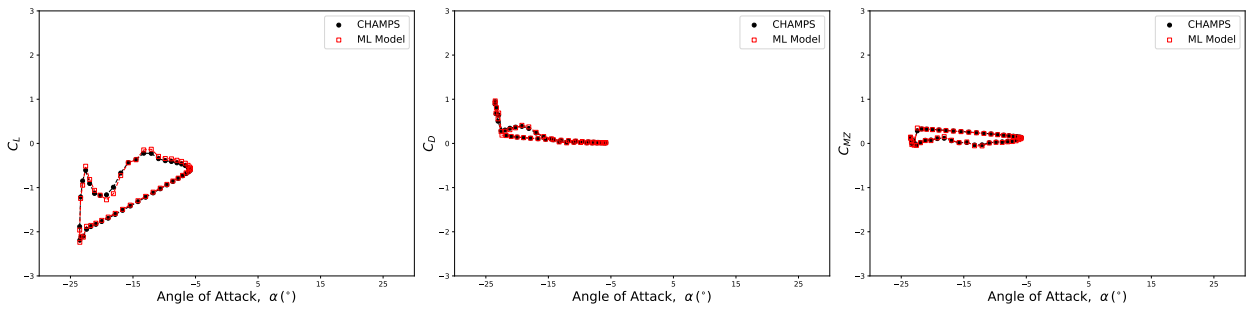
(a) $t = 8.63\%$, $\alpha_0 = -5.00°$, $A = 9.66°$, and $\omega = 8.43$ rad/s



(b) $t = 14.51\%$, $\alpha_0 = -5.80°$, $A = 4.14°$, and $\omega = 10.60$ rad/s



(c) $t = 7.95\%$, $\alpha_0 = 13.97°$, $A = 5.03°$, and $\omega = 16.14$ rad/s



(d) $t = 14.62\%$, $\alpha_0 = -14.67°$, $A = 8.87°$, and $\omega = 16.90$ rad/s

Figure F.4 Unsteady model predictions of hysteresis polar plots (series 3)